Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Electric Electronic and Communication Engineering (38.th cycle)

Low Latency, Network Resilience, and Accessibility in Networked Music Performance

Leonardo Severi

* * * * *

Supervisors
Prof. Cristina Rottondi, Supervisor
Prof. Andrea Bianco, Co-supervisor

Politecnico di Torino
May 21, 2026

I hereby declare that, the contents and organisation of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

. . . . . .

Leonardo Severi

Turin, May 21, 2026

Summary

Networked Music Performance (NMP) is an interaction between geographically distributed musicians who perform together in real time over telecommunications networks. NMP systems must satisfy stringent technical requirements that distinguish them from conventional videoconferencing applications, including the minimization of latency and the preservation of high audio quality. Beyond audio, musicians interact through visual cues, primarily represented by gestures, which are essential for coordination and musical expression.

This thesis presents several solutions attempting to make NMP practical for every kind of user, exploring technologies from dedicated hardware to accessible software platforms. At the hardware level, a system based on FPGA technology is presented, featuring a custom processor architecture designed for ultra-low-latency audio handling. This specialized approach demonstrates how dedicated hardware can minimize local processing delays. Complementing this, software-based solutions are explored, including web-based implementations that enable users without technical expertise to participate in high-quality NMP sessions through intuitive interfaces. The practical viability of these systems was validated through a real distributed concert between Turin and Wrocław, demonstrating that current technology can support actual artistic performances across international distances.

Network impairments pose significant challenges to audio quality. To support research addressing these issues, the thesis introduces DUST, a dataset containing UDP audio traces that captures fine-grained information about packet arrivals and playback conditions under diverse network configurations. Building on this foundation, packet loss concealment techniques are assessed for both MIDI and sampled audio. For MIDI, different configurations of MIDI streaming over Real-Time Protocol are benchmarked. For sampled audio, a sparse linear prediction algorithm achieves effective reconstruction with minimal computational requirements suitable for embedded hardware.

Beyond audio transmission, the thesis addresses visual communication essential to musical interaction. A Virtual Reality-based approach is presented where conductor gestures are tracked and transmitted to remote musicians as avatar representations, achieving lower bandwidth than video streaming, addressing the limitation of traditional audio-focused NMP systems that neglect visual conducting cues.

For blind and visually impaired musicians unable to perceive visual cues, haptic feedback offers an alternative. Through interviews with musicians, specifications are developed for wearable devices that translate conductor movements into vibrotactile patterns, with body placement varying according to instrument requirements. The accessibility discussion extends to musicians with auditory or mobility impairments, exploring multimodal feedback and personalized rendering as enabling technologies.

Acknowledgements

Acknowledgements deserve to be written in Italian.

Seppur non sia solito scrivere questo genere di sezioni, sembra che sia buona norma farlo al termine di un percorso del genere e farò del mio "meglio" per ringraziare le – sicuramente non tutte quelle che dovrei – persone che lo hanno agevolato, cercando di mantenere almeno un briciolo di serietà e scusandomi in anticipo per l'esercizio del dono della sintesi.

La prima persona che devo ringraziare è Matteo, ad alcuni noto come il "Signor Bro", che per primo mi ha fatto conoscere il gruppo, il mondo NMP e il progetto MEVO, all'epoca in fase embrionale. In tutto questo tempo il Signor Bro mi ha fatto un po' da apripista, consentendomi di trarre vantaggio da esperienza non maturata direttamente da me. Si potrebbe menzionare anche la condivisione dell'aspetto ludico sul luogo di lavoro, ma forse è bene non peggiorare ulteriormente la nostra immagine.

Inoltre, chiaramente merita un ringraziamento chi mi ha dato concretamente l'opportunità, accolto nel gruppo, creduto in me e fatto crescere, ossia Cristina e Andrea.

Infine, devo ovviamente ringraziare tutta la mia famiglia: nello specifico i miei genitori per i sacrifici fatti in precedenza per permettermi di arrivare fin qui e anche, insieme anche a Deanna e Delfi, per avermi trasmesso la curiosità che mi ha portato a scegliere di fare un dottorato. Menzione d'onore per Alex per il sostegno nelle (non rare) giornate grigie di questo periodo, alcune fatte di cose da scrivere sotto scadenza, di risultati non sempre soddisfacenti, per avermi sopportato quando non facevo altro che lamentarmi, ma soprattutto per tutto il resto che non ha sicuramente a che vedere con questa tesi e con il percorso qui raccontato.

Grazie a tutti.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Networked Music Performance (NMP) allows musicians in different locations to play together as if they shared the same room, relying on real-time audio transmission over the Internet [136]. To achieve this effect, NMP systems must keep end-to-end latency within roughly 30 ms [191] and preserve audio quality far beyond what videoconferencing applications typically offer. This thesis investigates technologies aimed at making NMP practical for broader adoption, addressing challenges that have historically confined these systems to expert users.

The field emerged in the early 2000s with the spread of Internet connectivity and Voice over IP services. Specialized software solutions established the technical foundations during the first decade of the century, introducing dedicated protocols and architectures optimized for musical performance. The COVID-19 pandemic intensified interest in remote musical collaboration, yet widespread adoption of these specialized NMP systems remained limited. During this period, users often gravitated toward familiar videoconferencing tools over purpose-built NMP applications, despite the latter offering superior audio performance when properly configured. This preference revealed fundamental accessibility barriers: existing specialized systems required technical expertise that most musicians lacked, creating a significant gap between the theoretical capabilities of NMP technology and its practical accessibility.

The research presented here began from this observation. Three interconnected problems emerged as central concerns: minimizing end-to-end latency, maintaining audio quality despite network impairments such as jitter and packet loss, and lowering the technical barriers to entry.

1.1 Motivations

The guiding question of this research was whether an NMP system could be built that satisfies professional audio requirements while remaining accessible to

non-expert users over standard residential Internet connections.

The latency threshold that defines acceptable NMP performance corresponds roughly to the propagation delay experienced by two musicians standing a few meters apart. Beyond this limit, the slight asynchrony becomes perceptible and disrupts the natural feedback loop that musicians rely on to stay coordinated. Videoconferencing tools operate with delays sufficient for spoken conversation, where turn-taking masks moderate latency, but entirely inadequate for ensemble playing where simultaneity matters. This gap explains why NMP demands a distinct technological approach rather than incremental improvements to existing platforms.

Yet during the pandemic, when physical rehearsal spaces closed and remote collaboration became necessary, musicians used to choose videoconferencing tools over dedicated NMP applications. The pattern held even in conservatories and music schools, where audio quality is supposedly paramount. The reasons were pragmatic: videoconferencing tools required no installation beyond what institutions had already deployed, no configuration beyond clicking a link, and no learning curve beyond what users had already climbed for other purposes. NMP applications, by contrast, asked users to install specialized software, tune some parameters to make the software ready to operate in the specific setup, and troubleshoot possible issues (mostly related to drivers) that presuppose technical knowledge most performers do not have and should not need.

This preference for convenience over performance pointed to a design problem rather than a user failing. If musicians avoided superior tools because those tools were difficult to use, then the path forward lay in reconciling low latency with low friction. The question decomposed accordingly: could web technologies, familiar to anyone who has used a browser, deliver audio quality competitive with native applications? And where web constraints proved insurmountable, could a dedicated hardware device be made simple enough that non-technical users would accept it?

A second concern arose from the nature of Internet transmission itself. Real-time audio transmission relies on protocols (RTP-like/UDP) designed for speed rather than reliability: data may arrive late, out of order, or not at all. When audio information fails to arrive before its scheduled playback time, the result is an audible glitch i.e., a gap or click that interrupts the musical flow. Real-time streaming systems attempt to absorb timing variations by holding incoming data in a buffer, briefly before playing it back, but waiting longer (enlarging the buffer) means higher latency, recreating the problem they were meant to solve. The trade-off between jitter resilience and responsiveness has no universal optimum; it depends on network conditions that vary across connections and fluctuate over time.

Packet Loss Concealment offers a complementary strategy: rather than buffering against the possibility of late arrivals, the receiver synthesizes replacement samples when packets fail to appear. Effective concealment requires predicting what the missing audio would have sounded like based on what came before. Any algorithm intended for NMP must handle this task while running fast enough to meet real-time

constraints, potentially on hardware as modest as a single-board computer. The computational budget is particularly stringent: processing must complete within the duration of the receiver buffer (usually few milliseconds).

The final research direction extended beyond the general population of musicians to those who face additional barriers. Musical interaction is not purely auditory: orchestral performers rely on visual cues from a conductor to coordinate entries, shape dynamics, and maintain tempo. Standard NMP systems transmitted only sound, leaving conducted ensembles without the visual cues important for maintaining ensemble coordination. Transmitting video alongside audio addressed this gap in principle, but camera-based solutions introduced audio-video desynchronization due to the additional encoding/decoding latency required for video transmission.

For Blind and Visually Impaired musicians, even a low-latency video stream provides no benefit. Conducting gestures encode information through spatial movement (e.g., the height of a beat) that has no natural auditory equivalent. Translating this gestural vocabulary into a non-visual modality requires identifying which aspects of conducting carry essential information and designing feedback that conveys them without overwhelming the performer or interfering with instrument technique. The challenge is not merely technical but perceptual: whatever system emerges must be learnable, interpretable, and unobtrusive enough for practical use.

1.2 Research Contributions

The work presented in this thesis follows a progression from foundational system design toward increasingly specialized applications. The starting point was the development of tools capable of supporting the rest of the research. At the hardware level, a system based on FPGA technology implements a Transport Triggered Architecture achieving local audio processing latency below one millisecond, leaving most of the tolerable delay budget available for network traversal. Complementing this specialized approach, software-based solutions are explored that enable users without technical expertise to participate in high-quality NMP sessions. A web-based implementation is envisioned where users join sessions directly from a browser, routing uncompressed audio samples through WebRTC [9] data channels rather than media streams to avoid the compression and buffering imposed by standard browser audio paths. Beyond the web-based approach, a complementary command-line tool was also implemented and subsequently validated. The proposed MEVO system runs on Raspberry Pi devices. Its practical viability was demonstrated through a real distributed concert connecting musicians in Turin and Wrocław, achieving latency levels suitable for remote musical collaboration across over a thousand kilometers despite minor perceptible degradation.

Once a working platform existed, attention shifted to handling network conditions. Real-time audio transmission inevitably encounters jitter defined as the

variability in packet arrival times. When packets arrive too late to be processed, they must be discarded, resulting in audible glitches. Studying these phenomena required realistic data, which led to the creation of DUST, a collection of packet traces recorded during actual NMP sessions under varied network conditions. The DUST dataset captures not only arrival times but also playback buffer states and audio clock behavior. This enables simulation of partial packet losses and precise drift estimation, aspects typically neglected in existing network trace collections. Following this infrastructure development, a packet loss concealment algorithm based on sparse linear prediction was implemented. The algorithm reconstructs missing samples by fitting a sparse linear predictive model to recent audio history, selecting predictor lags in a way that captures periodicity without excessive computation. Implementation on embedded hardware demonstrates sub-millisecond processing times, meeting real-time requirements while maintaining reconstruction quality comparable to traditional approaches.

The final direction addressed users for whom audio alone does not suffice. Orchestral musicians rely on visual cues from a conductor, cues that standard NMP systems ignore entirely. A Virtual Reality headset was repurposed to track hand and head movements, transmitting pose data to remote players who see the conductor rendered as an avatar. Because only skeletal pose data travels over the network rather than video frames, bandwidth requirements decrease to a few kilobits per second compared to the megabits required by standard video streaming solutions, reducing of two to three orders of magnitude. For Blind and Visually Impaired musicians, even an avatar remains inaccessible. Interviews with performers led to specifications for wearable devices that convert conductor gestures into vibrotactile patterns, with placement on the body chosen to avoid interference with instrument technique.

1.3 Outline

The remainder of this thesis is structured as follows. Chapter 2 provides background knowledge to aid comprehension of subsequent chapters. The chapters (Chapters 3 to 11) present the scientific contributions, each corresponding to a published article or a demonstration during a conference. Finally, chapter 12 concludes the thesis with a summary of the main findings and directions for future research.

The scientific contributions reported in this thesis have been published to peer-reviewed conferences and journals (except for chapter 5, which backed a demonstration during the International Symposium of the Internet of Sounds 2023) as follows:

(i) Chapter 3: *FPGA-based Low-Latency Audio Coprocessor for Networked Music Performance* [21]. This work represents the first attempt to implement an

NMP solution by bypassing operating system overhead and general-purpose CPU limitations. It proposes a hardware solution based on an FPGA with the capability to implement audio mixing, digital signal processing (DSP), and potentially packet loss concealment (PLC) with hardware acceleration.

(ii) Chapter 4: *Pristine Quality Networked Music Performance System for the Web* [195]. This chapter describes a unified NMP environment that supports both native/embedded and web-based implementations, enabling high-fidelity uncompressed audio transmission while maintaining ease of use through web interfaces. The work eventually led to the development of the MEVO system.

(iii) Chapter 5: *Demonstration of a Networked Music Performance Experience with MEVO* [217]. This chapter presents a practical demonstration of the MEVO system in a distributed concert between Turin (Italy) and Wrocław (Poland), showcasing the system's capabilities in a real-world performance scenario.

(iv) Chapter 6: *Introducing DUST: a Dataset of real-time UDP Sound packet Traces* [218]. DUST is a publicly available dataset of UDP audio packet traces collected by the MEVO system under various network conditions, designed to support research on packet loss concealment, jitter characterization, and clock drift estimation. The dataset enables assessment of PLC techniques under real-world network conditions.

(v) Chapter 7: *Assessment of Recovery Journal-Based Packet Loss Concealment Techniques for Low-Latency MIDI Streaming* [216]. This chapter lays the foundation for managing MIDI audio in NMP. It assesses techniques to stream MIDI over the Real Time Protocol (RTP), focusing on bandwidth usage and evaluating it with respect to different loss recovery approaches.

(vi) Chapter 8: *Sparse Linear Prediction for Packet Loss Concealment in Networked Music Performances* [215]. This chapter introduces a sparse linear prediction algorithm for real-time audio packet loss concealment (PLC), achieving sub-millisecond processing times on embedded hardware while maintaining reconstruction quality comparable to traditional autoregressive models. This research was motivated by the need to develop a practical PLC technique for the MEVO system.

(vii) Chapter 9: *Towards an Inclusive Framework for Remote Musical Education and Practices* [192]. This chapter moves the focus to accessibility and presents a comprehensive framework for inclusive networked music performance systems that address the needs of disabled users. It reviews technological solutions including haptic feedback, motion tracking, and immersive audio rendering, and proposes their integration within NMP systems to support remote musical education for students with visual, auditory, and mobility impairments.

(viii) Chapter 10: *Remote Orchestral Conduction via a Virtual Reality System* [219]. This research lays the foundation for investigating the transmission of conductor gestures to remote musicians, thus including non-audio information in the capabilities of NMP systems. The chosen technology is Virtual Reality (VR), which permits the use of off-the-shelf hardware to provide both immersivity and motion tracking.

(ix) Chapter 11: *Designing Haptic Feedback Stimuli to Convey Gestures of an Orchestra Conductor to Visually-Impaired Musicians* [250]. This chapter continues the theme of accessibility, exploring how haptic feedback systems can convey orchestral conducting gestures to blind and visually impaired musicians, with potential applications extending to remote performance contexts, deriving specifications for wearable devices through interviews with musicians and analysis of conducting practices.

Chapter 2

Background

This chapter provides essential background knowledge, which might help the reader understand the topics presented in the next chapters. It covers:

(i) the physical and perceptual definitions of sound and music (Section 2.1);

(ii) digital audio as sampled representation of physical sound (Section 2.2);

(iii) the MIDI protocol as an alternative approach to digitally represented audio (Section 2.3);

(iv) network fundamentals, specifically the Internet protocol stack (Section 2.4);

(v) existing Networked Music Performance systems (Section 2.5);

(vi) haptic feedback and its potential applications for accessibility (Section 2.6);

(vii) the Extended Reality paradigm and its applications (Section 2.7).

2.1 Sound and Music

To understand the work presented in this thesis, it is useful to recall the elementary concepts of sound and music. Sound is defined [4] as:

1. Oscillation in pressure, stress, particle displacement, particle velocity, etc., propagated in a medium with internal forces (e.g., elastic or viscous), or the superposition of such propagated oscillation.

2. Auditory sensation evoked by the oscillation described in point 1.

The first definition is physical and objective; the second is perceptual, rooted in the faculty of hearing. A sound wave propagating through air carries physical

properties, namely frequency, amplitude, phase, and waveform; the auditory system translates these into perceptual qualities such as pitch, loudness, and timbre.

Music is the organization of sounds to convey expressive content. Musical sound encompasses both periodic, pitched tones and irregular, noise-like components such as breath noise, bow scraping, or cymbal wash, a distinction well established in the audio analysis literature [212]. Because music is ultimately defined by human experience, any technical system intended to capture or reproduce it must be designed around perception. The human auditory system is sensitive to the approximate range of 20 Hz to 20 kHz, which directly determines the bandwidth requirements of music-oriented systems.

The systems considered in this work are electronic. Their fundamental principle is the transduction of mechanical oscillations into electrical signals via sensors such as microphones, and their subsequent conversion back into mechanical vibrations via actuators such as speakers. Working in the electrical domain offers greater flexibility than direct mechanical manipulation, and digital signal processing, which operates on discrete samples of the signal, enables sophisticated manipulation and transmission of audio information.

2.2 Sampled Audio

Sampled audio is the digital representation of sound created by measuring the continuous analog sound wave at regular intervals (sampling), and converting these measurements into numerical values with finite precision (quantization). This process transforms the smooth, continuous nature of acoustic sound waves into discrete digital data that computers can store, process, and reproduce. An analog-to-digital converter (ADC) transforms analog sound waves into digital data.

1. A sensor captures sound waves and trasduces them into an analog electrical signal
2. The ADC samples this signal thousands of times per second
3. Each sample captures the amplitude (loudness) of the wave at that precise moment
4. These measurements are converted into numbers (digital data) represented in binary form

The amount of samples to acquire per time unit is called sample rate. According to the Nyquist theorem, the sample rate must be at least twice the highest frequency you want to capture. Since human hearing extends to about 20 kHz, a sampling rate of at least 40 kHz is required. For instance, a 44.1 kHz sample rate can accurately reproduce all audible frequencies and is a standard in music distribution.

The number of bits used to represent each sample's amplitude is called the "*bit depth*" and determines the dynamic range and precision. Higher *bit depth* means more accurate representation of quiet sounds and greater dynamic range between the softest and loudest sounds.

The electronic device that is in charge of converting analog audio (voltage) to a binary numeric representation is the ADC (Analog-to-Digital Converter). Once audio is stored digitally, we need a DAC (Digital-to-Analog Converter) to convert it back to analog form for playback through speakers or headphones. DACs and ADCs are the main components of sound cards and are present in every digital system capable of recording (ADC) or playing (DAC) audio.

Pulse Code Modulation (PCM) is the most common method for encoding sampled audio. It is the natural input for DACs and output for ADCs, and it is the result of the sampling, linear quantization, and binary encoding of audio. Pure PCM audio can be found in WAV files (Windows), Audio CDs and few other software/hardware medias.

2.3 MIDI

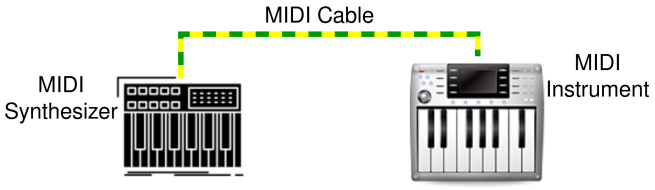


Figure 2.1: Block diagram of a generic MIDI setup. A *MIDI instrument* (left) converts a performer's physical action into symbolic MIDI messages; the messages travel over the MIDI transport (centre) to a *MIDI synthesiser* (right), which interprets them and produces the audio signal. The two roles are logically distinct even when embedded in the same physical device, such as a modern keyboard workstation.

The Musical Instrument Digital Interface (MIDI) is a standardized protocol for representing musical performance information as electronic data [159]. Unlike audio recordings, MIDI encodes the actions performed during a musical performance, such as note onsets, note releases, and control parameters, rather than the resulting sound itself. Fig. 2.1 depicts a typical MIDI configuration, where a MIDI instrument transmits messages through a MIDI cable to a synthesizer that interprets these commands and generates the corresponding audio output.

The MIDI protocol operates through a stream of messages, each typically consisting of 2 or 3 bytes. Each message modifies the state of the receiving device:

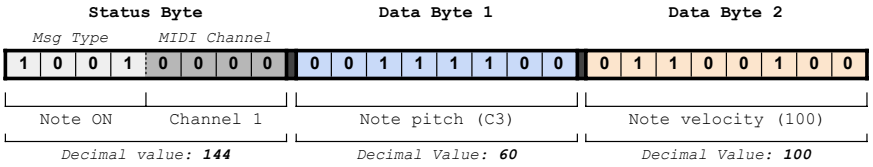


Figure 2.2: Example of a MIDI message (Note On message referring to the C3 note onset with velocity 100 on MIDI channel 1).

for instance, a Note On message triggers the onset of a specific note with a given velocity, as illustrated in Fig. 2.2.

2.4 Network

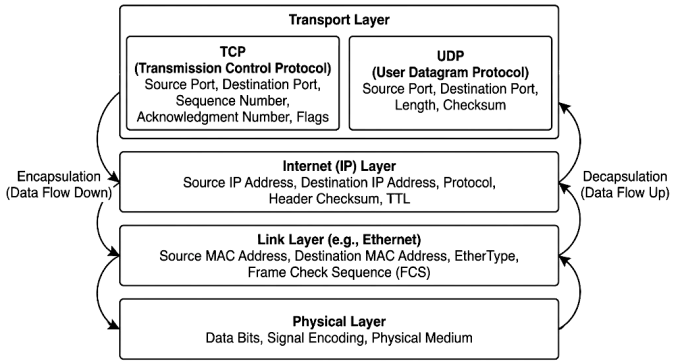


Figure 2.3: The four layers of the IP stack showing data flow between layers.

Computer networks enable data exchange between devices, but this task involves solving multiple distinct problems that operate at different scales. The IP stack addresses this by decomposing network communication into layers, where each layer solves a specific set of technical challenges using the services provided by lower layers. Although some special cases (e.g., the use of Virtual Private Networks) modify the predefined stack structure, from bottom to top, the logical layers are: (*i*) *physical*, (*ii*) *link*, (*iii*) *Internet*, and (*iv*) *transport* (see Fig. 2.4). Each layer solves progressively higher-level problems in network communication.

The *physical layer* is responsible for converting bits into physical signals, such as electrical voltage, radio waves, or light pulses that can travel across physical media. Building on this foundation, the *link layer* provides delivery of data packets between directly connected devices. Protocols operating at the link layer handle framing,

error detection, and medium access control, ensuring reliable packet delivery over the physical connection.

However, the *link layer* has a fundamental limitation: it can only deliver packets within a single network segment where devices share the same physical or logical medium (e.g., devices connected to the same Ethernet switch or Wi-Fi access point).

The *Internet layer*, implemented by the Internet Protocol **IP** [112, 54], overcomes this limitation by introducing a hierarchical addressing scheme and routing mechanisms that enable communication across multiple interconnected networks. It accomplishes this by assigning globally unique addresses to devices (IP addresses) and defining routes that specify how packets should traverse intermediate networks to reach their destination. This enables the creation of large-scale networks by aggregating smaller local networks, allowing devices across the globe to communicate.

The *transport layer* sits atop the Internet layer and serves a different purpose: it multiplexes communications between different processes running on the same device. Since multiple processes may simultaneously use the network, the *transport layer* provides port numbers to distinguish between different communication endpoints. Additionally, it offers two fundamentally different service models to processes.

The first model, exemplified by the ***UDP*** protocol [243], is connection-less and preserves direct packet-level semantics. Applications using *UDP* have explicit control over individual packets but must handle potential packet loss, reordering, and size limitations imposed by the underlying layers. This simple approach is preferred when low latency is more critical than perfect reliability, as retransmitting lost packets would introduce unacceptable delays. The second model, primarily represented by the ***TCP*** protocol [62], provides a connection-oriented, reliable byte-stream abstraction. *TCP* implementations handle packet loss, reordering, and fragmentation transparently, presenting applications with a simple interface for reading and writing arbitrary amounts of data. While *TCP*'s reliability guarantees are beneficial for many applications, they come at the cost of increased latency and overhead.

The choice between *UDP* and *TCP* depends on application requirements. For data streaming applications, where timely delivery of recent data is more valuable than guaranteed delivery of old data, *UDP* is often the preferred choice. Indeed, a significant portion of the work presented in this thesis, which directly concerns real-time data streaming, assumes the use of the *UDP* protocol.

While the *transport layer* provides the fundamental communication primitives between processes, applications require higher-level semantics for specific tasks. Application-layer protocols, built atop transport protocols, provide these standardized semantics and enable interoperability between heterogeneous implementations. The World Wide Web exemplifies this: *HTTP* [70] operates over *TCP* connections to transmit requests and responses with standardized message formats. Clients send requests specifying methods (e.g., GET, POST) and resource identifiers (URLs), while servers respond with status codes and content. This standardization ensures

that different browsers and server implementations can communicate correctly, as all conforming implementations adhere to the same protocol specification.

2.5 NMP Systems

Networked Music Performance (NMP) systems leverage network technologies to connect musicians in geographically distant locations, enabling them to perform together in real time. This practice emerges at the intersection of telecommunications technology, computer music, and artistic performance, and has developed significantly since the 1990s with the advent of high-speed Internet connectivity. The primary objective is to overcome spatial barriers, enabling musical collaborations that would otherwise be impractical or prohibitively expensive.

The principal challenges in NMP systems concern latency, which can severely compromise synchronization and interaction between musicians. Acceptable mouth-to-ear delays are typically on the order of tens of milliseconds, with 30 ms being a commonly referenced threshold in the literature [191]. However, this value may need to be lower for musical genres requiring precise timing. Other critical factors include audio quality and connection stability, particularly with respect to jitter.

Despite these technical challenges, NMP systems have found applications across various contexts: distance music education, experimental artistic performances, and distributed concerts involving international ensembles.

State-of-the-art NMP solutions consist mainly of software frameworks, but some hardware approaches have also recently gained attention. For a thorough comparison of existing NMP solutions, the reader can refer to [191] and [263]. In the following, we provide a brief overview of the most widely adopted ones.

One of the first software-based NMP solutions, developed in the early 2000s, is Jacktrip [29], an open-source application that currently runs on most general-purpose Operating Systems (OSs). To achieve real-time musical interaction, it transmits uncompressed audio samples through the network using the User Datagram Protocol (UDP) at transport layer. Similarly, DIAMOUSES [7] is an open-source framework that streams uncompressed audio signals while also incorporating live video streaming.

SoundJack [104, 36], in contrast, prioritizes bandwidth efficiency by employing a compressed audio format. This enables integration of Packet Loss Concealment (PLC) techniques directly into audio codecs, at the price of increasing the mouth-to-ear latency due to the time overhead introduced by compression algorithms.

The inclusion of video streaming introduces further challenges, since video acquisition and encoding delays are typically higher than those required by audio signals, leading to misalignment between the two streams. To address this issue, LOLA [58] resorts to uncompressed video streaming, at the price of increasing the required bit-rate to a few Gbps.

The social distancing measures enforced during the SARS-CoV-2 pandemic generated significant interest in NMP applications, prompting rapid development in this field. In 2020, video streaming features were incorporated into Jacktrip and SoundJack, leveraging the low-latency streaming capabilities of the WebRTC paradigm [201] to provide web-based interfaces. This period also saw the emergence and popularization of several other NMP solutions, including Jamulus [72], JamKazam [117], SonoBus [42], and FarPlay [67], each offering different trade-offs between ease of use, latency, and audio quality.

While the aforementioned solutions rely entirely on general-purpose computing hardware, some approaches integrate dedicated hardware to further reduce latency in the acquisition and processing chain. Elk Audio [63] offers a complete setup for real-time remote performances, consisting of both hardware and software components. Their hardware audio interface, called Elk Bridge, is optimized for low-latency audio applications and designed to be used with a proprietary software called Elk LIVE.

A different hardware-assisted approach was pursued in [154], where a software environment running on a single-board computer (BeagleBone Black) leverages the Xenomai kernel extensions to implement real-time Digital Signal Processing (DSP) functionalities. Its performance was benchmarked in [153], showing that this is the only tested solution able to meet stringent jitter and latency requirements. At the more accessible end of the hardware spectrum, community-driven solutions based on the Raspberry Pi have also emerged: the JackTrip community published a reference design pairing the platform with a dedicated audio HAT to act as a self-contained JackTrip bridge [115], while the JackStreamer by NetworkSound [167] offers a comparable endpoint built around a USB audio interface. Both designs aim to decouple real-time audio I/O from the host OS while retaining a general-purpose kernel in the audio path.

2.6 Haptics

Haptics is the science of touch-based feedback. It has gained significant impact on how people interact with digital devices. The vibration when typing on a smartphone and the force feedback in gaming controllers are examples of tactile sensations aimed at carrying information to improve user experience. At its core, haptic feedback uses actuators, motors, or other mechanisms to generate vibrations, forces, or movements that users can feel. Modern implementations range from simple vibration alerts to complex systems that can simulate different textures, pressure, and nuanced resistance. Beyond entertainment and convenience, haptic technology holds potential for accessibility. For visually impaired individuals, haptic feedback can transform how they navigate digital interfaces. Tactile notifications and vibration patterns can convey information that would otherwise require visual confirmation, making smartphones and computers more intuitive to use without sight

[129]. Haptic-enabled devices can provide spatial awareness through vibration cues, helping users understand screen layout and navigate menus. Some innovative applications use haptic feedback to "translate" visual information into touch sensations, allowing users to feel graphs, maps, or images through varying vibration intensities and patterns. For people with hearing impairments, haptics offers an alternative communication channel-vibrations can alert users to notifications, calls, or alarms they might otherwise miss [74]. In educational contexts, haptic technology can make abstract concepts more tangible for students with different learning needs, providing a multisensory approach to learning.

2.7 Extended Reality

Extended Reality (XR) is an umbrella term that encompasses a spectrum of immersive technologies designed to merge physical and digital environments. The term includes Virtual Reality (VR), which creates fully synthetic environments for user immersion; Augmented Reality (AR), which superimposes digital information onto real-world views; and Mixed Reality (MR), which enables interaction between physical and virtual objects within a shared space.

XR technologies have found applications across multiple domains, including education [222], entertainment, and manufacturing [23]. In medical training [145], VR environments allow practitioners to simulate procedures in controlled settings. The development of XR systems has been facilitated by advances in computing power, display technologies, and spatial computing algorithms, which have progressively improved the fidelity and responsiveness of immersive experiences. Spatial audio technologies play a complementary role in XR environments by providing three-dimensional sound cues that enhance the sense of presence and spatial awareness, contributing to more coherent and realistic user experiences.

Chapter 3

FPGA-based Low-Latency Audio Coprocessor for Networked Music Performance

This chapter addresses the challenge of minimizing local processing delays through dedicated hardware. The stringent latency requirements of NMP suggested that general-purpose computing might be fundamentally inadequate. This chapter explores that hypothesis, investigating whether dedicated hardware could bypass the delays inherent in operating systems and conventional processors. The FPGA-based coprocessor presented here represents a first attempt to push local audio processing latency to its theoretical minimum.

3.1 Introduction

NMP leverages an ultra-low latency audio stream over a telecommunication network, where maintaining stable real-time communication between users presents several technical challenges. Meeting the latency requirements established in Section 1.1 is not straightforward, as several bottlenecks exist throughout the audio acquisition, processing, and transmission chain.

To reduce latency, different software and hardware solutions have been proposed. However, since latency requirements are very severe, a purely software-based approach is not always the best option. An alternative solution, described also in [191], is the use of a system based on a Field-Programmable Gate Array (FPGA), where audio packets are completely handled by dedicated hardware. In this way, the sources of additional latency that exist in a general-purpose architecture can

be bypassed or accelerated.

The solution considered in this chapter is based on the preliminary work presented in [57] and [20]. It consists of a dedicated hardware system designed to ensure extremely low latency and is created by combining a software approach and a FPGA-based one. In particular, the design of an Application-Specific Instruction set Processor (ASIP) implemented by taking advantage of a Transport Triggered Architecture (TTA) is proposed. This architecture is particularly suitable for the NMP application domain, as it can be exploited to process audio samples with a reduced amount of added latency.

To understand the possible advantages of this kind of approach, some other solutions will be first examined in Sec. 3.2. In Sec. 3.2, an introduction to the TTA processor architecture and the toolset exploited for its design is provided. The proposed system is described in detail in Sec. 3.3, providing an overview of its components and the interfaces between them, with a particular focus on the FPGA-based audio coprocessor and its firmware. Then, in Sec. 3.4, the results obtained so far from a theoretical and an empirical point of view are summarized, focusing in particular on the performance of the FPGA.

3.2 Background

As reviewed in Section 2.5, state-of-the-art NMP solutions range from software frameworks to hardware-integrated approaches.

While several approaches to exploit FPGAs for real-time audio processing have already been investigated in [258, 182, 187], to the best of our knowledge, apart from some preliminary attempts appeared in 2013 [265, 16], this study proposes for the first time an implementation of an ultralow-latency audio streaming coprocessor based on FPGA, specifically designed for NMP applications.

The proposed system makes use of a custom Transport Triggered Architecture processor. The datapath of this architecture comprises several Functional Units (FUs), register files and interconnecting buses.

The instruction set philosophy for the TTA processor revolves around a single type of instruction, namely the “move" operation. In this system, more complex operations can be accomplished by orchestrating a sequence of “move" instructions between different FUs, which then actually process data. As a consequence, at compile time, each user program is mapped to a discrete set of operations, moving data from one FU to another via the interconnected buses. This compilation process ensures the efficient execution of higher-level tasks using the fundamental “move" operation as a building block. The length of each instruction is usually very long because it is divided into several move slots, each one dedicated to one of the available transport buses. For this reason, a TTA processor enables a natural parallelism in the execution of instructions. A picture of a simple TTA architecture is shown in Fig. 3.2. Due to their characteristics, TTA processors are particularly

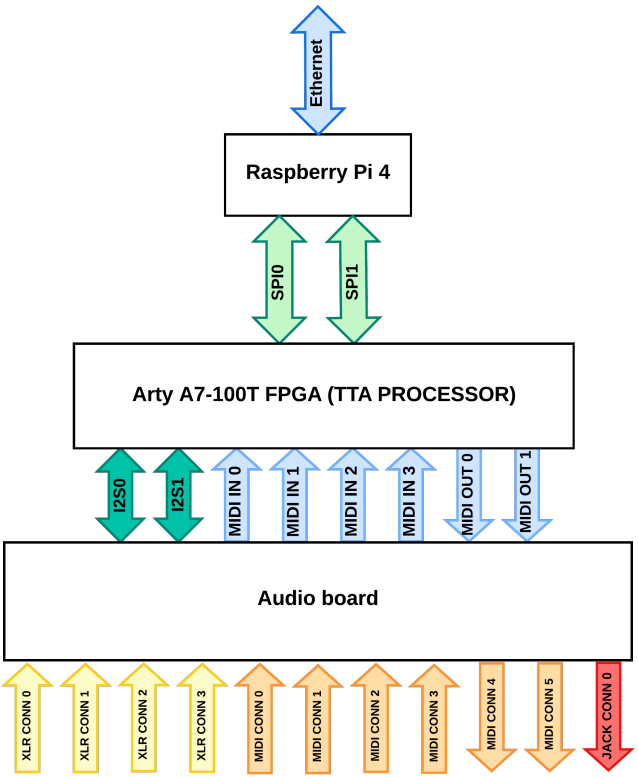


Figure 3.1: System block diagram. The FPGA coprocessor handles the real-time audio I/O, including ADC/DAC conversion via the I2S bus, the DSP pipeline on the TTA core, and the control/data exchange with the host. The Raspberry Pi runs the NMP session-layer software and communicates with the FPGA through the SPI link. Audio and MIDI paths are described in Sec. 3.3.

suitable to implement ASIPs, enabling designers to create custom units to accelerate specific tasks. Given the focus of this chapter on NMP, a TTA proves to be a valuable choice to reduce the latency of audio data handling. Specifically, tasks like audio playback, audio source mixing, and audio stream processing (e.g., effects, equalization) can be efficiently sped up using the TTA architecture. To achieve this acceleration, dedicated FUs can be designed for each of these functions. The TTA's inherent parallelism allows these specialized FUs to work simultaneously, enabling efficient and concurrent execution of multiple NMP tasks. This approach harnesses the power of parallel processing to meet the stringent latency requirements and demanding audio processing needs of NMP applications.

To facilitate the creation of a custom TTA processor, TTA-based Co-design Environment (TCE), now known as OpenAsip [114], was adopted in this project. Developed by the Customized Parallel Computing group at Tampere University,

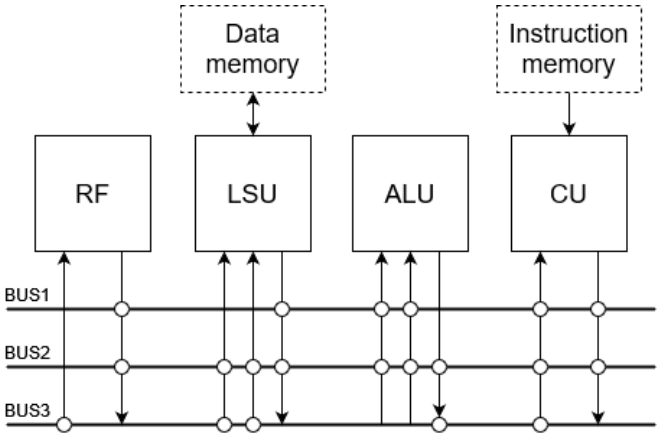


Figure 3.2: Simple TTA architecture

this environment offers a comprehensive set of tools to aid users in co-designing the Register Transfer Level (RTL) architecture of a TTA processor and the high-level software running within it. Thanks to TCE, users can seamlessly design and interconnect transport buses, register files, and basic FUs, forming the foundation of the TTA processor. Furthermore, custom FUs described by means of a Hardware Description Language (HDL), like those discussed in this chapter, can be integrated into the system. To enable high-level programming for the generated architecture, a custom LLVM/CLANG compiler is employed. Users can write their high-level code in C/C++ programming languages, which the compiler then translates into instructions compatible with the custom processor. This cohesive ecosystem ensures efficient co-design and implementation of custom TTA processors, empowering developers to effectively tailor their solutions for specific tasks.

3.3 System Description

The system proposed in this chapter is an initial hybrid solution combining a software and a custom digital hardware approach. In particular, as shown in Fig. 3.1, it consists of three main components, each of them implemented on a different hardware support: an audio board, an audio processor and a Raspberry Pi.

The three main components of the system exploit different communication protocols to interface with each other (see Fig. 3.1):

- the Inter-IC Sound (I2S) protocol, between the audio board and audio processor.
- the Musical Instrument Digital Interface (MIDI) protocol, between the audio board and audio processor.

- the Serial Peripheral Interface (SPI) protocol, between the audio processor and Raspberry Pi.

Internally, the audio processor and the audio board can handle up to 24-bit samples, even if at this moment the Raspberry Pi host machine works with 16-bit samples.

3.3.1 Audio Board

The custom audio board (Fig. 3.3) serves as a hardware circuit designed specifically to function as a sound card within the system. Its conception revolves around meeting stringent low-latency requirements and addressing issues commonly associated with common consumer-grade sound cards based on USB 2.0.

The Universal Serial Bus (USB) 2.0 is a widely adopted half-duplex serial interface known for its versatility and ease of use. It is commonly utilized for connecting various peripherals and devices to computers and other host systems. USB is designed to be flexible and support a wide range of functionalities, making it suitable for tasks such as data transfer, power supply and device control. However, USB is not inherently optimized for low-latency applications, such as real-time audio processing in the context of NMP. The USB protocol was originally designed to cater to various devices with different data transfer requirements, including keyboards, mice, printers, and storage devices. As a result, the USB protocol includes a complex control software stack that introduces some inherent overhead. This software stack, while necessary for managing the various devices and their functionalities, can introduce additional processing delays, which may impact real-time audio processing in NMP scenarios. The overhead introduced by the USB software stack can result in unpredictable variations in data transfer times and latency, making it challenging to achieve the precise and consistent timing required for real-time audio processing.

To ensure optimal performance, the developed audio board is engineered to efficiently acquire and play sound samples using multiple audio interfaces. It seamlessly exchanges data with the processor via the I2S protocol, employing minimal buffering and filtering. This reduction in buffering and filtering helps minimize added delays and optimize latency performance. The board is equipped with readily available commercial audio Analog-to-Digital Converters (ADCs), specifically two PCM1803 [177], and a Digital-to-Analog Converters (DAC), the PCM1771 [177], which can be directly employed as a headphone amplifier. MIDI [159] interfaces are derived by converting UART messages through commonly used optically-isolated circuits.

The custom audio board is designed to accept input DC voltages ranging from 3V to 18V. It efficiently derives the necessary power supplies for both the digital and analog sections through a combination of switching and low-noise linear regulators. Additionally, the board has the capability to generate a 48V Phantom power supply to facilitate the operation of condenser microphones directly on board.

Furthermore, the audio board assumes the critical role of directly interfacing with users for control purposes. It provides user controls, such as volume adjustment for each input audio channel, enabling smooth and intuitive interaction for performers and participants in an NMP session.

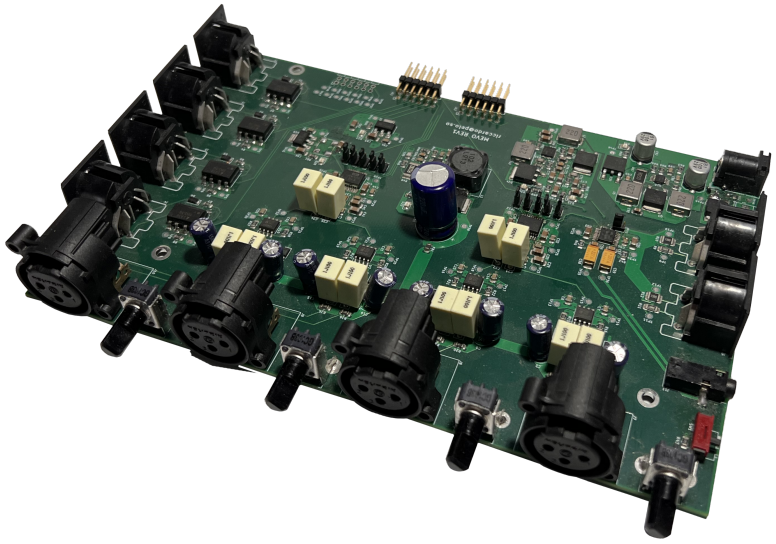


Figure 3.3: Custom audio board

3.3.2 Audio Processor

The second and most crucial component of the developed system is the audio processor, implemented as a custom TTA ASIP. Its goal is to create a communication channel between the audio board and the Raspberry Pi device, introducing the lowest possible amount of latency, as well as providing custom audio processing and manipulation, thus removing the computational load from the Raspberry Pi and accelerating these tasks. To host the processor, an FPGA-based board was chosen because of its flexibility in implementing digital circuits and its possibility to be completely reconfigurable. The choice for the development board fell on the development board Arty A7-100T produced by Digilent, featuring the Artix-7 XC7A100TCSG324-1 FPGA chip from Xilinx [13]. This FPGA model boasts an abundance of digital resources that can be harnessed for future advancements in the audio processor's capabilities.
Fig. 3.4 depicts a block diagram of the top module of the processor. Several components are shown:

- *TTA core*, the actual processor, hosting the different FUs.

- *Instruction memory*, containing the instructions obtained by compiling the user code.
- *Data memory*, used to store data and global variables.
- *I2S clock generator*, needed since the developed architecture acts as the master device for the I2S protocol. It generates the required clocks, namely the bit clock (bclk) and the word-select signal (lrclk).
- *Phase-locked loop (PLL)*, generating the master clock running at 11 MHz.

Both clocks are generated on-chip by the I2S clock generator, which is driven by the PLL's 11 MHz master clock; the external codec on the audio board operates as the I2S slave.

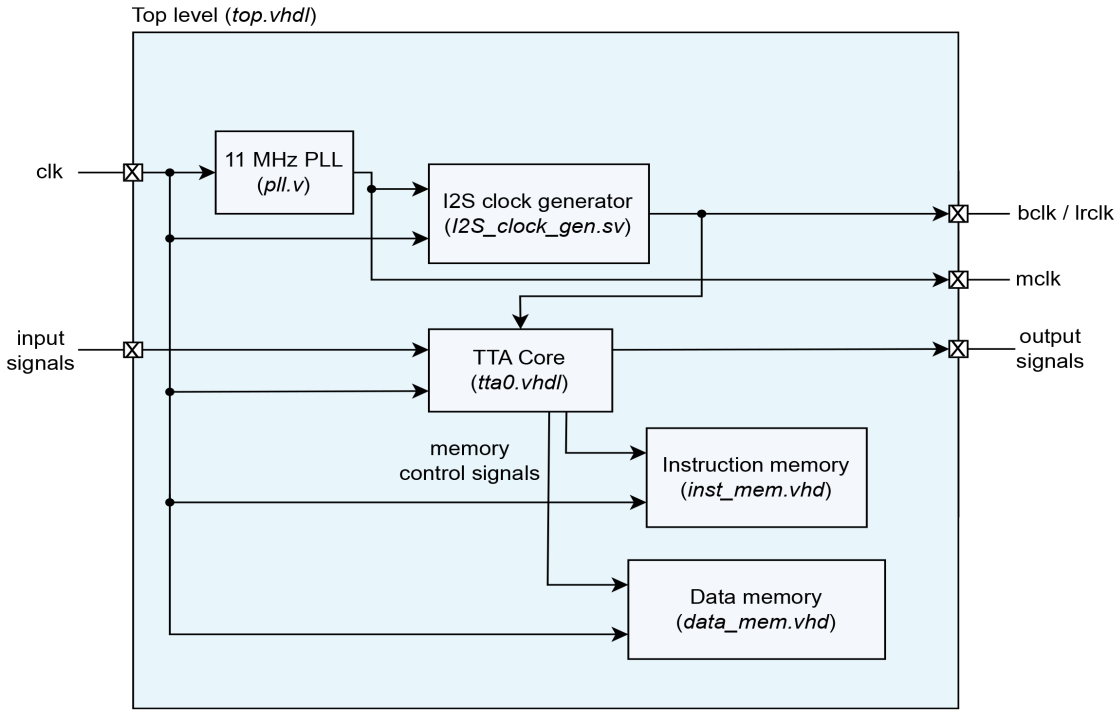


Figure 3.4: Audio processor top-level block diagram

It is possible to observe in Fig. 3.5 that the developed TTA processor is composed of several FUs connected together by means of four 32-bit transport buses. Some of these units are common to most architectures, such as Arithmetic Logic Unit (ALU), Global Control Unit (GCU), Load-Store Unit (LSU), two register files ("bool" and "REG_FILE_0" in the figure) and a timer ("TIMER_0" in the figure). Then, some custom units were developed in order to deal with the Light Emitting Diodes (LEDs) and switches of the FPGA board but, most importantly, to create

peripherals that interface the system components and provide some processing on the audio data:

- *I2S_LJ_master_TX_x*: I2S transmitter with Left-Justified format and 24-bit audio samples on 32-bit frames.
- *I2S_LJ_master_RX_x*: I2S receiver. Its behavior is symmetrical to that of the I2S transmitter.
- *UART_TX_x*: UART transmitter, working at 31250 Hz rate. It sends 8-bit MIDI messages.
- *UART_RX_x*: UART receiver, working at 31250 Hz rate. It is used to receive 8-bit MIDI messages.
- *SPI_slave_x*: an SPI transmitter/receiver, working with 8-bits frames.
- *MIXER_0*: performs the mixing operation on multiple audio sources. It features gain and left-right panning controls for each source.
- *REVERB_0*: applies a reverb effect on an audio source. It features a dry/wet control.

The last two units of the list, as well as other possible future ones, can be exploited to accelerate time-consuming audio processing tasks typically required by musicians. Moreover, the introduction of a reverb unit can have a positive impact on the way users interact, leading to more synchronized performances [66].
All the functional units are linked together with a fully connected structure, i.e., each one of them has access to all the available buses. In this way, the compiler is free to choose how to route the data and how to organize the move operations, depending on the required tasks.

For what concerns the uploaded firmware routine, the FPGA processor's task is to collect data coming from the audio board peripherals (I2S and MIDI) and route them to the SPI peripheral connected to Raspberry Pi, and vice versa.

SPI was selected as the interconnection protocol between the FPGA and the Raspberry Pi due to its widespread availability as a standard interface for prototyping purposes. This synchronous and dependable interface is well-suited for transmitting versatile data payloads, encompassing both audio data and command signals. For this task, I2S has not been employed as it is primarily designed for the transmission of stereo audio streams and incorporates an embedded clock signal. One advantage of the custom-designed TTA I2S interface is its internal buffering, which eliminates the need for a master clock. Sample re-synchronization occurs only when audio data is sent to the Digital-to-Analog Converter (DAC), ensuring precise timing and synchronization. DSP tasks are executed asynchronously at the

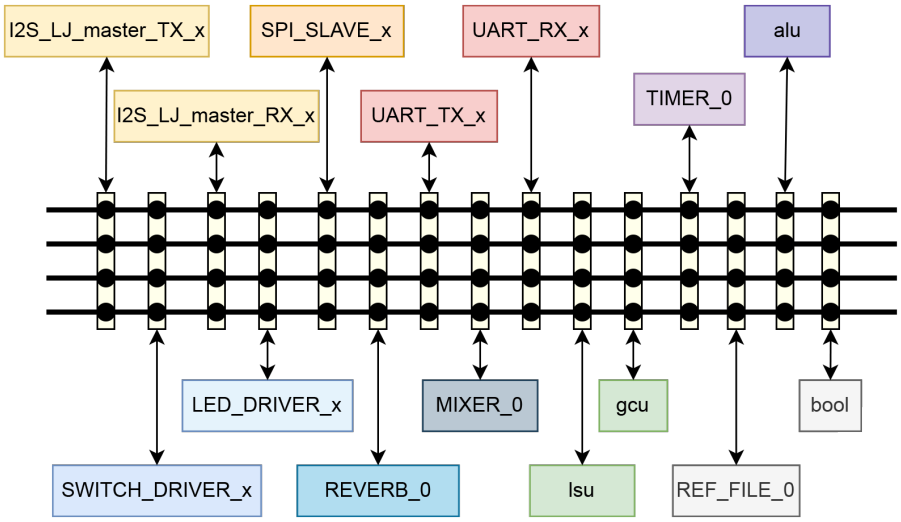


Figure 3.5: TTA core: FUs and interconnections

FPGA's system clock frequency. This design approach is geared towards minimizing latency, thereby ensuring that audio processing tasks are carried out with the lowest possible latency.

Support for two different operating modes has been considered, depending on whether the mixing is done at the Raspberry Pi or FPGA level. FPGA-level mixing accelerates the mixing task and speeds up the system, but will likely experience limitations for what concerns the maximum number of sources it can handle, due to the bandwidth of the SPI link (more details in Sec. 3.4.4).

During operation, the FPGA processor waits for a command coming from the Raspberry Pi, that is acting as an SPI master device. Three possible commands can be issued:

Command 1 is used to change the environment configuration variables, such as the number of audio samples bitwidth, the number of enabled input channels on the audio board, the number of sources coming from the Raspberry and others.

Command 2 starts the streaming of a packet of samples and MIDI messages from the audio board to the Raspberry Pi and vice versa. The mixing operation is performed by the Raspberry Pi.

Command 3 starts the streaming of a packet of audio samples and MIDI messages from the audio board to the Raspberry Pi and vice versa. In this case, the mixing operation is done at the FPGA level.

Furthermore, an analysis has been carried out regarding the effects of errors that can potentially occur in the SPI channel. While the impact of sporadic errors in

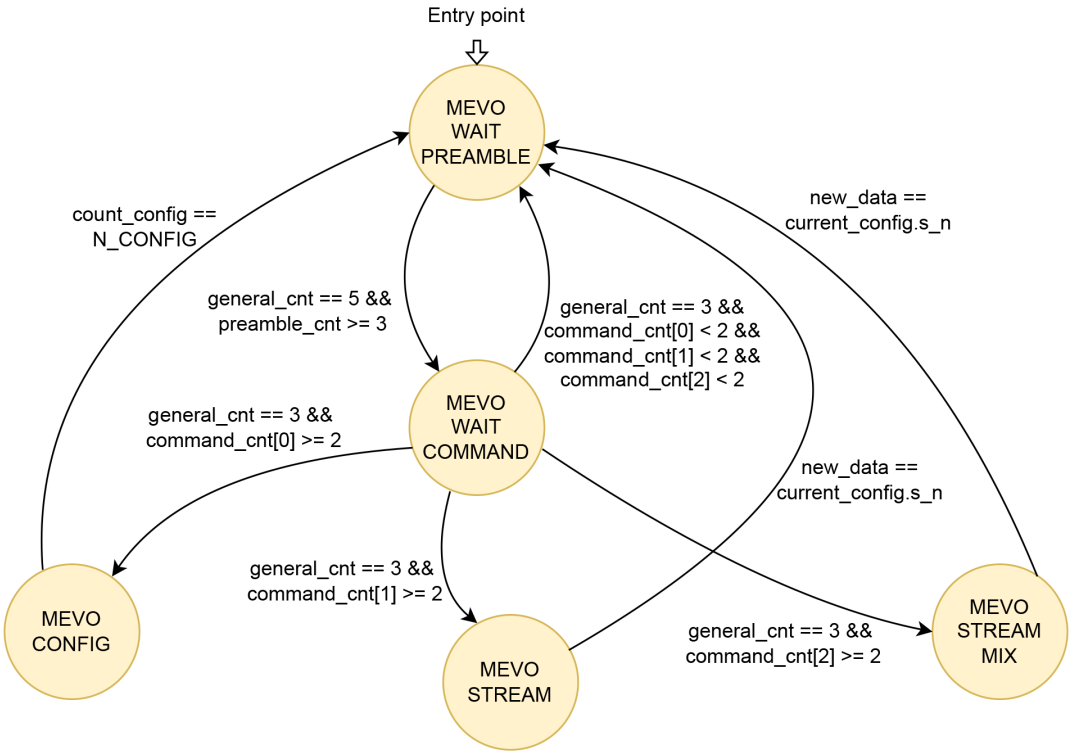


Figure 3.6: Flow diagram of the firmware routine

random audio samples, resulting in occasional audio glitches, can be deemed tolerable, keeping the error rate as low as possible is crucial when issuing commands, in particular command 1. In fact, if one or more of the configuration variables are not correctly received by the FPGA side, the whole communication could fail. To address this problem, a custom communication protocol has been developed. First of all, each command is preceded by a preamble sequence informing that the Raspberry Pi is asking to start the communication. Then, the transmission continues with an ACK/NACK mechanism. Moreover, redundancy in the preamble and commands is intentionally introduced, guaranteeing that the correct information is exchanged.

In the final implementation, SPI communication packets will incorporate Error-Correcting Codes (ECCs) such as Cyclic redundancy checks (CRCs) to enhance the reliability of the communication channel. The ECC scheme will be supported by firmware or hardware during the packet encoding and decoding processes, thereby providing an additional layer of assurance for data transmission integrity.

Fig. 3.6 represents a flow diagram of the firmware routine, devised as a Finite State Machine (FSM) consisting of the following states:

- **MEVO_WAIT_PREAMBLE**: during this state, the processor waits for

the preamble sequence coming from the SPI side.

- **MEVO_WAIT_COMMAND**: when the preamble sequence has been identified, the processor enters this state expecting a command to be issued.
- **MEVO_CONFIG**: in this state, command 1 is executed.
- **MEVO_STREAM**: in this state, command 2 is executed.
- **MEVO_STREAM_MIX**: in this state, command 3 is executed.

Currently, the processor does not perform many operations and is clearly underused. Anyway, its real potential is still to be exploited in enhanced versions of our architecure, that are forseen as future work. Transferring tasks from the Raspberry Pi to the FPGA offers several advantages, particularly in terms of speed. Consider, for instance, a reverb effect. In software, processed samples must continually shuttle back and forth between the processor and memory. Conversely, hardware implementation allows for efficient utilization of parallelism and pipelines, resulting in a notable reduction in computational overhead. It is true that developing hardware solutions entails higher upfront costs, but in a latency-critical context like NMP, the investment is worth the effort.

3.3.3 Raspberry Pi

In the proposed system, the Raspberry Pi 4 model B [184] plays a crucial role in managing the transmission and reception of audio packets over the network through its full gigabit Ethernet port. Given the stringent latency requirements of NMP, the choice of an appropriate device was crucial, and the Raspberry Pi 4 specifications proved to be a fitting option. It boasts a 1.5 GHz Quad-core 64-bit ARM Cortex-A72 processor and 8 GB of RAM, providing ample processing power and memory capacity for real-time audio processing tasks.

Moreover, the Raspberry Pi 4 offers 28 General Purpose Input/Output (GPIO) pins, providing various interface options and including support for multiple concurrent SPI communication lines. These capabilities make it a versatile platform for handling audio data efficiently. To ensure a robust and flexible system, the Linux operating system was deployed on the Raspberry Pi, benefiting from its stability and adaptability within the Raspberry Pi framework.

From a networking perspective, two crucial design choices were made. Firstly, UDP was chosen as transport protocol for communication. UDP's lack of handshake techniques for connection setup enables faster data transmission, making it well-suited for time-critical applications like NMP. However, it is important to acknowledge that UDP communications are not reliable and may require PLC methods to address missing data packets. Secondly, a peer-to-peer architecture was

implemented among the users of the NMP environment. This choice further reduces perceived latency compared to a client-server solution, facilitating direct and low-latency communication between participants.

3.4 Results

The following sub-sections present the results obtained so far for the proposed system. It is worth mentioning that, since the architecture design is still ongoing, most of the experimental evaluations cannot yet be assessed, especially the ones related to the Raspberry Pi component. For the same reason, a thorough evaluation of the impact of network delay and of the overall system performance will be possible only once the design phase is completed.

3.4.1 FPGA Utilization

The proposed TTA processor was implemented on FPGA leveraging the Vivado IDE by Xilinx to evaluate the efficiency of the design and the resource consumption. The reported utilization results, shown in Fig. 3.7, indicate how the ASIP makes use of the FPGA resources. Specifically, approximately 21% (13479 out of 63400) of the available logic elements, i.e. Lookup Tables (LUTs), were employed, while only 8% (20 out of 240) of the DSP cells were used. These outcomes emphasize that many signal processing operations on sound samples can still be handled by means of specific units, performing tasks which are recognized to be particularly time consuming in a general-purpose environment.

Regarding memory blocks, it can be noticed that 10% (13 out of 135) of the available Block RAMs (BRAMs) and 38% (7193 out of 19000) of the Lookup Table RAMs (LUTRAMs) were used, mainly to store instruction and data memories of the TTA processor and to address the need for long delay lines in the reverb unit. This results indicates that the design made efficient use of the FPGA, leaving room for potential inclusion of additional functionalities which require to store data. Thanks to this available programmable logic, several features can be taken into account for future developments as custom FUs, such as new audio effects and hardware PLC techniques to cope with missing data packets.

3.4.2 FPGA Power Consumption

Power consumption is a critical factor, especially in portable or energy-constrained systems. To assess the power efficiency of the proposed FPGA-based processor, the Vivado static power consumption estimator has been used. The power consumption of the developed TTA architecture was found to be significantly lower compared

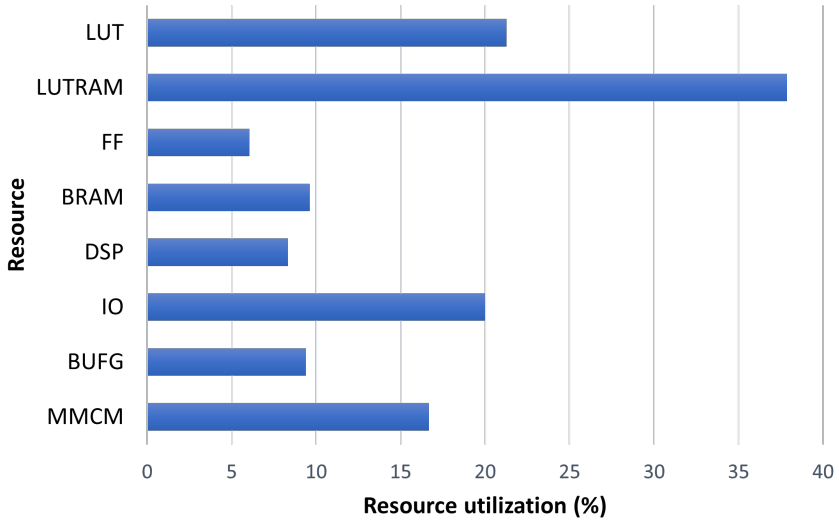


Figure 3.7: FPGA resource utilization

to traditional software-based approaches, with a total estimated power of approximately 350 mW. In particular, in Fig. 3.8, the different power contributions are reported. The main ones are those related to the PLL circuit (indicated as "MMCM" in the figure) and to the static power consumption of the ASIP (indicated as "PL static" in the figure).

The power consumption results obtained so far are acceptable estimations provided by Vivado. However, to understand the real power dissipation of the processor, future empirical evaluations will be considered by exploiting FPGA power monitoring capabilities.

3.4.3 Local System Latency

The primary performance metric for NMP applications is the total system latency, which directly impacts the real-time musical interaction between remote performers. To evaluate the effectiveness of the proposed solution, latency measurements were carried out with a local audio loopback test on the system composed of the audio board and the FPGA-based processor (the Raspberry Pi was not considered in this initial analysis). Throughout these tests the coprocessor operates at the NMP target rate of 44.1 kHz, 16-bit stereo. Tests were conducted without involving the audio mixing unit, since this process was assumed to be performed by the Raspberry Pi. Moreover, the reverb effect was excluded from the analysis, because it could alter the time delay evaluation. The experimental setup consisted of a software tool capable of generating a chirp sound and collecting it after the audio loopback to measure the latency introduced by the system. Since

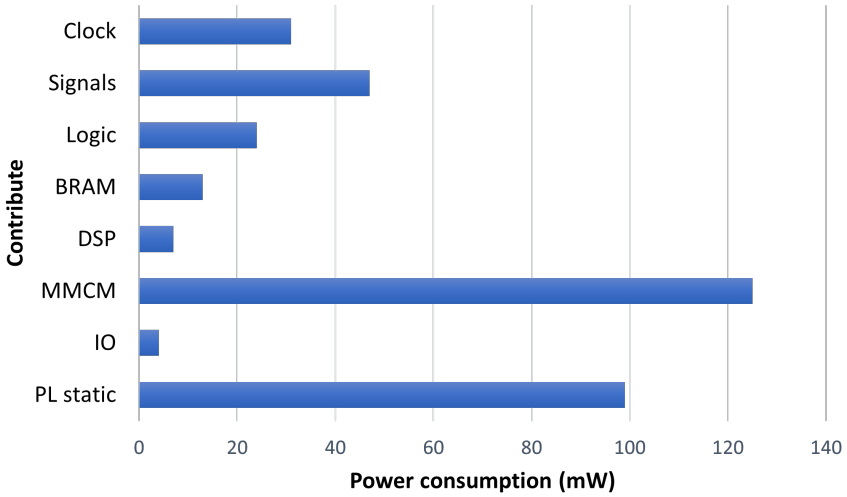


Figure 3.8: FPGA power consumption contributions

the sampling rate of the calibrated audio analyser used as measurement reference is 192 kHz (this figure applies to the analyser only, not to the FPGA audio path), empirical results are affected by an uncertainty of 0.005 ms.

From a theoretical point of view, the overall latency considered can be summarized as:

$$T_{lat} = T_{ADC} + T_{I2S} + T_{proc} + T_{I2S} + T_{DAC}$$

where T_{ADC} and T_{DAC} are the time delays associated with audio conversions on the audio board, T_{I2S} is the data transfer delay (considered two times since data are transferred back and forth) and T_{proc} refers to the latency introduced by the TTA processor. By performing this computation, it was possible to estimate a theoretical value for the loopback test of 0.89 ms, which was then confirmed by empirical results, showing a 0.859 ms delay.

3.4.4 SPI Bandwidth and Protocol Overhead

The required bandwidth to link the FPGA to the Raspberry Pi through the SPI interface is primarily determined by the number of audio channels being utilized. With a fixed audio sample rate of $f_s = 44.1\,\text{kHz}$ and $w_s = 16$ bits per sample, each channel demands the following bandwidth

$$BW_{chann} = f_s \cdot w_s = 705.6\,\text{kbps}$$

However, it is essential to consider the overhead introduced by the communication protocol introduced in Sec. 3.3.2, as it affects the effective data rate. For instance, when dealing with command 2 (Raspberry Pi mixing mode), the protocol introduces redundancy that needs to be accounted for in the bandwidth calculation. To

quantify the overhead, a formula can be used, taking into account the samples bit width (w_s), the number of channels sent by the FPGA through the communication link (n_{CHF}), and the number of samples inside an audio data packet (s_n). For example, with $w_s = 16$ bits, $n_{CHF} = 4$, and $s_n = 112$, the total overhead introduced in a packet of samples can be computed as

$$OH = \frac{8}{(\frac{w_s}{8} \cdot n_{CHF} \cdot s_n) + 8} \approx 0.88\%$$

where the number 8 at the numerator represents the number of bytes required by the preamble+command sequence, while the denominator term represents the total number of bytes transferred during one occurrence of command 2.
Another interesting figure is the number of audio sources that can be streamed through the SPI link, especially when the mixing operation is performed at the FPGA level. To be sure to work in “safe" conditions, BW_{chann} can be increased to a higher value, for instance $BW'_{chann} = 750\,\text{kbps}$. According to the datasheet of the chip BCM2711, which implements the SPI communication on the Raspberry Pi device [19], the maximum speed the link can reach is $BW_{RPI} = 125\,\text{MHz}$. Thus, supposing that the electrical connection between the Raspberry Pi and the FPGA board permits to reach this speed, the theoretical maximum amount of sources that can be streamed through the SPI link is

$$N_{SRC_{MAX}} = \frac{BW_{RPI}}{BW'_{chann}} = \frac{125\,\text{MHz}}{750\,\text{kbps}} \approx 166$$

Moreover, Raspberry Pi features two SPI peripherals, that can potentially be used in parallel to increase the throughput and/or to spread the bandwidth requirement on both of them.

Chapter 4

Pristine Quality Networked Music Performance System for the Web

The preceding chapter established technical foundations at the hardware level. This chapter addresses the central challenge identified in the introduction: reconciling professional audio quality with accessibility for non-expert users. It does so by foreseeing a platform made of a native client for scenarios requiring tight latency control and a web interface that allows users to join sessions directly from a browser. Both clients share the same interface and transmit identical uncompressed audio, differing only in the underlying transport mechanism. Furthermore, it evaluates the impact and perception of quality degradation through subjective tests.

4.1 Introduction

As discussed in Chapter 1, the preference of musicians for familiar videoconferencing tools over specialized NMP applications revealed fundamental accessibility barriers. This chapter addresses that challenge by developing a hybrid web-based and native application that demonstrates how professional audio quality can be achieved without compromising ease of use.

The proposed solution provides a unified environment, both web and native, for the transmission of pristine (uncompressed) stereo audio, attempting to find a compromise between technical sophistication and accessibility for a broad range of users.

After a brief overview of the state of the art in Section 4.2, the description of the architecture of the proposed solution follows in Section 4.3, while the audio quality results obtained after a subjective assessment phase are reported in Section 4.4.

4.2 Background

Among widely adopted videoconferencing platforms (Zoom [268], Skype [157], Google Meet [86]), as will be demonstrated in the subsequent sections, the audio quality attained through Zoom and Skype is generally deemed superior to that achievable through Google Meet. Nevertheless, the latter platform has also garnered a substantial user base, even among audiophiles. One plausible explanation for this phenomenon is the clear distinction that exists between these tools. Whereas the two former ones are native applications that necessitate downloading, installation, and configuration on local computers, the latter is a web-based application that operates on web browsers, thereby obviating any need for installation and configuration procedures.

The trend toward web-based software solutions for remote musical interactions has also emerged in the aftermath of the pandemic. A good number of the new solutions that appeared on the market in the last two years adopted the form of web-based services that can be accessed directly through the web [116, 63, 225, 51], thereby enabling even those with limited computer proficiency to utilize them without encountering any skill-based impediments.

However, the actual improvement of these new services mainly focuses on the implementation of new features, such as additional audio controls and mixing options, to meet the requirements of musicians. On the contrary, in terms of audio quality, their implemented improvements are marginal and do not go beyond the ones achievable with a fine-tuning of the options provided by Skype or other web-based applications like Jitsi (Google Meet, unfortunately, does not allow for such customizations). Moreover, the communication latency remains within the range of interactive speech communications, which typically falls between 100-300 ms [231]. This latency is far beyond the requirements necessary for networked music performance established in Section 1.1.

The reason for such a limitation is that, as far as we know, all these web services were implemented using the Web Audio [5] and WebRTC [27] standards, that do not permit transmitting audio data with low delay and without perceptual audio compression. Even with high bit rates, such compression hinders the audio quality and adds algorithmic delay during the encoding process [201].

In addition, audio processing algorithms designed for voice communications can introduce sound artifacts due to automatic gain adjustment mechanisms that change sound levels. Furthermore, noise-canceling algorithms tend to dampen sustained stationary sounds, while algorithms for adapting playback speed to channel bandwidth fluctuations can alter the pitch and timbre of sounds.

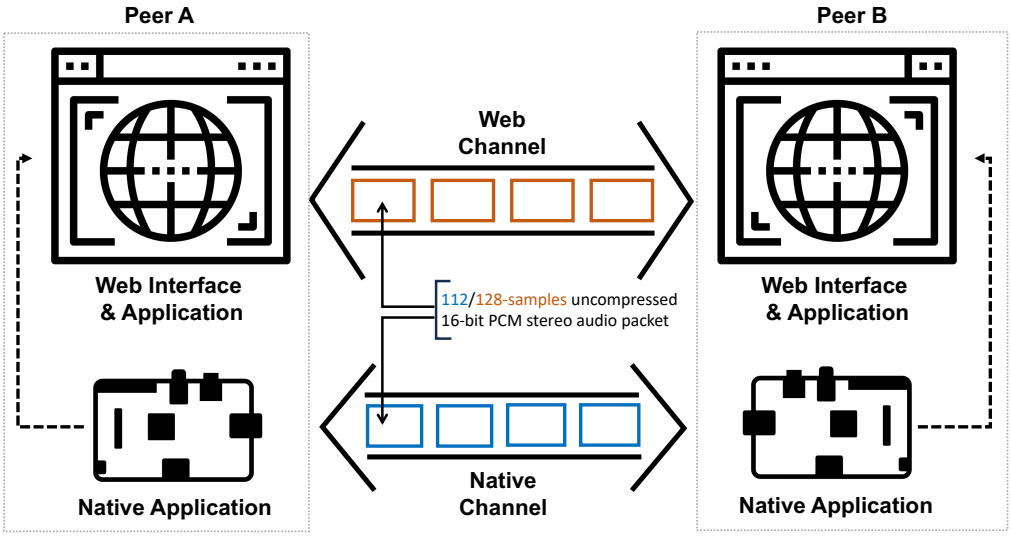


Figure 4.1: The proposed application web interface is shared between the native and the web implementation. Users can musically interact with other peers leveraging either the web or the native channel, using the same audio representation.

4.3 Software Architecture

The service architecture leveraged in this chapter is an enhanced version of the one presented in [199]. The service is structured around a client component which has two implementations, a web-based one and a native one. Independently of the implementation, in both versions of the client, the user interacts via the same Graphical User Interface (GUI) accessible through the web browser.

As shown in Fig. 4.1 the user can choose to activate the interactive music communication either through the web browser itself or through the native implementation that runs on a dedicated device.

In short, the rationale behind opting for a hardware component over a personal computer is that it eliminates the need for software installation and the possibility of conflicts with other software. Additionally, it allows for the implementation of the application to suit the specific hardware, thereby optimizing computational performance and audio input/output chain. In fact, direct access and configuration of the audio card (something that is not available in the web context) permit to reduce the audio latency of the NMP system to its minimum. The native implementation is based on a Raspberry Pi 4B, it runs on top of a custom Linux Os, and it reaches a latency in the order of 10-15 ms. Additional details on the performance of the native implementation can be found in [199].

On the other side, the server component of the system is limited to managing the signaling functionality, which enables the discovery of the various clients and the initialization of the communication.

Despite the differences in the communication protocols, both the communication channels, i.e., the one initiated by the web application and the one initiated by the native application, carry the same audio format; thus, they are equivalent in terms of audio quality. The audio data is transmitted as plain uncompressed 16-bit PCM [143], which ensures unaltered audio quality at the receiver. While sending uncompressed audio is straightforward in a native application, the relevant contribution of this chapter is the achievement of the same result also on a peer-to-peer communication between web browsers.

In fact, the proposed system avoids using the conventional WebRTC components for the management of audio communications, i.e., the MediaStream and RTCPeerConnection. Instead, a new implementation (initially presented in [197]) has been tested, which, after the audio acquisition utilizing an AudioWorklet, extracts the audio samples from the MediaStream and redirects them to an RTCDataChannel for transmission over the network. This implementation allows for controlling the audio processing and communication channel at a very low level, i.e., bypassing the speech-oriented algorithms built into the RTCPeerConnection that alter the audio signal. In addition, the SharedArrayBuffer object used for the data transfer between the AudioWorklet and the main thread showed to be extremely efficient, i.e., with negligible latency and processing overhead. The system shows latencies as low as 40 ms on Firefox on macOS, and in general achieves a latency reduction of 10 to 50% w.r.t. the conventional WebRTC setup. Further details on the architecture of the system and its performance can be found in [201].

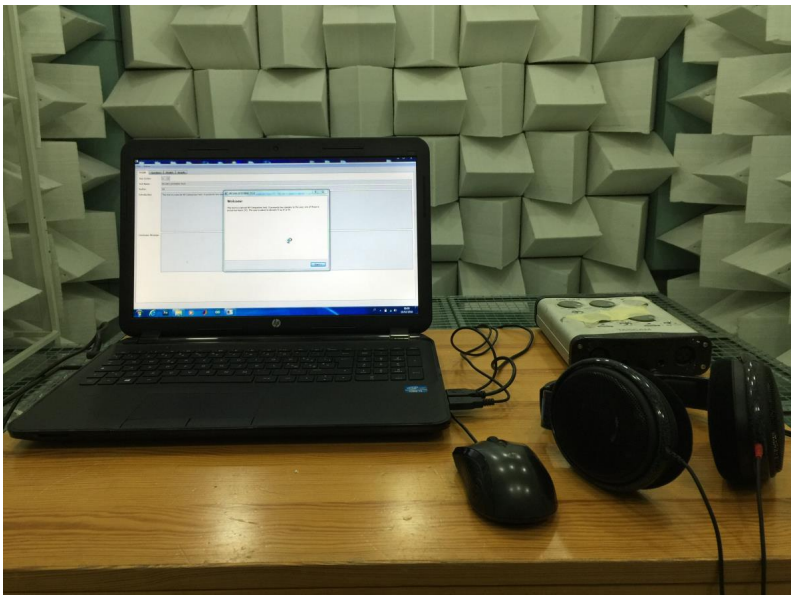
The combination of the AudioWorklet, SharedArrayBuffer, and RTCDataChannel proved to be able to efficiently handle the strict communication requirements of NMP, both in terms of bitrate and frames per second, without any noticeable performance reduction with respect to the conventional RTCPeerConnection implementation.

4.4 Audio Quality Evaluation

From the point of view of the system's audio quality, the performance of this software implementation is compared to that of state-of-the-art NMP software.

To evaluate the best performance achievable by the different applications while excluding the impact of network conditions (that are out of our control of the software engineer), the experiments have been carried out on a wired local network operating at 1 Gb/s, where the impact of transmission bandwidth, latency, and network losses are negligible. In this context, where network delay variability can be assumed more consistent than on the public Internet, a small de-jitter buffer has been used, i.e., a buffer introduced at the receiver side to prevent losses due to late packets. The size of the de-jitter buffer is roughly comparable among the different applications being considered (in the order of 5-9 ms for the native solutions and 20 ms for the web solutions).

(a) Anechoic room setup



(b) Listening test hardware

Figure 4.2: Listening test performed in the anechoic chamber of Politecnico di Torino [220].

4.4.1 Subjective Audio Assessment Without Packet Losses

A subjective audio test has been performed to compare the proposed solution with state-of-the art solutions used for remote music interaction during the COVID-19 pandemic.

The test was carried out in an anechoic chamber with a background noise level in

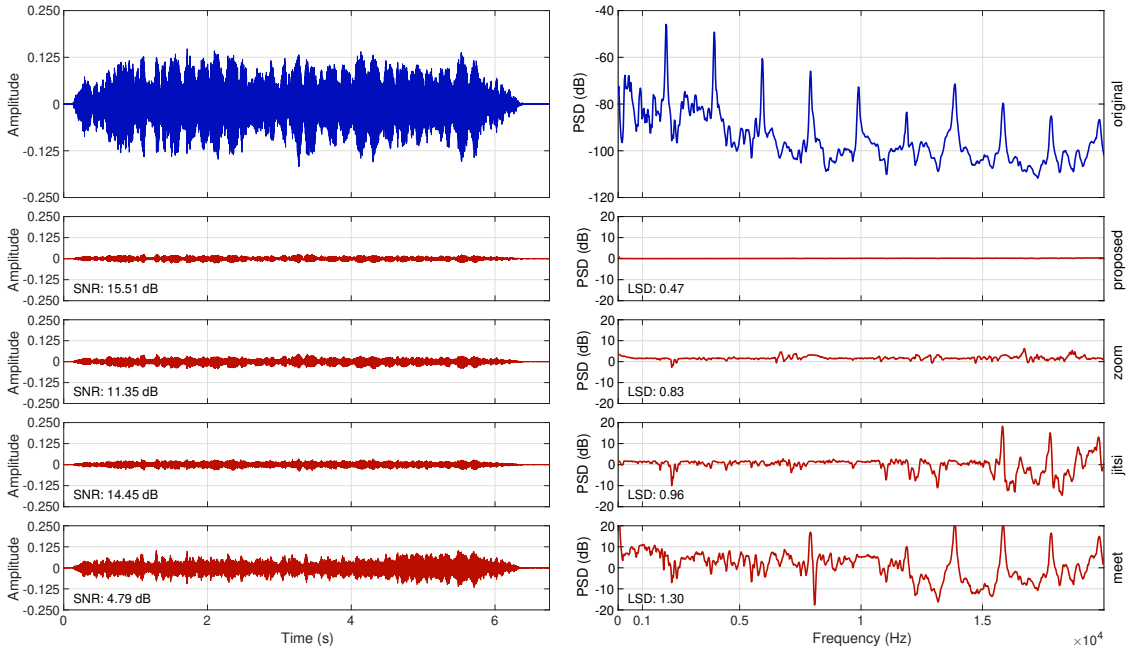


Figure 4.3: The first row shows the time and frequency domain representation of an audio segment (bowed violin). The following rows show the distortions introduced by the proposed audio transmission framework and by three state-of-the-art video-conferencing solutions, mainly due to perceptual codec compression and automatic digital signal processing applied to the original signal. The plots also indicate the Signal-to-Noise Ratio (SNR) and the Logarithmic Spectral Distortion (LSD) computed between the original and the processed signal for the proposed framework and then for Zoom (with the original audio feature turned on), Skype (with audio processing disabled) and Google Meet.

accordance with the recommendations of ITU-R BS.1116-1 by 26 subjects who self-declared to have non-impaired hearing conditions and who evaluated four stimuli corresponding to different types of audio selected from the material already used in a previous study [106]. For each stimulus, the uncompressed audio transmitted by the proposed solution was compared to the compressed audio transmitted by the benchmark applications through an ABX-type test [163] presented to each participant through headphones (Sennheiser 600HD).

In this procedure, three stimuli are presented to the listener: stimulus A and stimulus B, which have a known difference, and stimulus X. The listener's task is to identify whether X equals A or B. If there is no audible difference between the two signals, the listener's responses should be binomially distributed, such that the probability of responding $X = A$ equals the probability of responding $X = B$, i.e., 50%. This score is interpreted as an indication of the absence of perceptual differences between A and B.

The minimum number of correct answers needed to indicate a perceptual difference can be given by the inverse cumulative probability of a binomial distribution based on the number of trials, confidence level and probability of correct answer. For the conditions of the test performed, the minimum number of correct answers necessary to indicate a statistically significant perceptual difference is 17.

The four stimuli presented to the subjects represented different audio material: *i*) a bowed violin playing pitch B6 for 6.7 seconds (Violin), *ii*) a cello playing pitch C6 for 6.9 seconds (Cello), *iii*) a female sex soprano singing a C5 for 2.8 seconds (Female NV), and *iv*) a female sex soprano singing a G4 with operatic vibrato production (Female V). Each stimulus was then recorded after the processing and transmission with each one of the four considered applications: *i*) the proposed solution, *ii*) Zoom with "original sound" option turned on, *iii*) Jitsi with gain control, noise reduction, and echo cancellation disabled, and *iv*) Google Meet with its default configuration (since it is impossible to customize its audio processing features).

In the first row of Fig. 4.3, the time and frequency representation of an audio track is presented, consisting of a bowed violin playing the pitch B6 quietly (-12 dB). The subsequent rows display the audio impairments produced by various applications, including, from top to bottom, the proposed solution, Zoom, Jitsi, and Google Meet. For each application, the Signal-to-Noise Ratio (SNR) [119] and the Log-Spectral Distance (LSD) [183] were computed and shown in the figure respectively in the time and frequency domain plot. It is acknowledged that the time domain difference and SNR computation are not significant for audio distortion. In fact, due to the involvement of an analog section in the communication path, a slight misalignment of even a portion of a sample period leads to low SNR, not justified by the listening and by the spectral difference evaluation. The spectral difference representation yields instead significant results, as further demonstrated by subjective listening tests, indicating that the proposed implementation can achieve almost transparent quality even if implemented as a web application. A quality that is clearly superior to the one offered by similar software such as Jitsi and Google Meet. Although there is some distortion at low frequencies in the Zoom plot, the subjective tests evaluated its audio as nearly transparent. It is highly likely that the spectral difference is due to components of the sound that are masked by the original signal and, therefore, imperceptible.

Figure 4.4 shows the overall results of the subjective experiment. In the case of audio transmitted by the proposed system, only 12-14 subjects out of 26 correctly identified the non-original audio in the various listening tests, while in the case of audio transmitted by a third-party web application, 13-25 out of 26 people correctly identified the non-original audio. As mentioned in the previous paragraph, the results of correct answers above 17 have been considered as statistically significant in order to confirm the perceived differences. The results indicate that the audio of the proposed system does not show any significant difference for all the stimuli,

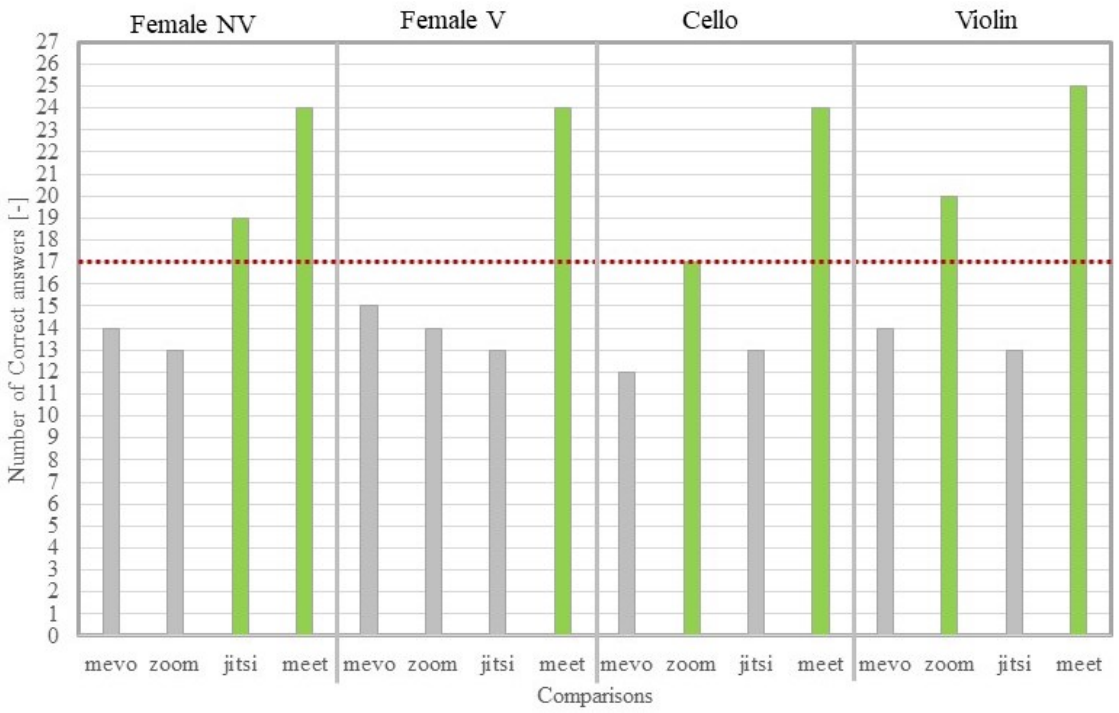


Figure 4.4: Results of the subjective experiments with no losses. Four stimuli – female sex soprano (Female NV), female sex soprano with vibrato (Female V), cello (Cello), and bowed violin (Violin) – processed by four different systems – the proposed one (P), Zoom (Z), Jitsi (J), and Google Meet (M) – have been presented to 26 subjects. The statistical significance threshold of correct answers is 17.

conversely, Google Meet presents significant differences independently on the type of stimuli. On the other hand, Zoom shows significant differences for the instrumental stimuli, i.e. Cello and Violin, while for Jitsi, a significant difference emerges for the Female singing a C5 (Female NV) only.

4.4.2 Subjective Audio Assessment with Losses

For the proposed solution, an additional simulation and a corresponding subjective test have been performed to evaluate the effect of audio packet losses on the audio quality perceived by the receiving user. Unfortunately, the same experiment could not be performed on the other software because they are not open source and thus their source code cannot be accessed to simulate the effect of packet losses and their concealment.

Whenever an audio packet is unavailable for playback at the receiver in due time, either because it was dropped by an intermediate network node or because it was excessively delayed, a gap may occur in the playout. Since the retransmission of the

data from the sender is not applicable due to the strict latency constraints of NMP, Packet Loss Concealment (PLC) techniques are usually employed to reconstruct the original audio frame in order to reduce the auditory perception of an artifact or glitch [179].

In the following, we present the performance of the concealment technique described in [200], when applied to the audio playout streams generated by the NMP application presented in this chapter. More in detail, the missing audio frames are reconstructed by means of an autoregressive model that is capable of predicting and synthesizing the lost audio section based on its historical trend, without introducing any additional delay [91]. This technique is compared with two standard concealment algorithms, typically implemented in NMP software, that fill the audio gap with silence (silence substitution) or with the repetition of the last received audio frame (pattern replication).

Two audio files with a duration of 5 seconds were selected from six audio categories (violin, drums, piano, guitar, songs and generic audio) for a total of twelve audio samples. Each sample was then divided into audio frames of 128 samples, i.e., about 5.8 ms at 22'050 Hz. Packet losses were simulated by selecting twenty random frames from each sample that have been reconstructed with the three PLC techniques. During the subjective quality test, each subject was presented with twelve questions, one for each audio sample. First, each subject was asked to listen to the original audio version and then to the three reconstructed signals, in a random order, without any indication of the corresponding concealment technique. Finally, each subject was asked to select, out of the three reconstructed signals, the one which sounded more similar to the original version. This test involved a total of 25 subjects. A minimum number of 17 answers are required to indicate a statistically significant preference for the AR models PLC technique w.r.t. the traditional PLC techniques.

The results of the test are shown in Tab. 4.1. In most cases, the most effective PLC technique is the one based on AR models. In particular, AR models show particularly good performance with signals that contain sustained sounds, as in the case of piano, violin, and guitar, while their benefits are reduced in the case of sounds containing abrupt transients, such as drums. This is due to the fact that, since AR models predict future samples based on the past history of the audio stream, sustain-oriented signals have low variability, thus it is easier to predict their future evolution, whereas transient-oriented signals have high variability, thus their prediction is less accurate. When songs are considered, no clear preference is revealed based on the test results, since in that case, due to the presence of many audio sources, audio gaps are less noticeable, and, as a consequence, also the difference between the concealment techniques is not particularly evident to human listeners.

Table 4.1: Subjective test of three PLC techniques: silence substitution, pattern replication, and autoregressive (AR) models. Each column contains the number of subjects that perceived the relative PLC technique variant closer to the original version. A total of 25 subjects were involved.

Audio sample	Silence substitution	Pattern replication	AR models
Violin A	0	2	23
Violin B	0	1	24
Drums A	2	5	18
Drums B	5	3	17
Piano A	3	2	20
Piano B	0	4	21
Guitar A	1	3	21
Guitar B	2	1	22
Song A	5	7	13
Song B	7	6	12
Generic A (trumpet)	0	1	24
Generic B (voice)	1	4	20

Chapter 5

Demonstration of a Networked Music Performance Experience with MEVO

Laboratory measurements establish technical feasibility, but real-world validation under performance conditions provides essential evidence of practical validity. This chapter presents the NMP software prototype named MEVO and documents a distributed concert connecting musicians in Turin (Italy) and Wrocław (Poland), demonstrating that the MEVO system can support actual artistic performance across international distances. The concert served both as a public demonstration and as a data collection opportunity, bridging the gap between controlled experiments and live musical practice.

5.1 Introduction

An NMP framework is a system providing the technological facilities to support remote musical performances. The objective of a NMP system is to be as transparent as possible to its users, enabling them to play together as if they were in the same room.

This chapter presents an example of experience with MEVO (Music rEVOlution), the NMP system developed at Politecnico di Torino, by describing its use in a remote concert which took place in June 2023 and involved performers in two locations: Politecnico di Torino and Wrocław University of Science and Technology.

The remainder of the chapter is structured as follows: section 5.2 provides a brief background on NMP, section 5.3 presents the experience of the distributed concert

and section 5.4 shortly discusses the results based on telemetry data collected during the experiment.

5.2 Background

In NMP, maintaining low mouth-to-ear latency is essential for realistic musical interaction, though depending on the musical piece and on the musicians' ability to apply delay-coping strategies, higher values may still be tolerated. Usually, latencies above 60 ms lead to significant tempo deceleration, which degrades the musical performance [26].

The second aspect to take into account is that latency varies along time due to several reasons, e.g., evolving network traffic conditions, misalignment of audio card clocks (audio drift), etc. Thus, a NMP system should deal with these issues in a way as transparent as possible to musicians, aiming to keep the perceived latency as steady as possible. Our NMP system addresses jitter issues by relying on a dynamic estimation of the playback buffer, i.e. the buffer that stores incoming audio frames prior to their reproduction, that dynamically adapts to the network conditions.

While the playback buffer addresses timing variations, it cannot recover packets that are discarded or never arrive. Packet loss concealment (PLC) techniques address this complementary challenge by synthesizing replacement audio when data is missing. In our current implementation of the system, no packet loss concealment technique is implemented, so in case of packet loss missing samples are replaced with silence, but PLC techniques such as those presented in [200] are in the process of being integrated.

Many NMP systems are available in the literature, but the final implementation of MEVO will differ from those already available at either commercial or experimental stage, as it will specifically focus on supporting remote music education and integrate dedicated hardware and software to ensure accessibility also by visually, auditory and mobility-impaired users.

5.3 Experiment

MEVO features a Raspberry Pi 4B connected to a USB audio card. The software part consists in the headless version of the Raspberry Pi Operating System (Raspberry Pi OS Lite), with Linux 6.1 PREEMPT_RT, a control system written in JavaScript and running on Node.js 18.16 and the main program written in C++17. Furthermore, the operating system runs at the stock clock of $1.5GHz$, using 3 over 4 of its cores for general purpose activities, whereas the fourth one is reserved for the management of input/output to/from the audio card. The main program is responsible for capturing the Pulse Code Modulation (PCM) audio by

means of ALSA's APIs, possibly converting the sample format (without adding any extra encoding), encapsulating it in UDP packets and transmitting it to all the other players in a peer-to-peer fashion. Concurrently, it receives the packets from the other players' devices, buffers the audio samples they contain and plays them back through the sound card.

We demonstrated MEVO in a distributed concert between the cities of Wrocław (Poland) and Turin (Italy). The setup was the same in both locations. The concert featured three musicians located in Turin (a cellist, a flutist, and a pianist playing a midi keyboard), and three in Wrocław (a singer, a clarinetist and a percussionist). The live performance lasted approximately one hour, but the system was powered on and working for $\approx 2h\,45'$. The musical pieces were composed ad hoc for the event, and premiered during the concert. The musicians opted for playing five pieces with the help of a metronome click generated at the Turin side. During this part of the concert, the musicians performing in Turin were not listening to their counterpart in Wrocław, whereas the musicians in Wrocław were receiving the metronome's click together with the live audio of the musicians in Turin. The metronome click was only audible by the musicians and not by the in-presence audience at the Wrocław side.

This approach was motivated by the round-trip delay measured during rehearsal, which the musicians judged too high to maintain ensemble synchrony without an external reference. In this configuration, the Turin performers did not need to listen to their remote counterparts and could anchor their playing to the local click; the Wrocław side, in turn, received the same click mixed with the Turin audio, giving both groups a shared timing reference.

By contrast, the final piece featured the cellist and the percussionist performing without any additional synchronization mechanisms.

Figure 5.1: Photography of musicians performing in Turin

5.4 Results

At both sides, we configured the NMP software to collect telemetry data, including the automatic sizing of the playback buffer (proportional to the standard

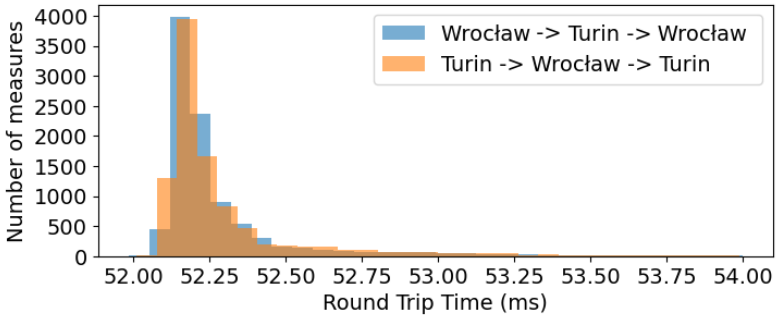


Figure 5.2: Distribution of Round trip times. Figure reports value between 0 and $54ms$ for ease of reading.

deviation of inter-packet arrival times), the round trip time (RTT) and the amount of lost audio frames, either due to lost UDP packets or to late ones. The data capture process took place regularly once every second by a software thread designed to minimize the impact on the performance of the rest of the software. The minimum measured RTT value was $51.985\,ms$ at one side and $52.011\,ms$ at the other one. Overall, the network exhibited a stable behavior, 99.986% of the measured RTTs was $< 59ms$ at both sides. Fig. 5.2 shows the RTT distribution.

Fig. 5.3 illustrates the amount of lost frames during the concert. The sampling rate was 44100 Hz, thus that the total count of lost frames at Turin side corresponds to approximately $18s$ of lost audio over $\approx 2h\,45'$. More precisely, the ratios of lost frames are 0.131% at Wrocław side and 0.177% at Turin side. Furthermore, based

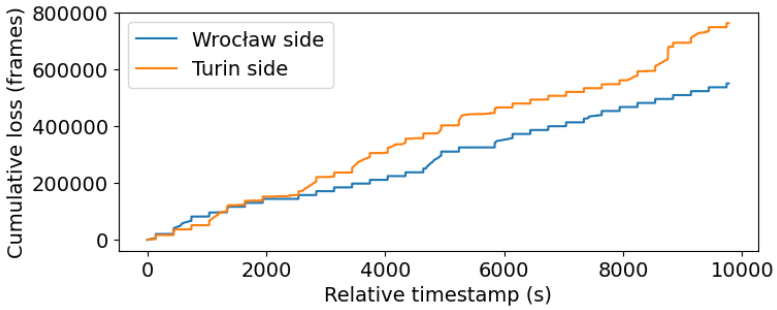


Figure 5.3: Cumulative losses at both sides in terms of absolute count of lost audio frames.

on measurements conducted prior to the concert, we estimated that the audio driver and soundcard latency contribution is $\approx 5ms$, and that the receiver buffer added a delay of $18.34ms$ (Confidence interval: $[1.29,29.75]ms$) at Wrocław side, and $10.88ms$ (Confidence interval: $[1.13, 30.25]ms$) at Turin side. Thus, the overall M2E latency experienced by the musicians was within $[32,61]ms$.

Even though the development of MEVO is in its early stage, the overall feedback received from the musicians and the audience was that the system is already stable and transparent enough to introduce negligible impact on the perceived quality of experience. Nevertheless, the experiment highlighted that integration of PLC techniques is needed to address packet losses, that in some cases were clearly audible, and that further improvement of the audio buffer dynamic estimation methodology is necessary.

Chapter 6

Introducing DUST: a Dataset of real-time UDP Sound packet Traces

The distributed concert presented in Chapter 5 demonstrated the practical viability of the MEVO system under real-world performance conditions. However, systematic improvement of NMP algorithms, particularly packet loss concealment, jitter buffer adaptation, and clock drift compensation, requires realistic data that captures the characteristics of actual network transmission. This chapter presents DUST, a dataset collected from MEVO sessions under varied network conditions. A distinguishing feature is the inclusion of local playback buffer state alongside packet arrival timestamps, enabling researchers to determine not only which packets arrived late but also which contained audio samples that could still have been partially used. Furthermore, the playback buffer state, coupled with packet arrivals, permits to also model sound cards' clock drift.

6.1 Introduction

Real-time audio streaming over the internet is employed by a wide range of applications, such as videoconferencing and Networked Music Performance (NMP) tools. While deferred audio streaming, as adopted in music on-demand services, primarily focuses on preserving the quality of the received data, real-time audio streaming imposes the additional constraint of maintaining a low latency. For NMP applications, this constraint is particularly stringent, arising from the necessity to ensure a seamless and responsive user experience. The reference metric for this purpose is the mouth-to-ear (M2E) latency.

The basic setup of a real-time audio streaming application involves an audio

source device at one location and an audio destination device at another (remote) location, interconnected via a telecommunication network such as the Internet. The best-effort design of the Internet Protocol (IP) does not permit to concurrently achieve timely and error-free data delivery, thus introducing a trade-off between these two objectives.

More in detail, factors such as network configuration, link-layer connection type (e.g., Ethernet, Wi-Fi), network congestion, and routing changes influence packet delivery timing, creating observable *network jitter*. Network jitter, defined as the variation in packet inter-arrival times, can significantly impact the performance of real-time applications. To mitigate such impact, the receiver end of a real-time audio streaming system usually features a playout buffer, an in-memory buffer designed primarily to smooth out the jitter effect.

No guarantees are provided regarding the actual packet delivery, leading to potential late or lost packets, whose audio content cannot be reproduced in due time. Such *packet losses* deteriorate the quality of the audio playback by introducing audible artifacts. Therefore, Packet Loss Concealment (PLC) techniques aimed at mitigating the perceptual impact of missing data portions in a multimedia stream, due to either delayed or lost packets, need to be adopted.

Finally, since communication occurs between two distinct machines, two separate sound cards are involved for audio capture and playback. Despite operating at the same nominal frequency, the clocks of the two sound cards are never exactly identical. Since time synchronization strategies typically cannot be applied, this leads to the necessity of tackling the problem of clock *drift* [133].

Even though these three factors can be modelled independently, in a real scenario they influence each other. Indeed, a high jitter has the potential to cause local buffer underruns. Additionally, drift must be corrected to prevent unrecoverable overruns and underruns. Finally, packet loss can only be handled if promptly detected, i.e., if the buffer is large enough to permit the application of any recovery strategy in due time.

To the best of our knowledge, currently publicly available datasets consider multimedia streaming over HTTP of pre-recorded material (e.g., [137]) or videoconferencing applications (e.g., [164]), instead of NMP frameworks, and predominantly focus on the analysis of one of the three above-mentioned aspects (e.g., [56]), but do not permit to evaluate the intrinsic interconnection among them. Moreover, packet traces are typically constructed under the hypothesis that every packet can be either lost or correctly received, under the common assumption that a late packet is treated as a lost packet. However, this assumption does not necessarily hold in a real-time audio streaming scenario. A packet that arrives too late w.r.t. the due playout time of the first audio sample in its payload may still contain a portion of samples whose playout time has not yet expired. In such a case, the packet content should not be entirely discarded, as part of the carried audio signal could still be reproduced in due time.

To address such shortcomings, this chapter presents DUST: a dataset of network packet traces transmitted by a NMP application (i.e., MEVO [217]), which leverages UDP as a transport layer protocol. The traces were collected under varying network conditions using different audio devices, and were captured and stored at the receiver side. DUST has been created with the purpose of enabling simulations of the behavior of a real-time audio streaming application in realistic network conditions. The dataset is designed to integrate information about the playout process with audio packet arrivals captured in successive snapshots. This enables the simultaneous simulation of the reception or loss of a number of samples smaller than the packet size and an accurate quantification of the clock drift. In turn, such features permit a more thorough evaluation of the performance of state-of-the-art PLC techniques.

The remainder of the chapter is organized as follows: Section 6.2 provides details about the dataset, the data collection method, and a description of its structure and content. Section 6.3 provides some details on how to process the dataset to quantify drift and sample losses.

6.2 The Dataset

6.2.1 Data Collection Method

Data has been collected leveraging a modified version of the MEVO system. The system features a Raspberry Pi 4B (RPI) as computing device and an off-the-shelf professional sound card. The RPI is connected to the Internet by means of a wired Ethernet connection. The RPI runs the Raspberry Pi OS with the Linux PREEMPT_RT kernel. The MEVO software running on it controls the sound card through the in-kernel ALSA driver and sets the hardware parameters of both the capture and playback interface to the lowest possible hardware buffer size, and the sampling rate to 44100 Hz. Since sender and receiver features can be independently set, in the following we will describe the behavior of the two processes separately, from a software perspective.

Sender Process

As soon as a chunk of audio is ready to be captured, it is read and stored in a small memory buffer by the capture thread `TC`. According to a given policy, after a sufficient number of samples (usually between 30 and 132) have been collected, a second thread, sender thread `TS`, stores them in a UDP packet, using a simple custom application layer protocol, and sends it to the remote receiver process.

Receiver Process

Upon reception of a packet from the network, network thread `TN` extracts audio samples from it and enqueues them in the playout buffer. Playout thread `TP` constantly reads audio samples from the playout buffer and outputs them through the ALSA driver. For the sake of the data collection, the behavior of `TN` has been modified to collect a snapshot of the playout buffer state and of the incoming packet when enqueueing it. The snapshot collection process has been designed to be computationally lightweight, to minimize its impact on the measured quantities. `TN` writes the snapshot to memory, and another (logger) thread `TL` is in charge of batch writing the snapshots to mass storage as lines of a CSV file (see Fig. 6.1). During the data collection process, the audio has been monitored at regular intervals to guarantee it was correctly audible at both sides.

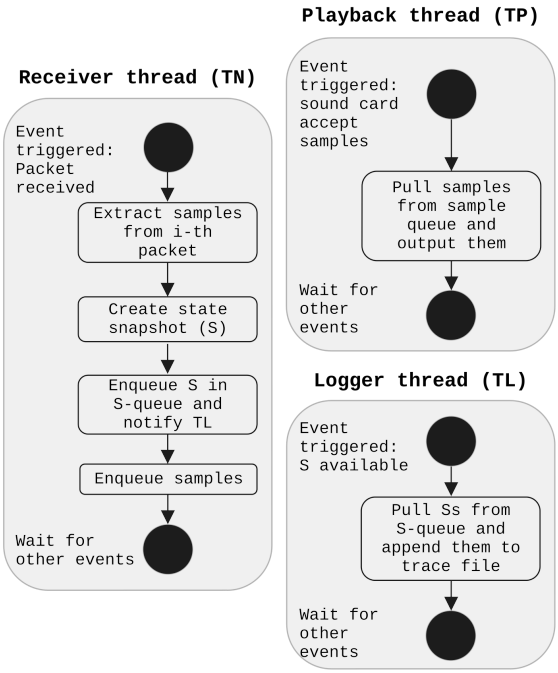


Figure 6.1: Data collection diagram

6.2.2 Dataset Structure and Content

The dataset is a collection of CSV files containing one record for each received packet. Each line consists of three columns:

- **Read index**: the number of samples read (consumed) from the buffer by `TP`.
- **Send index**: the index of the first sample in the packet being enqueued by `TN`.

- **Timestamp**: a local timestamp measured in nanoseconds, according to the RPI real-time-clock, relative to the first reception event by `TN`.

No data cleaning has been performed. All initial values are set to 0. For each experiment, three files are generated: a pair of CSV files, adhering the aforementioned format, produced by the two communicating devices, where each device acts both as sender and receiver, plus a third file that offers a high-level description of the experiment by providing the following details:

- The models of the sound cards used
- The locations of the experiment
- The duration of the experiment, expressed in hours (ranging from 1 to 9)
- The type of the *last mile* Internet connection
- Further comments

At the time of writing, three different sound cards were used:

- Behringer UMC404HD
- Scarlett 2i2 (1st generation)
- Audient ID14 MKII

Experiments have been performed through non-exhaustive combinations of network conditions. We combined a Fiber-To-The-Home (FTTH), Fiber-To-The-Cabinet (FTTC) and an enterprise network. Additionally, we collected data on a direct Gigabit-Ethernet connection between the two RPIs, using the same model of sound card at both sides to provide a baseline of the expected behavior. Some traces present behaviors which are strongly influenced by a third agent (e.g., another device concurrently sharing the same network access link).

6.3 Information Extraction and Utilization

6.3.1 Clock Drift and Jitter Characterization

Each file enables the extraction of the drift between the devices' clocks and the characterization of jitter. The drift d is generally expressed in parts-per-million (ppm). For audio devices, drift can be measured using the number of samples as a time indicator. Given a file of length N with remote device D_R and local device D_L, a possible simple definition of the average drift of the clock of D_R w.r.t. the clock of D_L can be given by the following formula:

$$d = 10^6 \left(\frac{S_{N-1}}{R_{N-1}} - 1 \right) \tag{6.1}$$

with S and R being respectively the *send index* and the *read index* (see Section 6.2.2), and their subscripts indicating the packet number (starting from 0).

Assuming a certain behavior of d (e.g., supposing that it remains constant during the experiment), it is possible to define a function f such that $\overline{S_i} = f(R_i)$ with $\overline{S_i}$ being the estimated value of S_i considering the drift. If d and f are representative of the behavior of the two devices, then it is possible to use the value $\epsilon_i = S_i - \overline{S_i}$ to characterize the jitter observed by the receiver.

6.3.2 Sample Loss Simulation

Identifying lost samples in the audio playout is pivotal for the assessment of PLC methods. When referring to audio streams, sample losses can be generated by three possible causes:

- *Hardware/Software (machine) faults* at sender side, for instance caused by a buffer overrun/underrun due to processor overload;
- *Packet drops* introduced by intermediate routers in the telecommunication infrastructure, when using a non-reliable transport protocol, i.e., UDP;
- *Latency spikes*, meaning that a chunk of audio samples, encapsulated in one or more consecutive packets, arrives at the destination significantly later than expected.

Publicly-available datasets used for packet loss simulation typically identify packet loss by assigning a boolean value to each packet to indicate whether it was correctly received. Differently, the information collected in DUST makes it possible to simulate more refined scenarios, e.g., where a portion of the data carried by a late packet can still be used for audio playout. For instance, suppose a packet carrying 128 samples being 1 ms late w.r.t. its expected playout time. In this scenario, supposing that the sampling rate is 44.1 kHz, the application could still be able to reproduce $128 - \lceil 1 \cdot 44.1 \rceil = 128 - 45 = 83$ samples contained in such packet. This may have a non-negligible impact on the performance of a PLC technique.

To use DUST for sample loss simulation, a dedicated simulator must be created. The simulator would produce a bit-stream (a stream of boolean values) associated to a given file of DUST. Each value of the bit-stream would mark the correct/wrong reception of the corresponding sample. Fig. 6.2 provides an illustrative example of how the simulator could produce the bit-stream. We highlight that the simulation output is not unique, as it is strongly dependent on the buffering and drift-correction policies implemented in the simulator. We can safely assume that there exists a controller that applies a correction to the packet's *send index* S_i, incorporating the buffering policy and the drift correction. A simple approach to implement such controller may assume a fixed user-defined target value for the buffer size (in number of samples) and the presence of an oracle that provides a bias B to be

applied to S_i based on the elapsed time and the prior knowledge of the clock drift d between the two sound cards. In Eq. 6.2, we show how to obtain a value W_i, defined as the *write index* of the incoming i-th packet, from S_i with the parameters B and d (expressed in ppm).

$$W_i = B + \frac{S_i}{(1 + \frac{d}{10^6})} \tag{6.2}$$

W_i is directly comparable to R_i such that

$$L_i = \begin{cases} 0, & \text{if } W_i \geq R_i \\ \min(R_i - W_i, C), & \text{otherwise} \end{cases} \tag{6.3}$$

with C being the number of samples per packet and L_i the number of initial lost

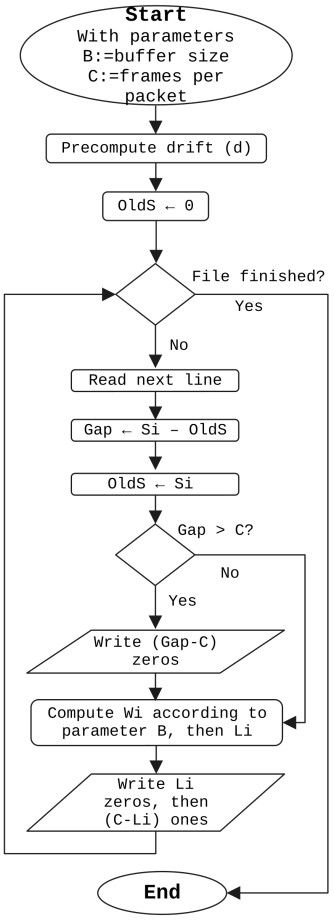


Figure 6.2: Example flow-chart for a sample loss simulator, given a file of DUST and parameters B (buffer size) and C (samples per packet). An output value of zero marks a lost sample.

samples of i-th packet. Fig. 6.3 shows an example of the behavior of $W_i - R_i$ assuming $B = 128$. We also assume that there are no out-of-order packets; otherwise, the file must be sorted in ascending order by the *send index* column before being fed into the simulator.

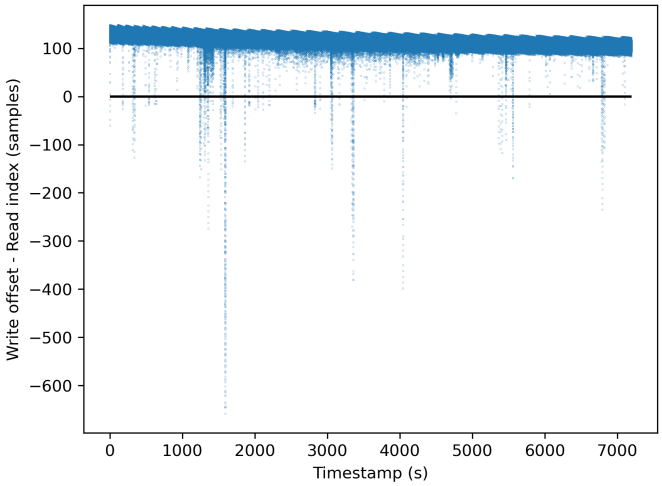


Figure 6.3: Example of the evolution of $W_i - R_i$ over time, assuming $B = 128$. Negative values indicate the presence of lost samples.

It is worth mentioning that the above described approach does not take into consideration potential machine faults, therefore, even though applicable to the files included in DUST, it does not have general validity.

Chapter 7

Assessment of Recovery Journal-Based Packet Loss Concealment Techniques for Low-Latency MIDI Streaming

The DUST dataset presented in Chapter 6 captures packet traces for sampled audio streams. However, musical interaction over networks can also rely on symbolic MIDI data, which presents fundamentally different challenges. MIDI represents discrete events whose loss may leave the receiving synthesizer in an incorrect state. The recovery journal mechanism, present in the RTP-MIDI standard, provides a framework for state recovery, but its configuration involves trade-offs between traffic overhead and sender-side complexity that this chapter quantifies.

7.1 Introduction

The MIDI standard can be used in a networked environment, where the instrument and the synthesizer are connected via the Internet, as described in Sec. 7.3. The MIDI specification defines a standardized file format for storing and retrieving MIDI data. Messages are organized into tracks, where each track contains a temporally ordered sequence of events corresponding to the performance actions captured by the instrument. In each track, up to 16 independent channels (channels could be associated, for example, with different instruments) can be defined. In the track, messages are interleaved with time information: time is encoded as variable-length

big-endian integers (1 to 4 bytes, for each byte 1 bit is used as prefix and 7 bits are used for value encoding) representing the difference in terms of ticks (fraction of quarter notes) between two subsequent events (0 for simultaneity). The first byte, also called "Status Byte", brings information about the type of musical event (e.g., note or command) and indicates which channel it belongs to. For each status byte, the count and the meaning of the successive bytes is documented in [158].

MIDI instruments can be modeled as Finite State Machines (FSM), where the set of possible states includes all the possible combinations of values of their controllable parameters. In other words, every combination of parameter values corresponds to a distinct state. RFC4696 [134] provides examples of implementations of the MIDI state concept in the context of the RTP-MIDI protocol. In this chapter, we define a MIDI system *state* $S(t)$, as the set of values assumed by MIDI parameters at a given time t, mainly focusing on *Note* and *Control Change* as classes of parameters that constitute a MIDI state. The MIDI state permits to identify a set of MIDI events that can lead the MIDI system to such state starting from any other state, including the default idle one, with no active events: in FSM jargon, this set of events represents a *transition*. Such events are those producing an active effect on the audio playback (e.g., an active note is a note which has been triggered by a Note On event generated at a time instant $t' < t$ and has not yet been stopped by a correspondent Note Off event). The system state can therefore be considered as a sort of “snapshot” of active/inactive MIDI events in the system, which, however, does not provide any information about the starting time or duration of any specific event.

7.2 Related Work

Ensuring reliable MIDI transmission over unreliable networks requires mechanisms to detect and correct inconsistent states at the receiver. The scope of this chapter is limited to the RTP-MIDI protocol [135], leaving aside alternative symbolic-control protocols such as OSC [264]; the literature review that follows surveys the main PLC approaches proposed for this context. In [135], Lazzaro et al. proposed a protocol based on RTP [208] to encapsulate MIDI events, as well as the usage of a so-called “Recovery Journal” (RJ) to implement PLC. The RJ encodes the difference between two states of the system. Let us denote as sender the application that produces MIDI events and sends them over the network, and as receiver the application that receives, synthesizes, and reproduces them. Considering states S_1 and S_2, where S_1 is the state of the sender at t_1 and S_2 the one at t_2, with $t_1 < t_2$, the RJ for states S_1 and S_2 represents the set of differences between S_2 and S_1. In the remainder of this chapter, such set of differences will be referred to as ΔS. Upon construction of an RTP packet, the RJ is calculated by considering S_2 to be the sender state just before the occurrence of the events carried in the packet under construction (which RJ is appended to), whereas S_1 is

chosen according to a policy negotiated at the beginning of the session: the default policy is *closed-loop*, which implies that the sender expects the receiver to send an acknowledgment whenever it receives a packet. Upon reception of an acknowledgment related to a packet generated at time t_i, $S_1 := S_i$ applies. This method constitutes the core of the RTP-MIDI protocol described in RFC-4695. Several software solutions already handle MIDI, for instance SonoBus [42] or HPSJam [107], which use a custom protocol to deliver MIDI data over the network. Instead, other Digital Audio Workstations (DAWs) can route MIDI messages through the network thanks to the implementations of the RTP-MIDI protocol offered by some operating systems, according to RFC-4695 (e.g. the Ableton Live DAW can stream MIDI thanks to the Apple driver for MacOs which implements the RTP-MIDI protocol [214, 194]).

A different PLC method specifically tailored for NMP over wireless telecommunication networks is presented by Virolainen at al. in [252]. It consists in categorizing MIDI messages to be sent as critical or non-critical. Messages belonging to the critical category are transmitted using a reliable transmission protocol, whereas non-critical MIDI messages are transmitted using a non-reliable transmission protocol. Therefore, this approach suggests a classification of MIDI messages, based on their priority level: for example, a Note On message belongs to a non-critical category, while the corresponding Note Off message belongs to a critical category. Thereby, delays/losses of messages transmitted through the unreliable channel could lead to the non-reproduction of MIDI events (e.g., a note is not played), while errors on messages transmitted through the reliable channel could lead to a delayed action (e.g., the duration of a note could be altered).

A variation of the above-mentioned method is proposed in [251], which implements a severity assessment phase carried out by the receiver, upon detection of a packet loss. If the error is considered severe enough, a recovery phase is activated, which involves replacing missing MIDI data with recovery data transmitted along an auxiliary channel. Recovery data is assumed to be generated by a dedicated server in real-time or pre-calculated.

Another approach for PLC of MIDI data specifically tailored for transmission over wireless networks is described in [166]. It leverages a packet acknowledgment mechanism that ensures retransmission of audio data that did not reach the receiver.

Differently, our proposed method operates in *open-loop*, i.e., it does not require the implementation of any acknowledgment mechanism, so that it can also be deployed on simplex channels. Moreover, it does not require two communication channels characterized by different transmission reliability guarantees nor the discrimination between critical and non-critical messages. In addition, the wireless communication channels considered in [252] adopt radio frequency or optical (e.g., infrared) transmission, whereas our proposed solution is agnostic to the transmission technology adopted at the physical layer.

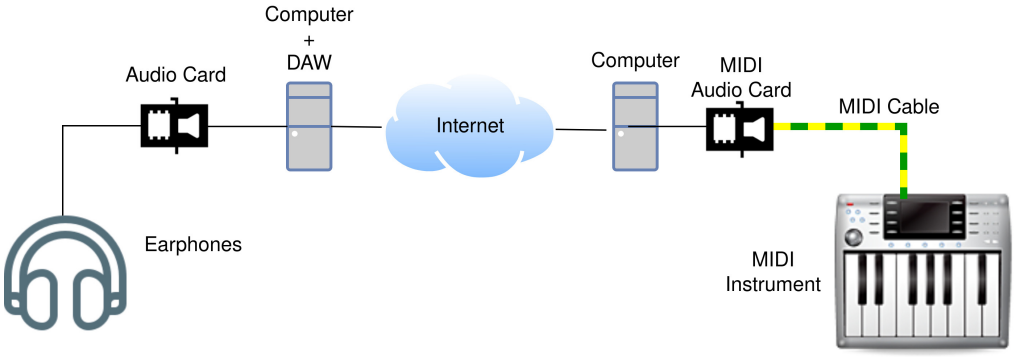


Figure 7.1: Example setup for a MIDI-based NMP session in which MIDI commands are produced by the MIDI piano, received by a local computer through a MIDI audio card and sent over the Internet, then received by the remote musician's computer, synthesized as analog audio by a Digital Audio Workstation (DAW) and reproduced through a local audio card.

7.3 MIDI NMP System Architecture

Access to an NMP session is typically negotiated between the software used by the musician and a server dedicated to NMP session management, which provides the joining musician with the list of (IP, port) pairs of the devices of all the remote musicians participating in that session and manages the connection setup. An example of an audio architecture for a MIDI-based NMP session is depicted in Fig. 7.1. For the sake of simplicity, in this description we limit the number of musicians to two and we only show a unidirectional flow from the right musician's instrument to the left musician's earphones. The instrument played by the local musician is connected to a device that runs a dedicated NMP transmitter application, which communicates through a telecommunication network with the NMP application running on the device of the remote musician that receives the audio data.

In a realistic scenario, two (or more) musicians acquire/transmit their own MIDI data and receive/reproduce the data of their remote counterpart(s). In such a case, the NMP application executed by each musician acts both as sender and receiver. The sender process is responsible for conveying the MIDI data generated by the musician's instrument to the remote player. Conversely, the receiver process obtains MIDI messages contained in the data stream generated by each of the remote musicians. The receiver application is responsible for concealing the effects of possible packet losses in each stream independently.

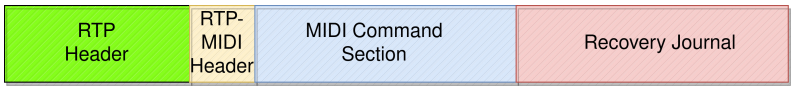


Figure 7.2: High level packet structure, according to RFC4695. The RTP Header is the one described in RFC3550, without header extensions.

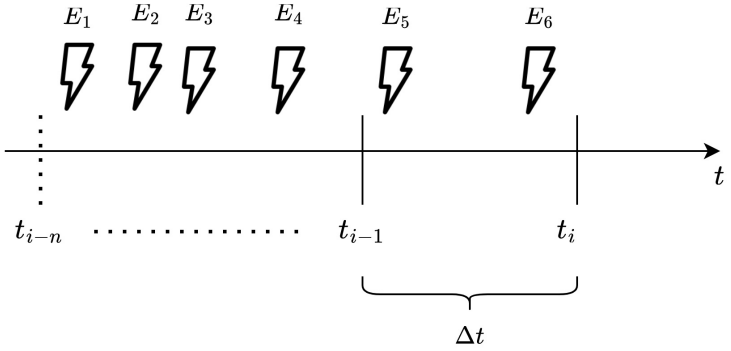


Figure 7.3: An example of RJ construction: using Δt as grouping period and assuming that a new period starts at time t_i, the Command section will contain E_5 and E_6 whereas the RJ will encode information about the difference between the system state at t_{i-1} and the system state at time t_{i-n} (including events from E_1 to E_4)

7.3.1 Communication Protocol and Message Format

We assume the usage of the RTP-MIDI application-level protocol running over UDP, as in RFC4695 [135]. RTP-MIDI deals with possible out-of-order datagrams by including a sequence number. This is provided by the RTP standard reported in RFC3550 [208], which implements source identification (and multiplexing), packet ordering, and timing. To send MIDI messages over RTP, we will refer to the message format described in [135], which takes into account all the possible scenarios that may occur during a MIDI session.

The high-level structure of the application layer packet, including an RTP and an RTP-MIDI header, a MIDI Command section and an RJ section, is depicted in Fig. 7.2. The sending algorithm [76] implemented by the NMP transmitter requires defining a grouping period Δt, which is the time interval between the generation of two consecutive packets. Straightforwardly, the longer the grouping period, the less the overhead generated by both the application and transport layers, but increasing the duration of the grouping period also leads to an increment of the perceived mouth-to-ear latency and to larger packet sizes (with the risk of exceeding the Maximum Transmission Unit). Practical values of the grouping period are in the order of few milliseconds [191], even though they are likely to

imply the transmission of several empty packets, since distinct musical events are usually temporally interleaved by at least some tens of milliseconds [73]. During each grouping period Δt, the sender process of the NMP application gathers all the MIDI messages representing the musical events generated by the musician's instrument, and populates the MIDI Command section of the packet, which follows the same data representation adopted for file tracks (see Sec. 7.1). Upon generation of a MIDI event by the user's instrument, the system state also gets updated. Conversely, the RJ Section of the packet implements a representation of the set of differences between system states (ΔS). Referring to Fig. 7.3, the RJ does not include events that are carried in the Command section of the packet it is appended to. Instead, it encodes the difference between states S_{i-1} and S_{i-m}, where S_{i-1} is the system state at time instant t_{i-1} (i.e., the beginning of the most recent grouping period) and S_{i-m} is the system state at time instant t_{i-m} (i.e., the beginning of m-th preceding grouping period). The choice of n determines the adopted policy. Ref. [135] focuses on the *closed-loop* policy, considering it as the default policy, and envisions as alternatives an *anchor* policy and an *open-loop* policy. The *closed-loop* policy expects the receiver to send back acknowledgments in forms of RTCP - Receiver Report (RR) about received packets. Whenever a RR for packet j is received, the RJ for packet i is computed as the difference between state S_{i-1} and state S_j. The *anchor* policy is the simplest one: every RJ is computed as a difference between states S_{i-1} and S_0. It is important to remark that, since in the *anchor* policy ΔS is computed as the difference between the sender's current state, S_{i-1}, and the initial (default) state, S_0, the receiver can adjust its current state (which we denote as S_c), even if such state is different from the default one. The steps necessary to change S_c to S_i, only knowing $\Delta S = S_{i-1} - S_0$ and the MIDI commands occurred in between time t_{i-1} and t_i, can be easily implemented as described in section 7.4.2. The *open-loop* policy lets the sender decide autonomously how to set m, which is then kept constant. This policy may lead to unrecoverable losses in the playback (for example, when more than m consecutive packets are lost): according to Ref. [135], this can be partially prevented in the initial negotiation phase by specifying a split for the set of MIDI Commands, according to which a subset of commands (e.g., Channel Volume (Controller 7)) must be sent in every RJ, as if the policy were *anchor*-based.

7.4 The Proposed PLC Policy for MIDI Streaming

In the following, we describe our proposed PLC policy, which enhances the *anchor* policy defined in Ref. [135] while avoiding the risk of introducing unrecoverable artifacts of the *open-loop* one (see again Ref. [135]), despite operating in open-loop fashion as well as the *anchor*-based one. Indeed, differently from the *closed-loop*

policy in Ref. [135], our proposed approach does not require any acknowledgment mechanism.

7.4.1 Enhanced Anchor Approach

Ref. [135] recommends the usage of the *closed-loop* policy, as it is the least bandwidth-consuming one. Indeed, the *anchor* policy implies a much larger overhead, due to the fact that two states interleaved by one or a few grouping periods usually differ by a limited number of events, whereas the number of differing events between a generic state S_i and state S_0 might be significant. Conversely, the usage of the *open-loop* policy is discouraged since it guarantees the correct recovery of the system state only under further assumptions (see section C.2.2.3 of Ref. [135]).

To reduce the introduced overhead, we propose to make the sending of the RJ non-compulsory. In our proposed *enhanced anchor* approach, we define a negotiable value k called *refresh rate*, which generalizes the *anchor* policy by imposing to send one RJ every k packets. The canonical *anchor* policy can now be seen as a special case of the *enhanced anchor* policy where $k = 1$. This new policy can be easily implemented with the same packet structure defined in [135]. With the *enhanced anchor* policy, the RJ section in Fig. 7.2 becomes optional, as it is present only once every k packets. Let t_0 be the time at which the session starts. Packets are generated and sent to the remote musicians at times $t_i = i\Delta t + t_0$ (with i positive integer) but only packets where $i = nk$ (with n positive integer) shall include the RJ Section. Note that in our proposed approach the RJ is computed as in the *anchor* policy described in [135], i.e., it encodes the differences between S_{i-1} and S_0.

7.4.2 State Maintenance and Update Procedure

To enable the receiver process to perform any correction, both the sender process and the receiver process have to keep track of the current system state. As depicted in Fig. 7.4, at each RJ reception the receiver must verify the correctness of its current state S_c by comparing the *parameters* (where the word "parameter" is meant to be an umbrella term that encompasses note velocities, control values, etc.) contained in the RJ to those that characterize S_c and by updating them with the value contained in the RJ in case of mismatch. This procedure fits both the *enhanced-anchor* policy and the *closed-loop* policy. The transition to the correct state is then performed by generating all the MIDI events that modify the system state from the "old" state S_c to the "new" correct one $S_{c'}$ (where$S_{c'} = S_{i-1}$), that will be considered as the current state from that moment on (i.e., $S_{c'}$ replaces S_c).

Conversely, the sender's current state is stored in memory, and it is updated whenever a new MIDI message is generated by the musician's instrument. Our

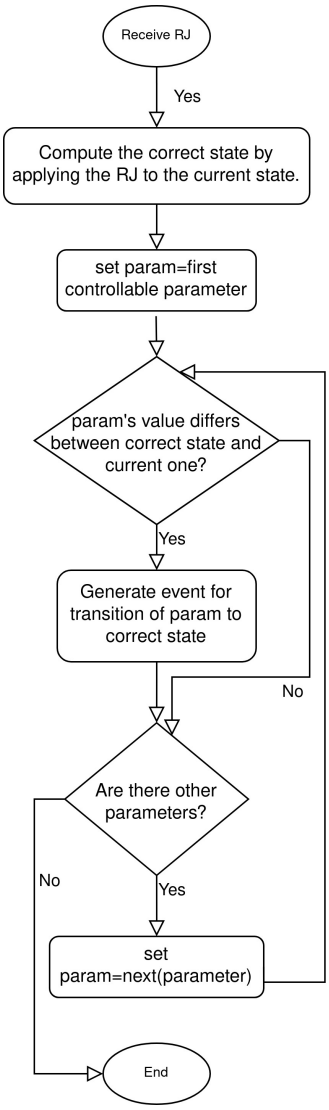


Figure 7.4: The algorithm to be implemented at the receiver side to perform packet loss concealment.

proposed *enhanced anchor* policy introduces a simplification in the sender's implementation compared to the *closed-loop* policy, as in the latter case the sender is responsible for memorizing S_i and also the most recent acknowledged state S_{i-n} (i.e., the state at the beginning of the grouping period carried by the last acknowledged packet) for every known receiver. Indeed, different receivers may provide RRs at different times, thus a dedicated RJ must be produced for every receiver, and for each of them the most recent acknowledged state should be updated upon RR reception. This is a known drawback of the *closed-loop* policy, as reported in section C.2.2.2 of [135]. Differently, our proposed *enhanced anchor* policy only

requires the sender to keep in memory exactly one state (i.e., S_i), regardless of the number of receivers.

Since several data structures to memorize a given state exist, Ref. [134] provides some guidelines for a possible implementation. From a high-level perspective, the state of the system should be a dictionary associating parameters (e.g. Notes) to their values (e.g. Note On with velocity = v, Note Off), although the suggested implementation is based on arrays for each parameter group (Notes, Controls, Programs...). This dictionary does not contain information about a parameter in case it has its default value. Straightforwardly, S_0 is represented by an empty dictionary. It is worth noticing that the sender's and receiver's software implementing the state-keeping cannot be agnostic about the specific semantics of MIDI messages. To motivate this statement, we provide the following example: let us assume we have a "dummy" software at the sender's side with no knowledge of the meaning of MIDI messages, and consider a basic assumption on Control change messages, i.e., that their default values equals 0. Assume that the MIDI Message with status byte = 121 is generated (its meaning is "all controllers off"). Our dummy software only understands that the status byte 121 is in the range of controls messages, and eventually modifies its state in the dictionary. This is not the right behavior, as it should reset to default all the others controllers that are in a non-default state.

The same holds at the receiver side. The receiver must, in fact, maintain the current state S_c of the local synthesizer. In case of lost packets, upon reception of an RJ, the receiver must be able to generate the messages necessary to adjust the synthesizer's state to match the received one, following the procedure reported in Fig. 7.4. The RJ representation proposed in Ref. [134] is meant to minimize the network bandwidth usage. It efficiently encodes the current state of all the parameters that changed their values. By doing so, if the starting state S_0 is the default one, the RJ brings all the information necessary to build the dictionary representing the current state.

7.4.3 Implementation Remarks

A simplified implementation guide for RJ-based PLC methods at the receiver side may be found in section 7 of RFC4696 [134], from which it emerges that the complexity of the PLC algorithm executed by the receiver grows linearly with the number of peers participating in the NMP sessions. Instead, for what concerns the implementation of our proposed enhanced anchor policy at the sender side, the sole task of the sender is to keep in memory a dictionary that reflects the MIDI instrument state. RFC4696 provides some hints regarding a possible representation in memory of such a dictionary, and how to build the associated RJ to be included in the packets that are required to contain it. As opposite to the closed-loop policy, which requires the keeping of a dedicated dictionary per each receiver, this data structure is the same for all the receivers. It follows that, at the transmitter side,

the complexity of the proposed enhanced anchor policy does not depend on the number of peers.

Concerning network characterization, we identify three factors that may have an impact on the performance of our method: *i*) *network latency*, *ii*) *jitter*, *iii*) *packet losses*. In the following, we discuss such factors one by one.

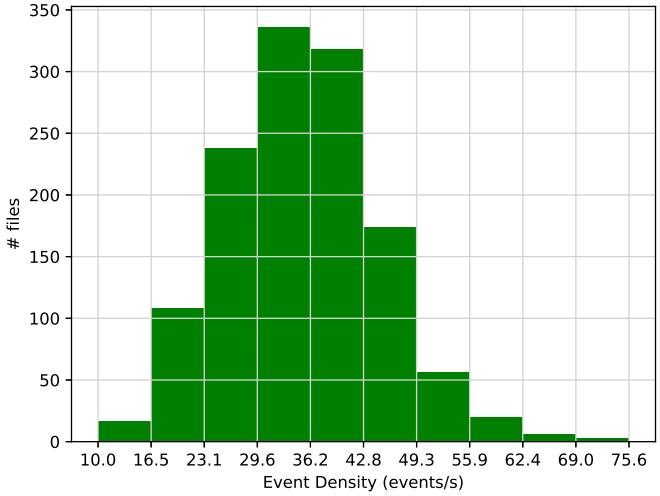
Network latency (i.e., the overall contribution to the mouth-to-ear delay due to transmission and propagation delays through communication link(s), as well as queuing and processing delays introduced by routers traversed by packets along their path from source to destination), does not have by itself a direct impact on the execution of the PLC algorithm: indeed, as reported in Fig. 7.4, the recovery journal is generated by the transmitter regardless of network conditions and compared by the receiver to the actual system state upon reception.

Differently, *jitter* may have an impact on the execution of the PLC algorithm, as the variation of the end-to-end delay experienced by consecutive packets may lead to an alteration of the packet sequencing (i.e., packets are received in a different order with respect to their generation sequence). If an out-of-order packet arrives too late to be reproduced by the receiver, it is practically equivalent to a lost packet. Therefore, by quantifying the packet loss probability, in our numerical assessment (see Section 7.5) we will capture the impact of both lost and late packets. To mitigate the impact of out-of-order packets, a buffer is typically maintained at the receiver to operate as "cushion": the larger the buffer, the higher the flexibility in re-ordering late packets, but the higher the increase of the mouth-to-ear delay.

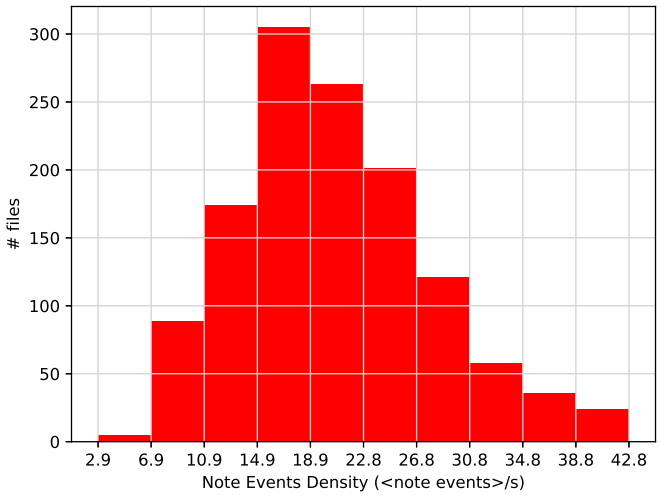
7.5 Performance Assessment

We provide a quantitative analysis of the performance of the proposed *enhanced anchor* PLC method in terms of similarity of the reproduced audio stream and network overhead, benchmarking it against the policies described in Ref. [135]. For the sake of reproducibility, the assessment is conducted on pre-recorded MIDI files instead of real-time generated MIDI streams. Our numerical assessment only takes into account the impact of packet losses and of delayed packets arrived too late to be reproduced. The impact of delayed but yet reproducible packets and of local clock drifts are not taken into account in this work since it is assumed that the MIDI streaming is implemented within a NMP system integrating an appropriate clock recovery mechanism (as proposed, e.g., in [68]) and adequate buffer size settings, capable of effectively managing the reordering of out-of-order packets. It is worth noting that the impact of clock drifting in MIDI streaming is milder than in audio streaming: in the latter case, clock misalignment may lead to buffer over/underruns that in turn generate audio glitches, whereas in the former case it may only cause alterations of the duration of active events, without introducing audio artifacts.

7.5.1 Dataset



(a) Empirical distribution of Events Density



(b) Empirical distribution of Note Events Density

Figure 7.5: Empirical distribution of (note) event density values in the considered MIDI dataset

We considered the Maestro Dataset v3.0.0 [100], which includes about 200 hours of paired MIDI and Audio recordings from International Piano-e-Competition events within nine years. For the purpose of this study, only the MIDI dataset was considered, consisting in 1276 MIDI files, with a duration that varies between 45 seconds and 43 minutes, with an average of 9 minutes. Being a piano dataset, it is representative of a NMP session using a MIDI keyboard, as it contains Note messages, one Program Change at the beginning of the track and Control Changes

messages affecting control 64 (Sustain pedal), control 67 (Soft pedal) and control 66 (Sostenuto pedal).

The dataset contains files with one track (namely one single track with MIDI messages representing MIDI events, hence not only meta-messages). Each file has a tempo signature of $120bpm$. It can be divided in two subsets based on the timing information. One subset has a granularity of 384 ticks per beat, the other has a granularity of 480. For simplicity, network overhead measures have been conducted on the first subset, using as time units integer multiples of the tick duration.

To evaluate the effectiveness of the *enhanced anchor* policy, we identify some features related to the complexity of the performed musical pieces, according to which we split our dataset. In particular, we characterize the musical pieces by means of two metrics: the *event density* and the *note event density*. The proposed *event density* parameter (ED) aims at quantifying the rhythmic complexity of the piece being played and is an adaptation of the event density parameter defined in [190, 132] as the average number of event onsets per second, where the term onset refers to the initial transient of a musical note (or of any other sound). In this study, since we are considering MIDI data, an event onset refers to the beginning of a generic MIDI event. The *note event density* parameter (NED) has the same meaning as the *event density*, except for the fact that only Note events are considered (i.e., only Note On/Note Off messages). Fig. 7.5 reports the distribution of *event density* and *note event density* values among the MIDI files comprised in the dataset. As we can see, the distribution of the ED is well fitted by a normal distribution with mean 35.19 and variance 89.56 whereas the NED has mean 20.7 and variance 50.9.

7.5.2 Evaluation Metrics for Similarity Assessment

The evaluation of the audio similarity achieved by the proposed *enhanced anchor* policy has been conducted by comparing the MIDI stream generated by the sender to the one played back by the receiver. For this assessment, the grouping time has been fixed to 3 ticks, which, in this dataset, correspond to $\approx 3ms$ or $\approx 4ms$, according to Eq. (7.1), where τ is the duration of a tick in milliseconds, bpm is the time signature of the track in beats per minutes and tpb is the tick granularity in ticks per beats.

$$\tau = \frac{6 \cdot 10^4}{bpm \cdot tpb} \tag{7.1}$$

To perform such comparison, the content of the original MIDI file is streamed by the sender process and a new MIDI file is generated by the receiver process. Such MIDI file consists in the sequence of all the MIDI events that are played back, with their corresponding time offsets. In the absence of any transmission error, the file transmitted by the sender and the file reconstructed by the receiver must be

Table 7.1: Loss probability p and corresponding confidence interval of 95% of measured loss ratio (CI)

p	0.01	0.2	0.3
CI	[0.008,0.011]	[0.194,0.203]	[0.291,0.301]
p	0.5	0.8	0.9
CI	[0.492,0.507]	[0.800,0.812]	[0.900,0.907]

identical. Conversely, if packet losses occur, the RJ section of a packet successfully received after one or multiple missing ones will be leveraged to conceal errors and to resume the correct system state. It follows that, in such a case, the file transmitted by the sender and the file reconstructed by the receiver will not be identical. For the sake of our analysis, packet losses are modelled as statistically independent events, i.e., each packet is assumed to be unavailable for reproduction at the due time (due to either a loss or an excessive delay) with probability p. It is worth mentioning that, considering a single file, the longer the file duration, the more closely the actual ratio of lost packets (l) to generated ones (g) approximates probability p: Table 7.1 shows the confidence interval of $\frac{l}{g}$ compared to p. The discrepancies between the two files are then quantified[1].

The comparison algorithm consists in comparing the state of the two streams, when reproduced simultaneously, on per-tick basis. For such comparison, we define a very simple state similarity function $e(t)$. Note that, for the sake of our similarity analysis, we consider a back-to-back configuration that introduces negligible mouth-to-ear latency between the sender and the receiver processes to avoid the need for time realignment among the two streams.

$$test(S_a, S_b) : = \begin{cases} 1 \text{ if } S_a = S_b \\ 0 \text{ otherwise,} \end{cases} \tag{7.2}$$

$$e(t) : = test(S_r(t), S_s(t)) \tag{7.3}$$

where $S_r(t)$ (respectively $S_s(t)$) is the state of the receiver (respectively the sender) at time t.

The stream similarity value s is then defined as:

$$s = \frac{\int_0^T e(t)\, dt}{T} \tag{7.4}$$

[1]Note that such evaluation method does not take into account the perceptual impact of transmission errors (i.e., any discrepancy between the two files is weighted equally). Future work will be devoted to the identification of more advanced performance evaluation metrics and subjective ratings.

Straightforwardly, formula (7.4) tends to 0 when the total time in which the two tracks produce the same musical output tends to 0, it tends to 1 when this time approaches the entire duration of the track.

We also define s_n, which follows the same definition as above except that we only consider note events (two states are equal if their set of active notes is the same, even if other events like control values may differ). This apparent simplification permits to exclude the impact of slow pedals variations. In other words, when a control command associated to a pedal (recall that this is the case for all the Control Changes in the reference dataset) is generated, the physical movement associated to the pedal pressure from up to down or vice versa spans, most of the time, multiple MIDI ticks (see Eq. 7.1), resulting in many intermediate Control Change events that do not significantly alter the reproduced audio stream (for example, giving an on-off interpretation to control change commands, by mapping the range of 128 possible values to a boolean variable which is set to 0 if the Control Change value is 0, to 1 otherwise, would imply no variation).

Note that the similarity evaluation analysis will focus only on the *enhanced anchor* and *anchor* policies. Indeed, as already mentioned, the *anchor* policy is obtained by setting $k = 1$ in the *enhanced anchor* policy, whereas the *closed-loop* policy behaves identically to the *anchor*-based one (in both cases, the receiver is capable of recovering the correct state of the system as soon as the RJ is received). The performance of the *open-loop* policy depends on the choice of m and on the burstiness of packet losses, but is surely upper-bounded by the performance of the *closed-loop* policy, so it is excluded from our analysis.

p	0.2				0.5				0.8			
k	1	2	3	1000	1	2	3	1000	1	2	3	1000
$ED \leq 27.25$	98.14	97.08	96.10	21.54	93.14	89.03	85.76	5.68	78.87	68.87	61.90	1.43
$27.25 < ED \leq 32.49$	97.61	96.28	95.02	19.04	91.27	86.22	82.25	4.51	74.05	63.01	55.81	1.09
$32.49 < ED \leq 37.21$	97.23	95.69	94.25	17.37	89.95	84.27	79.91	4.04	70.93	59.31	51.97	0.95
$37.21 < ED \leq 42.84$	96.91	95.19	93.60	15.96	88.88	82.67	77.94	3.57	68.33	56.18	48.64	0.79
$ED > 42.84$	96.27	94.21	92.32	13.28	86.81	79.62	74.25	2.95	63.59	50.60	42.86	0.68

Table 7.2: Similarity (%) obtained with the enhanced anchor policy for different packet loss probability values, depending on the event density range, assuming a grouping period duration of 3 ticks.

p	0.2				0.5				0.8			
k	1	2	3	1000	1	2	3	1000	1	2	3	1000
$NED \leq 14.71$	98.98	98.39	97.82	25.84	96.19	93.68	91.48	7.71	87.31	79.46	73.40	2.42
$14.71 < NED \leq 18.17$	98.61	97.82	97.05	22.98	94.86	91.56	88.69	6.57	83.43	74.12	67.31	2.20
$18.17 < NED \leq 21.82$	98.36	97.44	96.53	19.87	94.01	90.23	86.97	5.37	81.07	70.93	63.76	1.70
$21.82 < NED \leq 26.32$	98.08	96.99	95.94	17.85	93.02	88.64	84.93	4.70	78.23	67.14	59.65	1.63
$NED > 26.32$	97.46	96.05	94.68	14.84	91.60	85.56	81.12	4.35	73.22	60.99	53.15	1.67

Table 7.3: Note similarity (%) obtained with the enhanced anchor policy for different packet loss probability values, depending on the note events density range, assuming a grouping period duration of 3 ticks.

7.5.3 Evaluation Metrics for Traffic Overhead

To quantify the impact of the considered PLC policies in terms of traffic overhead, we computed the amount of bytes transferred at transport layer during the simulated performance, considering the overhead bytes introduced by the RTP-MIDI protocol and the 8-byte-long UDP header at each packet. For this evaluation, we consider a unidirectional MIDI stream from a sender to a remote receiver process. The parameters influencing the performance of the *closed-loop* policy are the Round Trip Time (RTT - i.e., the time elapsed from the start of the transmission of an RTP-MIDI packet to the reception of the corresponding RR), the grouping period Δt and the packet loss probability p. The RTT affects the *closed-loop* policy because it can delay the pruning of the RJ structure [134], especially when it is higher than the grouping period (i.e., when RTT$> \Delta t$). Moreover, every time a packet or its corresponding RR gets lost, the size of the RJ may increase. Conversely, the performance of the *enhanced anchor* policy is influenced by the grouping period Δt and the refresh rate k, whereas the packet loss probability is not influential because the content of the RJ section does not depend on the acknowledgment mechanism.

We focus on the *closed-loop* policy and on our proposed *enhanced anchor* policy, since the behavior of the *anchor* policy in terms of traffic overhead corresponds to that of the *enhanced anchor* policy with $k = 1$, whereas the behavior of the *open-loop* policy is analogous to that of the *closed-loop* one, provided that parameter m in the *open-loop* policy is tuned so that $m \cdot \Delta t =$RTT.

We compute the overhead by comparing S_{RJ} to S_{NRJ}, where S_{RJ} is the amount of bytes transmitted by the sender when the PLC policy under investigation is integrated in the RTP-MIDI protocol and S_{NRJ} is the amount of bytes transmitted by the sender when no PLC policy is adopted. It follows that S_{NRJ} depends only on the grouping period and on the amount of events in the track/performance. Depending on whether we include or ignore traffic generated by RRs sent by the receiver to the sender, the traffic overhead is defined as:

$$OH = \frac{S_R J}{S_{NRJ}} \tag{7.5}$$

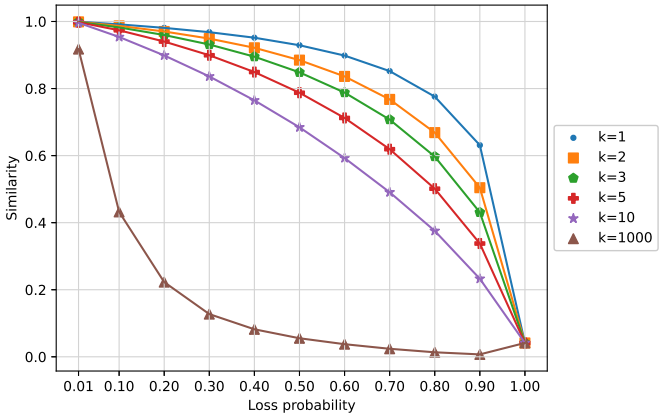
or as:

$$OH_{RR} = \frac{S_{RJ} + S_{RR}}{S_{NRJ}} \tag{7.6}$$

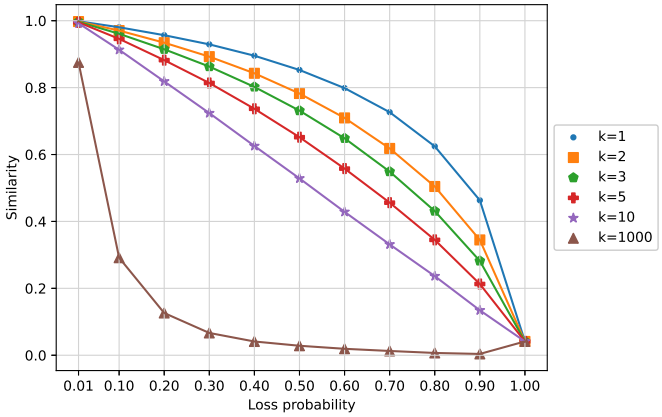
when backward traffic S_{RR} generated by the acknowledgment mechanism is taken into account ($S_{RR} = 0$ in the case of *enhanced anchor* policy).

7.5.4 Similarity Evaluation

Fig. 7.6 reports the similarity results obtained for different values of the refresh rate k, depending on the packet loss probability p, and show that similarity is



(a) With a grouping period of 2 ticks



(b) With a grouping period of 5 ticks

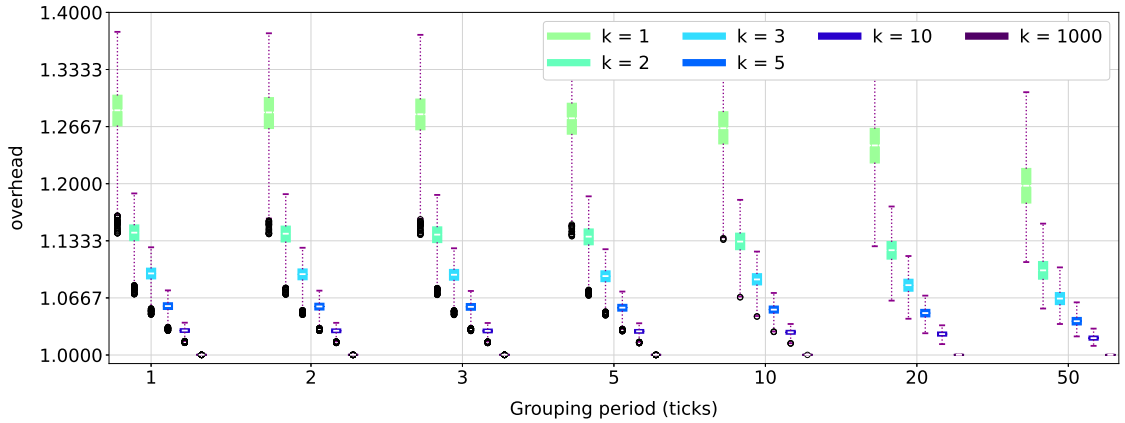
Figure 7.6: Similarity (%) obtained with the enhanced anchor policy, depending on the refresh rate and on the packet loss probability (confidence intervals are below 1.5% and thus not reported for the sake of readability)

heavily influenced by the refresh rate value. In particular, with a refresh rate $k = 1$ (which means that every packet carries an RJ, thus making the behavior of the policy identical to that of the *anchor* policy), the protocol achieves about 90% similarity when $p = 0.5$. This means that, even in case of a communication where half of the transmitted data is lost, it is possible to reach a very high fidelity in the final audio playback. Moreover, even with $p = 0.8$, the proposed PLC method reaches on average 70% similarity.

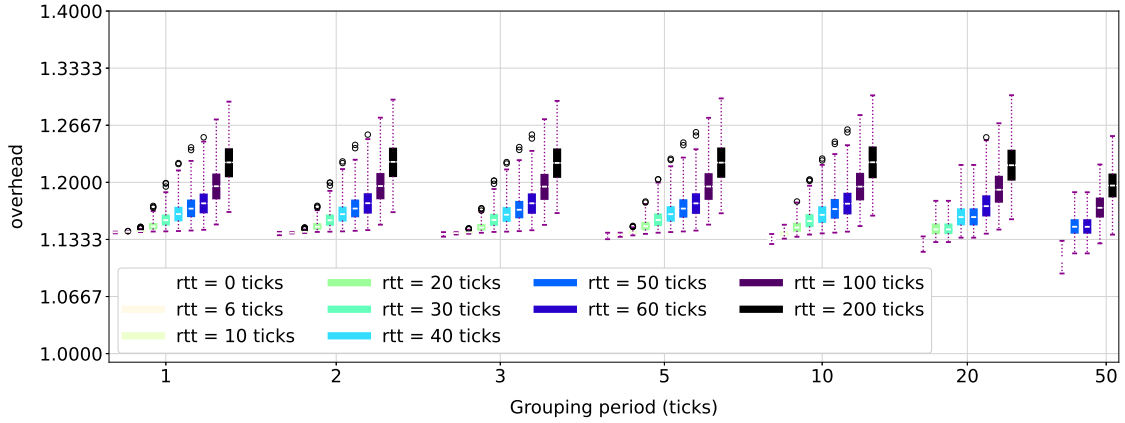
When increasing k, the similarity inevitably worsens. The benchmark case of a very high refresh period is also reported: with $k = 1000$ the similarity is about 4% when $p = 0.5$. Moreover, a 0.2 packet loss probability is sufficient to make the similarity drop to less than 20%. Results with $p = 0$ and $p = 1$ require particular

considerations: in both cases the achieved similarity does not depend on the refresh rate. This is because, for $p = 0$, there are no missing packets, while, for $p = 1$, all the packets are lost (no messages are received).

Note that, in the latter case, the similarity is higher than in the case with $p = 0.9$. This behavior is due to the fact that the absence of active events (i.e., silence) at both sender and receiver sides is considered as an identical state. Therefore, in the case of a reproduction characterized by total absence of events, the similarity represents the fraction of silence time among all the musical pieces. In cases with few packets received (e.g., when $p = 0.9$), a little amount of MIDI events is reproduced and, since the exact correspondence of silence periods is not preserved any longer, the average similarity is lower.



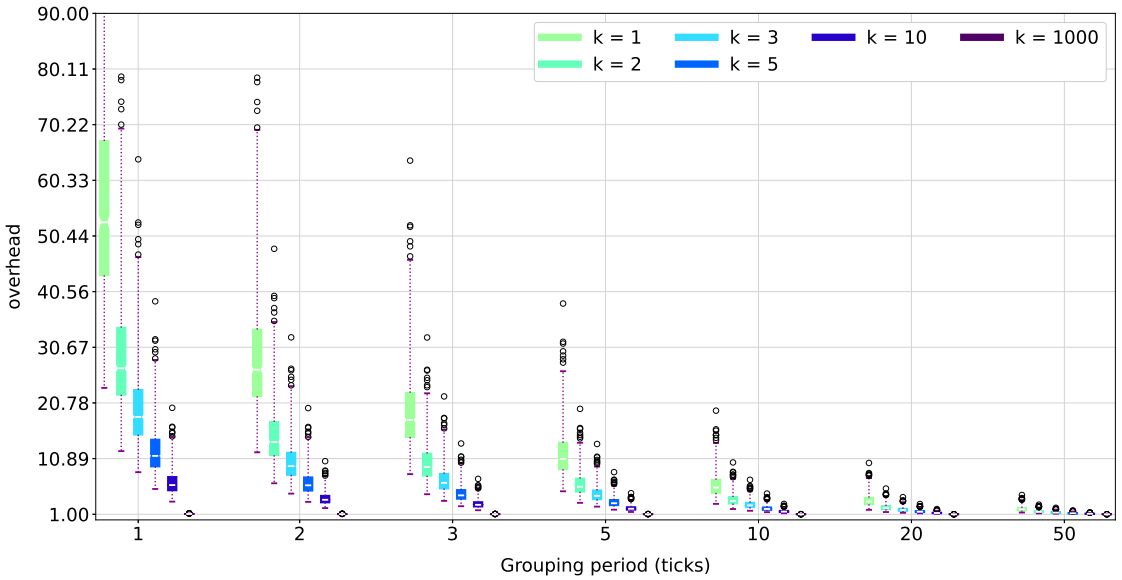
(a) Enhanced anchor policy ($k = 1$ represents the anchor policy).



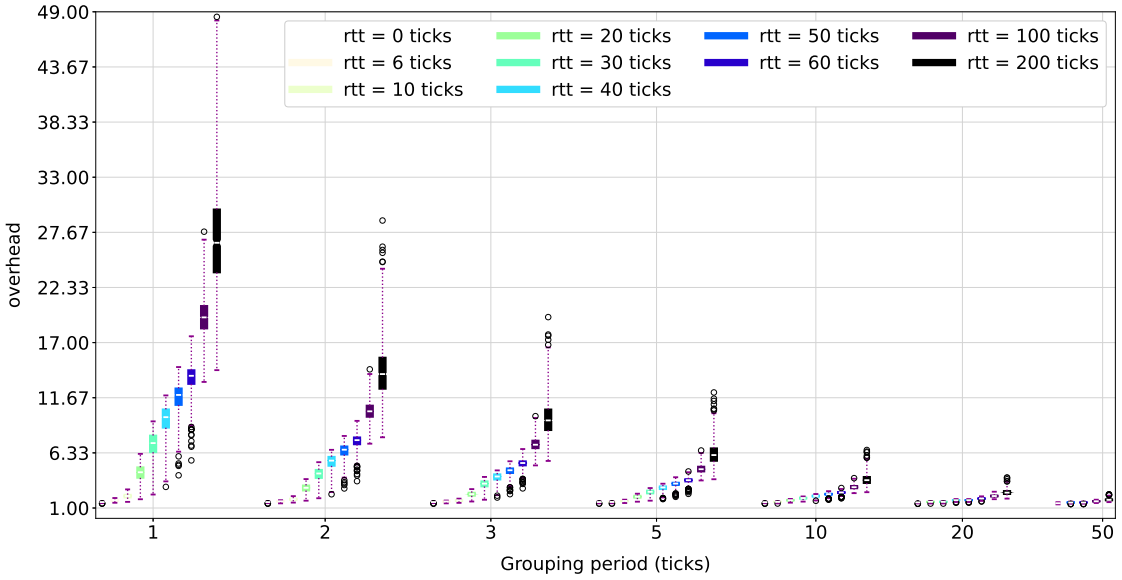
(b) closed-loop policy, assuming a loss probability = 0.

Figure 7.7: Transport level overhead with respect to a transmission with no RJ. Time is measured in MIDI clock ticks, $1\,tick \approx 1.3ms$.

To evaluate the impact of different event density and note event density values on the performance of the PLC method, the dataset has been subdivided into 5



(a) Enhanced anchor policy ($k = 1$ represents the anchor policy)



(b) closed-loop policy, assuming $p = 0.01$

Figure 7.8: Transport level overhead with respect to a transmission with no RJ, with the proposed efficient implementation. Time is measured in MIDI clock ticks, $1\,tick \approx 1.3ms$.

subsets according to their ED (resp. NED), by splitting the whole ED (resp. NED) range in 5 equally sized sub-ranges. The global similarity trends reported in Tabs. 7.2 and 7.3 for $p = 0.2, 0.5, 0.8$ and $k = 1,2,3,1000$ are similar to those reported

in Fig. 7.6, showing decreasing similarity as k increases. However, for a given packet loss probability and refresh rate, higher values of ED or NED lead to lower similarity. The motivation lies in the fact that, with a higher average number of events per second, the number of packets not containing any MIDI event decreases. Therefore, it is more probable that a lost packet carries one or more MIDI events.

7.5.5 Overhead Evaluation

Fig. 7.7 shows the traffic overhead introduced by the usage of the RJ Section, when ignoring RR-related traffic (OH). Note that, in Fig. 7.7b, the number of boxplots progressively decreases when $\Delta t \geq 10$. This is due to the fact that a scenario with $RTT < \Delta t$ is equivalent to the ideal scenario where $RTT = 0$ (i.e., RR acknowledging packet i is received before packet $i+1$ is ready for transmission).

In general, the *enhanced anchor* policy performs better when compared to the *closed-loop* one in terms of overhead, for $k \geq 3$. The lower bound of the overhead of the *closed-loop* policy is due to the fact that the Ref. [135] explicitly imposes that every packet must contain an RJ Section (even if empty). Note that the heaviest scenario occurs when $\Delta t = 1$ tick and $k = 1$ (*anchor* policy): in such situation, the highest average observed bitrate for a single stream was $38kB/s$, which is quite acceptable on modern networks, even compared to the worst simulated session with the *closed-loop* policy, which produced $22kB/s$ (obtained with loss probability $p = 0.8$, RTT$= 200\,ticks$, grouping time$= 1\,tick$).

However, as already mentioned, if the grouping period Δt is in the order of a few milliseconds, a significant amount of packets will have an empty Command section, since consecutive musical events are normally distanced by at least some tens of milliseconds. Therefore, with the aim of providing a more realistic estimation of the achievable bitrates, we also consider a more efficient sending policy by proposing an implementation that avoids unnecessary packet transmissions. Consequently, we repeat the computation of S_{RJ} and S_{NRJ} assuming that packets are transmitted only if non-empty. We define a non-empty packet as a packet which brings information useful for the playout at the receiver side. More specifically:

- for S_{NRJ}, non-empty packets are the ones that contain a non-empty MIDI command section;
- for S_{RJ} in the *closed-loop* policy, non-empty packets are those with either a non-empty RJ, or a non-empty Command section;
- for S_{RJ} in the *enhanced anchor* policy, non-empty packet packets are those either with a non-empty Command section or including an RJ (i.e., if, being i the sequential index of the packet, it holds that $i \, mod \, k = 0$).

We underline that the definition of non-empty packet with respect to the RJ section differs between the two policies. The reason behind this is that, in the *closed-loop*

policy case, the RRs act as an agreement between sender and receiver on the current state of the receiver. In such situation, an empty RJ is equivalent to an absent RJ as both mean that no events occurred at the sender side since the last acknowledged packet. It follows that, in the *closed-loop* policy, the variation of a parameter from a value different from the default one, to the default value, is a difference that must be encoded according to the rules of the protocol. Conversely, in the *enhanced anchor* policy, the sender is agnostic to the current state of the receiver. In this case, an empty RJ asserts that there are no differences between the sender state and the default state S_0, which is equivalent to saying that the sender is in the default state. When using the *enhanced anchor* policy, if a peer receives an empty RJ, it has to perform all the necessary actions (if any) to move to the default state. Thus, according to these observations, an empty RJ brings no information only in the *closed-loop* policy.

Moreover, in this new scenario we take into account the overhead generated by RRs (OH_{RR}) [2]. Results are reported in Fig. 7.8, which shows that with the aforementioned implementation the traffic overhead due to adoption of PLC policies substantially increases. In this case, the *enhanced-anchor* policy leads to higher overheads than the *closed-loop* one. Comparable or lower overheads are obtained by the *enhanced-anchor* policy for $k \geq 5$, when RTT$\geq 50ms$.

For a modern wired network connection, we can consider the following parameters for a plausible scenario:

- $\Delta t = 3\,ticks$ (≈ 4 ms assuming a tempo of 120 bpm and 480 ticks per beat), which is comparable to the time duration of a raw-audio segment (assuming stereo PCM audio with 16 bits encoding) conveyed by a packet of ≈ 700 bytes (i.e., approximately a half of Ethernet's Maximum Transmission Unit).

- $k = 3$, i.e., an RJ is sent approximately every 12 ms in the *enhanced anchor* policy. Scientific literature reports that the minimum time separation between two consecutive acoustic events ensuring that they are perceived as distinct is in the order of a few ms [103], and that a separation time of $15 - 20$ ms is required to make the listener able to report which of the two sounds preceded the other. Therefore, we speculate that setting $k = 3$ guarantees state recovery while making the perceptual impact of an incorrect state almost unnoticeable.

- RTT=30 ms, assuming a physical distance between remote players in the range of $500 - 1000$ km [37]: since the propagation time of a lightwave in an optical fiber is roughly 5 ms per 1000 km, and considering that queuing and processing

[2]The acknowledgment procedure at receiver side is implementation-specific, however if RRs are sent at a too low rate, the size of the RJ at sender side increases. In our scenario, we suppose that a RR is sent upon reception of every packet. Under different assumptions S_{RR} may decrease, at the expense of a potential increase of S_{RJ}.

delays at intermediate network nodes must be taken into account, as well as MIDI data acquisition and buffering delays at the two ends, we believe that such figure is representative of a wide range of practical NMP scenarios, with users placed at most some hundreds of km apart. It is worth remarking that the computational timings of our proposed PLC method are in the order of microseconds, thus they represent a negligible component of the mouth-to-ear latency.

- $p = 0.01$, i.e., we assume 1% of lost packets[3].

Under these assumptions, we measured an average bitrate of 2.62 kB/s (Confidence Interval: (2.27,2.89)) for the *enhanced anchor* policy and 1.37 kB/s (Confidence Interval: (0.48,2.58)) for the *closed-loop* policy[4]. Based on all the above reported results, it emerges that the proposed *enhanced anchor* policy achieves a good trade-off between traffic overhead and reduced operational complexity at the sender side. Therefore, its adoption is particularly suitable for large sessions where many musicians are involved, as its lower complexity and memory occupation at the transmitter side w.r.t. the *closed-loop* policy makes it more scalable. Its usage is also recommended for deployments with computationally-limited hardware (e.g., embedded processors). Moreover, the proposed policy offers higher flexibility, since the refresh rate k can be set and dynamically updated based on current network conditions to achieve the desired trade-off between transmission bitrates and fidelity of the reproduced MIDI stream.

[3]Note that a stable network, that is, a network exhibiting negligible packet loss probability, is considered a requirement for NMP. Anyway, higher loss probabilities have a small impact on the overhead of the *closed-loop* policy, especially with high RTTs. To prove that, we measured that $p = 0.2$ causes a +1.45% average bandwidth usage w.r.t. $p = 0.01$.

[4]It is worth reminding that the amount of participants impacts the overall data rates received/transmitted by a peer, which grows linearly with the number of peers, regardless of the PLC approach being implemented.

Chapter 8

Sparse Linear Prediction for Packet Loss Concealment in Networked Music Performances

Chapter 7 addressed packet loss concealment for symbolic MIDI streams through the recovery journal mechanism, which reconstructs the instrument state from redundant information embedded in successfully received packets. For sampled audio streams, however, a fundamentally different approach is required: rather than recovering discrete events, the algorithm must synthesize plausible waveform continuations to fill gaps in the audio signal. As the characterization of real network conditions provided by the DUST dataset (Chapter 6) confirms, packet losses do occur in NMP sessions, motivating the development of efficient concealment techniques for audio streams. This chapter presents a sparse linear prediction approach that exploits the periodic structure of many musical signals to achieve prediction quality comparable to traditional autoregressive models while substantially reducing computational cost. The algorithm's sub-millisecond execution time on ARM hardware makes it suitable for deployment on the Raspberry Pi-based MEVO client.

8.1 Introduction

Meeting the stringent latency requirements of NMP applications demands careful management of multiple factors, including packet transit time and buffering delays. Packet jitter and loss, being inherently unpredictable, can cause audio gaps when data is unavailable for playback. Receiver buffers can mitigate gaps due to late packets by absorbing timing variations, at the cost of added queuing delay, thus

leading to a trade-off between latency and packet loss rate. Moreover, packets discarded by intermediate routers remain unrecoverable, regardless of the receiver buffer size.

To mitigate this issue, various strategies can be employed. One of the most widely used being Packet Loss Concealment (PLC). PLC techniques aim to generate substitute audio samples to mask missing data in a way that is imperceptible to human listeners. While extensive research exists on PLC, most approaches have been tailored for speech applications such as VoIP or videoconferencing and thus hardly applicable in NMP scenarios.

To bridge such gap, we introduce an algorithm specifically designed for PLC in the context of NMP applications, addressing the unique challenges of real-time, high-fidelity audio performance over the Internet. The method consists of a Sparse Linear Prediction algorithm that selects a subset of previous noncontiguous lags, enabling modeling of periodic signals while maintaining computational efficiency suitable for real-time constraints. The algorithm relies on a sparse autoregressive model for signal reconstruction, ensuring smooth transitions through alignment-optimized cross-fading. Thanks to its simplicity, the algorithm is lightweight even when implemented on computationally-limited hardware: preliminary tests prove that it can fit the underlying data in some hundreds of microseconds on a Raspberry Pi 4, and predict each missing sample in some tens of nanoseconds on the same hardware.

The remainder of the chapter is organized as follows: Sec. 8.2 presents a brief literature review on PLC techniques, Sec. 8.3 describes the proposed algorithm, and Sec. 8.4 provides some implementation details. Preliminary results are provided in Sec. 8.5.

8.2 Related Work

8.2.1 Early Studies

Early packet loss recovery methods can be broadly categorized into sender-based and receiver-based strategies [179]. Sender-based approaches, such as Forward Error Correction (FEC) and packet retransmission, add data redundancy at the cost of increased bandwidth and latency [256]. Using sender-based approaches, packets are usually fully recovered. Conversely, receiver-based methods operate only on successfully received packets and consist in filling the missing audio portion with artificially-generated samples. PLC techniques fall under the latter category.

An historical approach proposed in the ITU-T G.711 standard [1, Appendix I], named pitch waveform replication, uses a pitch detection algorithm to identify periodic segments and replicates them during loss periods. However, it may produce artifacts when the signal characteristics change abruptly, or in the case of non-periodic signals.

8.2.2 Linear Prediction Based Methods

Linear Prediction (LP) has been widely adopted for PLC due to its ability to model the spectral envelope of speech signals efficiently. Gunduzhan and Momtahan [90] proposed an LP-based PLC algorithm for PCM-coded speech that extracts residual signals through linear prediction analysis and uses periodic replication for excitation generation. Kondo and Nakagawa [128] extended this approach by using bidirectional LP, i.e., predicting lost segments based on both preceding and succeeding packets. This enhancement improved reconstruction quality, particularly for longer loss bursts, achieving mean opinion scores similar to those obtained by [1, Appendix I] for packet loss rates up to 10%.

More recently, Ohidujjaman et al. [171] proposed to replace the traditional autocorrelation method with a modified covariance method for LP coefficient estimation, considering both the forward and backward prediction error in the optimization phase. This approach mitigates the ill-conditioning issues of traditional autocorrelation-based Autoregressive (AR) models.

In the context of NMP, Sacchetto et al. [200] proposed to implement LP leveraging the Burg model, due to its higher numerical stability in comparison to the more widely adopted Yule-Walker model. They demonstrated that AR models can effectively predict missing audio segments with computational requirements suitable for real-time processing, outperforming silence substitution and pattern replication methods commonly used in NMP systems.

The application of AR models to music signals presents unique challenges compared to speech, as music exhibits more complex harmonic structures and longer-term dependencies. The pitch-invariant properties of musical instruments make sparse representations particularly attractive for this domain.

8.2.3 Deep Learning Approaches

The advent of deep learning has opened new avenues for PLC. Mohamed et al. [160] provided a comprehensive survey of deep learning methods for speech PLC, highlighting the potential of generative models. Lee et al. [244] proposed a hybrid approach combining generative and predictive models, using a neural vocoder conditioned on features predicted by a separate model.

Deep learning methods typically require significant computational resources and introduce latency that may exceed the strict requirements of NMP applications. The ICASSP 2024 Audio Deep PLC Challenge [56] and the IEEE-IS2 2024 Music PLC Challenge [155] emphasized the importance of computational efficiency alongside reconstruction quality. Research on Deep PLC techniques has led to methods that perform well with a number of parameters in the order of 10^5, as in [156].

8.3 Proposed Method

PLC implementations for NMP applications should process audio frames within a time depending on the size of the de-jitter buffer. Such time is typically limited to a few ms, to meet the latency constraints imposed by NMP.

The objective of our proposed algorithm is to leverage the strengths of LP, which inherently supports low-latency operation, while addressing common limitations of AR approaches in long-term predictions. An AR model is a statistical framework that forecasts future values in a sequence based on its past observations. In this sense, the proposed method remains an AR model; however, it introduces two key enhancements:

- enforcing sparsity by using a limited number of parameters, so that predictions rely only on the most informative past values;
- mitigating error accumulation by excluding short-term predictors, which are more prone to compounding prediction errors.

8.3.1 Problem Formulation

Given a time-discrete signal $s[t]$, we seek to model it as:

$$s[t] = \sum_{i=1}^{K} \phi_i s[t - d_i] + \epsilon[t] \tag{8.1}$$

where $\{d_i\}_{i=1}^{K}$ are sparse lags and $\{\phi_i\}_{i=1}^{K}$ are the corresponding coefficients, with $\epsilon[t]$ being an error term. The key idea is the iterative selection of lags based on their correlation with the prediction residual, achieved through heuristics aimed at keeping processing times low while ensuring accurate enough predictions. This method is equivalent to an Orthogonal Matching Pursuit (OMP) where the dictionary of atoms corresponds to the lagged versions of the signal itself.

Sparse Representation and OMP

OMP, introduced by Pati et al. [174], has found applications in various signal processing domains. In audio processing, sparse representations have been particularly successful for source separation and compression tasks [207]. OMP evolves from matching pursuit with the key advantage of the re-optimization of all selected coefficients at each iteration, leading to better approximation quality at the cost of increased computational complexity.

Furthermore, under the assumption that the audio signal is weakly stationary, OMP equations can be expressed in terms of autocorrelation values, avoiding the issue of dealing with a potentially large design matrix.

Method Description

Given the signal $s[t]$, we define:

- $r[d]$ as the autocorrelation value of s at lag d,
- ϕ_d as the coefficient of the model relative to lag d,
- $\tilde{s}[t] = \sum_i \phi_i s[t - d_i]$ as the predicted version of the signal,
- $\epsilon[t] = s[t] - \tilde{s}[t]$ as the error signal,
- $\rho_{\epsilon,s}[d]$ as the cross-correlation between the error signal and the signal itself.

At each iteration, we try to select an informative lagged version of the signal $s[t-d_i]$ by choosing d_i such that:

$$d_i = \arg\max_d \rho_{e,s}[d] \tag{8.2}$$

We initialize the model by considering $\tilde{s}[t] = 0$, thus $\epsilon[t] = s[t]$, and $\rho_{e,s}[d] = r[d]$. The assumption of stationarity permits to simplify computations. The ordinary least squares (OLS) optimizer for simple linear regression is expressed as per Eq. 8.3:

$$\boldsymbol{\phi} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y} \tag{8.3}$$

where $\boldsymbol{X}$ is the design matrix, with each row being a sample of the input regressors space, $\boldsymbol{y}$ is the array of scalar responses, and $\boldsymbol{\phi}$ is the array of parameters. There exist a trivial parallelism between Eq. 8.3 and a modified version of the Yule-Walker equation:

$$\boldsymbol{\phi} = \boldsymbol{R}^{-1} \boldsymbol{r} \tag{8.4}$$

In Eq. 8.4, $\boldsymbol{R}$ is a matrix built such that $\boldsymbol{R}_{i,j} = r(|i - j|)$ and $\boldsymbol{r}$ is a column vector such that $\boldsymbol{r}_i = r(i)$. It can be proven that the two formulations are equivalent for weakly stationary signals. In our case, introducing sparsity, we define $\boldsymbol{R}$ such that $\boldsymbol{R}_{i,j} = r(|d_i - d_j|)$ and $\boldsymbol{r}_i = r(d_i)$. Under the same assumption, $\rho_{\epsilon,s}$ can be computed without the need of explicitly calculating $\epsilon[t]$, as it can be easily expressed in terms of autocorrelations by means of Eq. 8.5:

$$\rho_{\epsilon,s}[d] = r[d] - \sum_i \phi_i r[|d - d_i|] \tag{8.5}$$

These findings enable efficient iteration through the OMP algorithm, where each step involves lag selection (equations 8.2 and 8.5) followed by model replacement (equation 8.4), with the model size increasing by one at each iteration. Algorithm 1 presents a simple implementation of the proposed method.

Algorithm 1 OMP-based Sparse AR

1: **Input:** array of last w samples x, max iterations K, max lag L
2: Initialize sets of lags and coefficients: $\mathcal{D} = \emptyset$, $\phi = \emptyset$
3: Build array of autocorrelations $\mathbf{c}$ of length L (up to lag $L-1$)
4: **for** $j = 1$ to K **do**
5: Compute the L-length array $\boldsymbol{\rho}$ according to Eq. 8.5
6: Find $d_j = constrained\_argmax(\boldsymbol{\rho})$
7: **if** stopping criterion met **then**
8: **break**
9: **end if**
10: Update set of lags: $\mathcal{D} = \mathcal{D} \cup \{d_j\}$
11: Construct $\mathbf{R}_{ij} = \mathbf{c}[|d_i - d_j|]$
12: Construct $\mathbf{r}_i = \mathbf{c}[d_i]$
13: Solve $\mathbf{R}\boldsymbol{\phi} = \mathbf{r}$
14: **end for**
15: **Output:** Lags $\mathcal{D}$, coefficients $\boldsymbol{\phi}$

8.4 Method Analysis and Implementation Considerations

8.4.1 Model Fitting Constraints and Suggestions

The model inputs are: (*i*) an array of samples, whose size can be fixed or dynamically determined based on signal conditions; (*ii*) the maximum number of iterations, which determines the maximum number of parameters in the model; and (*iii*) the highest lag to consider, which determines the number of autocorrelations to compute.

An effective implementation should incorporate several criteria and hyperparameters to mitigate the possible drawbacks of the model as-it-is:

1. The number of recent samples to be considered for model fitting should be limited to a portion of the signal that exhibits stationary properties. Events such as note onsets may alter the local properties of the signal, and signal portions immediately before or after such events should be excluded.

2. The *constrained_argmax* function (line 6 of Alg. 1) can replace the standard argmax with additional policies, such as:

   - Excluding smaller lags to reduce error propagation by preventing predicted values from serving as predictors in subsequent predictions.
   - Excluding or penalizing lags that are too close to those already in $\mathcal{D}$.

- Stopping if the correlation associated to the last selected lag is too high or too low.

3. When constructing $\boldsymbol{R}$, regularization can be applied. For instance, Tikhonov regularization can be implemented by solving the system using $\boldsymbol{R}' = \boldsymbol{R} + \lambda \boldsymbol{I}$ instead of $\boldsymbol{R}$ at line 13. The computed coefficients can be further refined using more complex objective functions (e.g., LASSO regularization [229]), which may be effective for larger values of K.

4. The algorithm may terminate before K iterations, based on appropriate stopping criteria 7.

8.4.2 Continuity at Boundaries

The proposed method can predict multiple missing audio samples without depending on previously predicted ones. However, since the first predicted samples are independent of the last samples in the input window, discontinuities may appear where prediction starts, potentially causing audible glitches that propagate to successive portions of generated audio. These issues can be avoided by either predicting the first samples with a model that guarantees continuity, or using the proposed method to predict the few last present samples. In either case, two versions of the same audio portion are available, enabling cross-fading to mitigate discontinuity. The same approach can be applied at the end of the predicted audio portion, where real samples become available again.

Cross-fading can be used in fixed-sized regions at the predicted section boundaries. However, it is advisable to identify sub-regions that maximize similarity between overlapping portions and apply cross-fading there, following the overlap-add procedure of the WSOLA algorithm [247].

8.4.3 Stability

Without any regularization, the output of the model is often unstable. Taking the z-transform of Eq. 8.1, where $S(z) = Z\{s[t]\}$ and $E(z) = Z\{\epsilon[t]\}$ we obtain:

$$S(z) = \sum_{i=1}^{K} \phi_i z^{-d_i} S(z) + E(z) \tag{8.6}$$

which can be rewritten as:

$$S(z)[1 - \sum_{i=1}^{K} \phi_i z^{-d_i}] = E(z) \tag{8.7}$$

Therefore:

$$\begin{aligned} S(z) &= E(z)/\Phi(z) \\ \text{where } \Phi(z) &= 1 - \sum_{i=1}^{K} \phi_i z^{-d_i} \end{aligned} \tag{8.8}$$

This represents an all-pole infinite impulse response filter. If any pole of $1/\Phi(z)$ lies outside the unit circle, the model is unstable. Empirically, low-order models seem less prone to exhibit early instability in their predictions. In any case, the prediction output should be monitored for unstable resonances or exponential growth.

8.4.4 Relationship to Waveform Replication Method

The stopping criteria at line 7 can be implemented according to several strategies. When it is based on high correlation between predicted and original signals, strongly periodic signals (e.g., sustained notes) may cause the algorithm to terminate with a single lag. Setting the associated coefficient to 1 maintains constant signal power, making the method equivalent to pitch waveform replication [1].

8.4.5 Computational Complexity

The algorithm has been specifically designed to have a very low complexity. Once the autocorrelation values are computed, no more operations are performed on the historical samples. Additionally, the number of selected lags, which is directly linked to the number of iterations of the algorithm, is usually very low. The overall complexity of the model fit phase can be computed as follows:

- **Autocorrelation computation:** its complexity is $\Theta(Nlog(N))$ using a Fast Fourier Transform (FFT) - based approach.
- **Cross-correlation computation:** Each iteration k requires $(k-1)L$ operations to compute the error-signal cross-correlation array, resulting in an overall complexity of $O(K^2L)$. The additional lag selection phase is a linear search on the array ($O(L)$), which makes it negligible w.r.t. the array computation.
- **System solution:** Solving each linear system has a cubic complexity, thus iteration k requires k^3 operations. Note that, contrarily to AR models, the $\boldsymbol{R}$ matrix is symmetric, but it is not Toeplitz, so, unfortunately, the Levinson-Durbin recursion is not applicable. Overall, this phase has a $O(K^4)$ complexity.

It follows that the total complexity is $O(K^2L + K^4 + Nlog(N))$. With K small and constant and $L \approx N$, this reduces to $O(Nlog(N))$, dominated by the FFT-based autocorrelation computation.

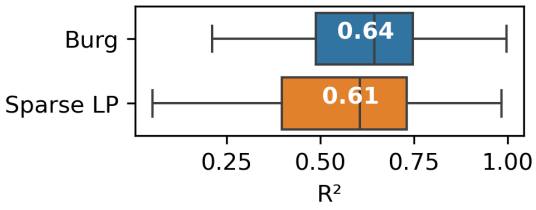


Figure 8.1: R^2 comparison between Burg-based AR (model order $p = 128$) and the proposed Sparse LP method

8.5 Results

The proposed method has been implemented in C++. FFT operations and linear algebra operations are performed respectively via the FFTW [80] and the Eigen3 [89] libraries. Furthermore, the code is executed under FIFO scheduling on a PREEMPT_RT Linux kernel to better reflect its impact on a NMP application. Results are obtained by running the implementation on a Raspberry Pi 4, on the same dataset (29 files with heterogeneous musical content, each of them corrupted by 100 lost portions of 128 samples each, randomly inserted with uniform distribution) of our baseline (i.e., the Burg-based AR model implemented in [196][1]). The prediction quality is assessed in terms of the coefficient of determination R^2 and reported in Fig. 8.1, which highlights that the two methods achieve comparable results.

The execution time does not exhibit a strong dependency on model order, which has been limited to 3. In the proposed method, continuity at the left boundary of the gap is guaranteed by a lightweight Burg AR model (with order $p \leq 8$), implemented based on [255] and exploiting the autocorrelation values already obtained with the FFT method. Samples generated by the Burg AR model are cross-faded with the LP-predicted ones. The median time to fit the whole model based on a history of 2048 samples is 256 μs, the minimum 234 μs, the maximum 571 μs and first and third quartiles are respectively 244 μs and 289 μs. This reduces by a factor of three the time required to fit the Burg method with hybrid denominator optimization, used as baseline, which requires $\approx$ 807 μs on the same hardware. Prediction time is between some tens and some hundreds of nanoseconds per sample, i.e., at least one order of magnitude lower than the playout time per sample, making it negligible in the majority of real application scenarios.

[1]Dataset and code are available at https://github.com/matteosacchetto/burg-implementation-experiments

Chapter 9

Towards an Inclusive Framework for Remote Musical Education and Practices

The preceding chapters focused on technical performance (latency, audio quality, loss resilience) that benefits all users. This chapter shifts focus to users who face additional barriers to musical participation. Musicians with visual, auditory, or mobility impairments encounter challenges that standard NMP systems do not address. This chapter analyzes the space of technological possibilities and identifies integration challenges for truly inclusive remote music education and performance.

9.1 Introduction

Information and Communication Technologies (ICT) and Artificial Intelligence (AI) are introducing dramatic changes in the way pedagogical processes are conceived: distance education and computer-based learning have started proliferating in the last decade and are expected to further expand [49], fostered by novel, interactive and immersive approaches enabled by technologies such as Internet of Things (IoT), Virtual and Extended reality (VR/XR), and the Metaverse [205]. This implies a global rethinking of the fruition modalities for educational services.

There are several categories of subjects who would benefit from the long-term adoption of distance learning frameworks. Among those, students with visual/ auditory/motor impairments or special needs, for whom traditional in-presence access to education is challenging or even precluded, represent a non-negligible share. According to [165], in 2020/21 15% of US public school students received special

education services, for a total of 7.2 millions. Such amount comprises more than 1 million students diagnosed with health impairments resulting in limited strength, vitality, or alertness, around 700 thousands students with hearing impairments, and around 350 thousands students with orthopedic/visual impairments or deaf-blindness. The same source reports that 1% of students with special educational needs are homebound or confined in hospitals or separate residential facilities.

As of today, remote teaching heavily relies on videoconferencing technologies, which are mainly designed to maximize speech intelligibility while reducing the required transmission bit-rate, resulting in the adoption of low sampling frequencies and highly efficient compression codecs, at the price of causing voice signal distortions. Differently from videoconferencing, NMP scenarios impose specific requirements in terms of latency, reliability, and audio processing capabilities, thus precluding the applicability of off-the-shelf videoconferencing technologies for remote musical education. In such applications, audio codecs would worsen the perceived end-to-end delay and reduce the quality of the streamed audio signal due to their compressive effects. Moreover, an adequate backend infrastructure to support low-latency and high-fidelity audio mixing would be required. The aforementioned limitations clearly emerge in [64], which highlights the difficulties encountered by music schools in delivering group lessons and supporting rehearsal sessions using commercial e-learning technologies during the first lockdown enforced to mitigate the spreading of the SARS-CoV-2 pandemic.

Motivated by the fact that the above-mentioned shortcomings heavily hinder the widespread adoption of distance education of music-related subjects, and consequently their adoption by subjects with disabilities, the aim of this work is threefold:

- offering a comprehensive overview of the educational and performative needs experienced by disabled users in the field of remote musical teaching and practices;
- identifying the technological solutions that could address such needs;
- discussing their integration in existing NMP frameworks to enhance the inclusiveness of their adoption, highlighting open challenges and future research directions.

The rest of the chapter is organized as follows: Section 9.2 reviews accessibility and interaction requirements to enable fruition of remote musical education and practices by several categories of disabled users. It also discusses the main technological solutions for the realization of accessible human-machine interfaces tailored to musical performances. Section 9.3 overviews existing NMP applications and compares their design choices in terms of implementation and infrastructure. Section 9.4 envisions an inclusive NMP framework and details its main building blocks. Some illustrative use cases are presented in Section 9.5. Open technical challenges and

future research directions involved in its realization are discussed in Sections 9.6 and 9.7, respectively.

9.2 Accessible Human-Machine Interfaces for Musical Education

9.2.1 Educational and Performative Issues Experienced by Disabled Users in Music-Related Fields

Research on how to accommodate the needs of students with disabilities in the field of music education started more than four decades ago, as testified by two surveys, respectively, appeared in 2007 [120] and 2015 [123]. Such studies triggered further investigations on how music teaching can embrace inclusiveness and diversity [130]. In Table 9.1, we report some of the most widely recognized barriers that preclude or strongly limit the inclusion in musical education of subjects with various types of disabilities, as well as a number of resources available to educators to make musical practices accessible to students with special needs.

In the following subsections, we review the main technological solutions already adopted for in-presence inclusive musical education. In particular, in the past two decades the research areas referred to as "Musical Extended Reality" (Musical XR) [236] and Internet of the Musical Things (IoMusT) [241] have started being investigated in the scientific community. The former introduced a paradigm shift by disrupting traditional notions on musical interaction and enabling performers and audiences to interact musically with virtual/augmented objects, agents, and environments [22], also in the context of music education [211]. Conversely, the latter identifies computational devices integrated into physical objects (e.g., smart musical instruments or wearables with music-related purposes) devoted to generating or receiving musical content, interconnected via a telecommunication infrastructure. Such smart musical objects can be leveraged also for educational purposes. Finally, we complete the section by reviewing some literature-reported use-cases that testify the adoption of the presented technologies in inclusive music education trials.

9.2.2 Haptic Feedback Techniques

A number of studies have investigated the conveyance of musical information through the sense of touch (see [186, 75] for a thorough overview). Tactile rendering setups normally focus on capturing a single music feature, e.g., rhythm, pitch, melody, timbre, and loudness.

The usage of musical haptic wearables to improve communication among performers sharing the same physical environment has been envisioned in [234]. The adoption of haptic feedbacks to enrich the emotive experience while listening to

Table 9.1: Overview of Educational Issues of Disabled Music Students

Type of Impairment	**Issues**	**Countermeasures**
Blind and Visually Impaired (BVI) students	Being music notation predominantly visual, it cannot be written, read or directly reproduced by the majority of BVI musicians, which heavily hinders their capability of conveying their creative intentions [176, 248, 203];	Availability of braille scores [3], modified musical instruments [79] and accessible music production interfaces [78].
Deaf and Hard of Hearing (DHH) students	DHH subjects experience difficulties in fine pitch discrimination, which often precludes practicing with instruments such as violin, cello or brass instruments, all of which require very precise intonation. Moreover, DHH musicians rely on visual cues to compensate for their reduced hearing capabilities, which may constrain their physical placement when playing in ensembles, bands, and orchestras [98].	Choice of instruments which easily convey vibrations to DHH players (e.g., guitar, electric bass). Strategic pairing of non-DHH to DHH performers to ensure dedicated guidance/cues [98]. Availability of modified musical instruments [79].
Mobility impaired students	Subjects with motor impairments often cannot play traditional musical instruments or use music production software due to lack of accessibility [39].	Wide offer of adapted musical repertoire (e.g., one-handed piano pieces) [3]. Availability of modified musical instruments [3].

music has also been experimented in [101, 38]. Furthermore, systems that transpose gestural cues into haptic feedbacks have proven to be useful to enrich musical perception and interplay for Blind and Visually Impaired (BVI) and Deaf and Hard

of Hearing (DHH) performers [88].

Notably, the sensitivity of the tactile system is quantified in [74] in terms of frequency, intensity and temporal features. The study reports that, through haptic stimulation at the arm, wrist, or hand, approximately 40 different frequency steps can be discriminated. Moreover, amplitude modulation sensitivity was found to be highest when using a carrier tone at 250 Hz. Finally, audio and haptic signals are perceived as simultaneous if the discrepancy between the haptic and audio signal onsets is within 25 ms.

In terms of bit-rate requirements, tactile feedback ranges from tens of to hundreds of kbps, depending on the type of stimuli being involved.

9.2.3 Motion Tracking Techniques

In the context of music education, several studies have already demonstrated the benefits of the usage of computer technologies for BVI people [102]. However, no scientific study has yet tackled the integration of motion tracking or haptic feedback functionalities in NMP systems to accommodate the special needs of disabled users. The exploitation of Kinect cameras for telerehabilitation has been investigated in [11] using a WebRTC-based communication protocol, though without integration of audio streaming functionalities. Kinect cameras for motion data acquisition have already been adopted to analyze gestures of music performers [92, 93].

Notably, from a networking standpoint, streaming pose data for avatar rendering rather than video frames significantly reduces transmission bit-rate requirements, as it only needs to convey the Cartesian position and rotation data of the represented joints. This results in savings of at least one order of magnitude compared to encoded video streaming, and potentially up to three orders of magnitude for unencoded video. Regarding motion-to-photon latency (i.e., the time it takes for a user's gesture to produce a visual change on a display), experimental values are below 40 ms [257], which is comparable to the delays introduced by a camera streaming compressed video frames. This would foster scalability in large deployments involving a high number of participants, (e.g., choirs or orchestras), bypassing the need to convey a dedicated video stream to each of them.

9.2.4 Music Visualization Techniques

Visualization can also be exploited to present music content to subjects with auditory impairments: within the rich corpus of scientific literature dedicated to such a topic (see [142] for a comprehensive survey), a few studies have specifically targeted the DHH community. In paper [125], sounds generated by a violin are digitized and associated with movements of objects such as flowers and plants, with the goal of achieving interactive sound visualization to support music education for children with hearing impairments.

Paper [55] presents ViTune, a music visualization prototype that enhances the musical experiences of the DHH community thanks to the usage of an on-screen visualizer generating effects alongside music.

Paper [152] introduces BufferBeats, a toolkit for creating multimodal experiences for streaming services, aimed at improving accessibility to online music streaming by DHH subjects. The study is motivated by the presence of Digital Rights Management anti-piracy encryption, which often restricts the access to audio data which is required to create multimodal experiences for music streaming.

To the best of our knowledge, music visualization techniques have always been adopted in local settings or applied to pre-recorded streamed audio material only, whereas their integration in NMP frameworks for real-time musical practices has not yet been reported in the literature. However, since such techniques typically rely on local processing of audio content, from a networking perspective they do not require transmission of other data than audio, thus not incrementing the transmission burden of NMP systems.

9.2.5 Immersive Audio Rendering

Immersive audio is rendered by means of sound diffusion systems aimed at offering to the listener a three-dimensional (3D) representation of acoustic sources, so that they can be perceived as coming from a specific spatial direction. For the specific use-case of NMP applications, the usage of headphones is predominant with respect to surround sound systems, thanks to their ease of portability and configuration, and lower cost. In such a scenario, binaural audio rendering is typically the adopted immersive audio solution. It relies on the rendering of acoustic cues such as interaural time/level differences and acoustic filtering (i.e., the spectral information that depends on the characteristics of the user's physical attributes such as the shape of ears, head, shoulders, torso). This rendering is achieved via Head-Related Transfer Functions (HRTFs), which convey the directionality of a sound source to the listener's eardrum [178].

Spatial audio technology has already proved to enhance the sense of virtual imagery and spatialization for BVI users [77, 69, 6, 24]. In the context of musical practices, 3D audio rendering can be leveraged to recreate a shared acoustical scene at the premises of every remote participant. Therefore, it is expected to facilitate the participation to ensemble performances of BVI subjects, who could take advantage of the spatialized placement of sound sources to enhance their perception of the auditory cues necessary to synchronize their playing with that of the other performers. Similarly, DHH users could benefit of a personalized design of the acoustic scene rendered at their premises, tailored to their specific auditory impairment.

Though a few examples of immersive and multimodal environments specifically designed to permit fruition also by subjects with disabilities exist (see, e.g., [12]), no

attempt to integrate immersive audio rendering systems in inclusive NMP frameworks has yet been reported in the literature. However, immersive audio settings for NMPs have started being investigated and assessed [31, 239, 238, 109].

Concerning bit-rate requirements, they depend on the desired sampling rate and amount of bits per sample, as well as on the number of audio channels to be supported. For example, an uncompressed mono audio stream at 48 kHz using 16 bits resolution accounts for 768 kbps. Such amount scales linearly with the number of audio channels. It follows that scenarios involving multiple participants and complex spatialized audio renderings may impose a significant burden in terms of data volumes to be transferred, potentially non compatible with up/downstream bit-rates supported by typical residential connectivity.

9.2.6 Exemplary Field Trials

In in-presence music teaching, vibrotactile feedbacks have been already demonstrated to convey the concept of rhythm and of the spectrum of sound frequencies to BVI and DHH students [83, 53]. Moreover, motion tracking technologies in connection with sound and visual production have been experimented to ease musical expressions of subjects with mobility or cognitive impairments [149]. The potentialities connected to the adoption of virtual and augmented reality technologies for music teaching have been investigated in several studies, especially for primary education [151], focusing on mobile settings [110], and even including disabled students [211]. Notably, the combination of passive haptics with virtual reality has proved useful for music conduction education [17].

Focusing instead on on-line music education, several pilot studies conducted with traditional videoconferencing systems have been documented (see, e.g., [126]), especially during the Sars-CoV-2 pandemic period [99, 33]. However, as emerged in [170], sustainable technology integration in music classrooms is still far from being achieved at large scale, mainly due to barrier such as lack of digital literacy, non-affordability of equipment costs and scarce alignment with the instructors' pedagogical praxis. This outcome is in line with the general trend observed in e-learning-based education, as testified by numerous studies on its technological acceptance in various geographical areas (see, e.g., [173, 175, 226, 204]). Another issue is the economical sustainability of ICT-based educational solutions, for which dedicated pricing models and cost-benefits analyses must be devised, as recently investigated in [168, 96, 213].

9.3 NMP Applications and Backend Infrastructures

9.3.1 Main Functionalities of NMP Systems

The core functionality of NMP systems is the support of professional-quality real-time audio streaming with low mouth-to-ear latency, as discussed in Section 1.1. Exceeding these latency thresholds induces a tendency towards tempo deceleration, thus making the maintenance of a stable tempo more challenging [191].

Additional features typically integrated in NMP systems are:

- local audio manipulation capabilities (e.g., recording, reproduction of pre-recorded streams, adjustment of volumes, equalization levels and panning);
- local digital audio workstations;
- communication and interaction interfaces such as chats, user status tracking etc.;
- collaborative music composition/editing/production tools.

For what concerns video support, only a few NMP solutions integrate video streaming capabilities. Unfortunately, typical video acquisition and encoding/decoding delays significantly exceed the 30 ms latency threshold, which is incompatible with remote musical performances. Therefore, the offered video-stream is usually desynchronized w.r.t. the corresponding audio-stream, thus heavily hindering reliance on visual cues during real-time remote musical interactions [30]. Alternatively, video is streamed without being encoded (as in [185, 58]), which requires an adequate telecommunication infrastructure capable of supporting an uplink/downlink bit-rate of a few Gbps.

9.3.2 Peer-to-Peer vs. Client-Server Architecture

Depending on the implementation choices and on the scalability needs of the users, either client-server or peer-to-peer architectural solutions are adopted. Typically, peer-to-peer architectures offer lower mouth-to-ear latency than client-server ones, but can accommodate only a few musicians in the same session. Conversely, for larger deployments client-server approaches are preferred, at the price of added latency caused by the server operating as intermediate relay node. The scalability limitations introduced by peer-to-peer architectures lie in the fact that, in a session with N participants, each one sends and receives $N-1$ audio streams. If the audio stream consists of uncompressed Pulse Code Modulated (PCM) stereo audio signals sampled at either 44.1 or 48 kHz, the required uplink/downlink bitrate is around 1.5 Mbps per participant. More sophisticated configurations, involving more than

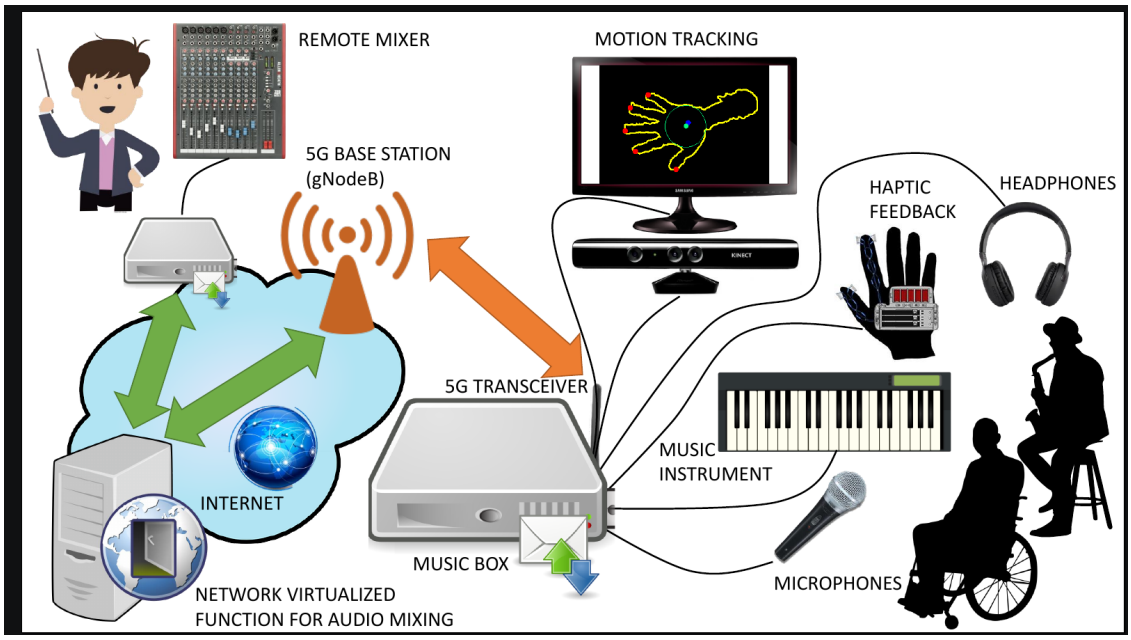


Figure 9.1: The proposed inclusive NMP framework for remote music education and practice

two channels per user, further increase such figures. It follows that the uplink bitrate offered to residential customers by Internet Service Providers (ISPs) may be insufficient for peer-to-peer NMP sessions involving a significant amount of participants. In such cases, a client-server configuration with a server node equipped with audio synchronization and mixing capabilities is typically leveraged. The server collects the audio streams generated by every musician in upstream, mixes them at the server's premises and produces a single stereo signal to be delivered in downstream, thus reducing bit-rate requirements per participant.

9.4 Towards an Inclusive Framework for NMP and Remote Musical Education

Though a number of NMP solutions already exist (see Section 2.5 for a comprehensive review), they show several limitations:

- Lack of accessibility for users with visual/auditory/mobility impairments.
- Lack of user-friendliness for subjects without technical background in communication technology (e.g., children): for instance, users may be requested to manually input IP addresses or UDP port numbers to enable multimedia streaming.

- Lack of portability and deployment flexibility: NMP solutions leverage wired networking infrastructures for the network access segment, due to the unacceptable latency introduced by the medium access control mechanisms of Wi-Fi and 4G technologies. The adoption of 5G connectivity for NMP practices has started being investigated very recently (see, e.g., [235]) but not yet fully integrated in NMP frameworks.
- Lack of a dedicated backend to support low-latency audio streaming, i.e., they treat latency contributions introduced by the telecommunication network as a "black box", with no effort to minimize it to improve the perceived Quality of Service (QoS) and Quality of Experience (QoE).

To address the aforementioned shortcomings, our envisioned inclusive NMP platform for remote music teaching and practices (depicted in Fig. 9.1) adopts a client-server architectural paradigm. It envisions the possibility of leveraging a dedicated easy-to-use hardware (music box), with the goal of overcoming the real-time processing limitations of desktop computers and operating systems. The music box can leverage a single-board computer (e.g., a Raspberry-Pi, as in [41]), or even customizable hardware (e.g., a Field Programmable Gate Array board) to support the technologies described in the following.

Interface control system based on a camera feedback and computer vision, or on motion sensors Enabling mobility-impaired users to provide a stream of control information to interact with the interfaces of the NMP platform and with remote performers requires wearable sensors and/or hand-tracking technologies. This represents a unique way for those users to participate in music performance and collaboration, overcoming barriers that prevent their involvement in traditional music-making settings.

User inputs can be captured by sensors tailored to the specific mobility abilities of the subject, such as:

- video-based hand pose reconstruction methods [45], or specialized unobtrusive sensors [85] to reconstruct small-scale hand and finger movements;
- specialized joysticks or buttons that can be actuated with minimal effort and range of motion using a hand or the chin;
- gaze tracking devices [48];
- sip-and-puff devices [52].

This would enable a mobility-impaired user to participate in music-making as a conductor or instructor, by providing a real-time stream of information to an ensemble of performers. The same input methodology can be used to allow the user to act as a performer [118, 221, 228], by controlling a synthetic instrument either by directly

determining the notes being played, or by shaping some aspects of the sound of an instrument that operates, at least in part, automatically (e.g., a sequencer playing a predetermined melody or an arpeggiator).

Gestural data streams acquired by means of the previously described motion tracking techniques can be rendered at the remote counterparts either via realistic avatars or by using stylized, simplified representations that still convey the user's intention. For example, the avatar could reproduce the pose of a conductor/instructor whose movements match the inputs of the user. In addition to enhancing usability, this brings a significant advantage in terms of bit-rate requirements, since gestural data can be encoded much more efficiently in terms of data volume and with lower latency than video data [162], thus fostering scalability while meeting the strict latency requirements of NMP systems.

Sensors and actuators for remote conveyance of vibro-tactile signals Haptic feedback mechanisms (possibly in combination with a touchscreen interface) can be adopted to ease perception for DHH and BVI users. Indeed, visually/auditory-impaired subjects heavily rely on haptic sensations to interact with the surrounding environment [129, 74]. Tactile wearable actuators [47] can also be used in combination with motion tracking systems (see previous paragraph) to convey haptic information regarding the gestures performed by the remote counterpart(s) to BVI/DHH users, e.g., those of a remote instructor or conductor.

Accessible graphical interface The graphical user interface of the NMP system should be adherent to the accessibility standards for web applications [50] (e.g., compatibility with screen readers), possibly enhanced with tactile feedbacks [202] (e.g., by means of a touchscreen interface with customizable electroadhesive characteristics to render various textures in different areas of the screen, such as the device adopted in [198]).

Remote control of mixing functionalities Sound technicians should be enabled to remotely synchronize and mix the audio tracks received by all the participants to a NMP session, adjust their listening volumes and/or add audio effects such as equalization and reverb, to create personalized mixes for each performer. This would foster inclusion of subjects with reduced control capabilities of local interfaces due, e.g., to physical impairments.

Visors for Integration of Augmented/Virtual Reality (AR/VR) Environments with Immersive Audio Rendering The integration of spatialized audio rendering in the envisioned NMP framework is fundamental for conveying to the performance the sense of sharing a single acoustical scene [109]. Moreover, the usage of personalized HRTFs can help DHH users to compensate partial auditory

deficiencies, with a moderate increase of bit-rate requirements. AR/VR environments can also be coupled to 3D audio rendering to further enhance the perception of being part of a shared musical environment [146].

5G access network connectivity To interconnect the user's front-end to a dedicated backend infrastructure, 5G offers an unprecedented level of flexibility to fulfill service-specific requirements in terms of bitrate, latency, and reliability. 5G/beyond-5G wireless transmission technologies promise to guarantee access delay to the backbone telecommunication infrastructure below 10 ms, and even as low as 1 ms for mission-critical applications [210], which matches the latency requirements of NMP [59].

Additionally, flexibility of NMP deployments would be greatly enhanced by the adoption of 5G technologies. Indeed, the usage of cabled Ethernet connections for local area network connectivity heavily constrains portability, whereas Wi-Fi wireless connections are typically avoided in NMP setups, as they introduce excessively high jitter [237].

Finally, 5G networks are considered to be enablers of the so-called "tactile Internet" [210], which entails ultra-low latency transmission of haptic signals. Therefore, 5G/beyond-5G technologies constitute a promising candidate to accommodate the specific requirements of inclusive NMP systems. However, to date only a few designs of 5G infrastructures for NMPs have been investigated [40] along with a paucity of testbed deployments and in-depth statistical analysis on their latency and reliability performances [59]. Usage of 6G technologies is even more embryonal, though speculated by recent studies [161, 14].

At present, the features described in the above paragraphs have never been jointly designed and implemented in an NMP framework. Their potential usage and expected benefits are summarized in Table 9.2.

9.5 Use Cases

In this section, three illustrative use-cases for the framework presented in Section 9.4 are discussed. All the inclusive features presented in the following are implemented in MEVO, an NMP system currently being developed in Politecnico di Torino and first demonstrated in June 2023 in a distributed concert involving musicians in Turin (Italy) and Wrocław (Poland) [217].

9.5.1 Accessible Graphical Interface

An accessible Graphical User Interface (GUI) is the first step towards an inclusive NMP system. Though several guidelines regarding how to implement accessible

Table 9.2: Inclusive Features of the Proposed NMP Framework and Related Benefits

Feature	Usage	Expected Benefits
Accessible graphical interface enhanced with haptic feedbacks	Speech/gesture-enabled navigation of the system interface, haptic presentation of visual/audio content	Improve autonomous interaction with the NMP system of DVI/mobility-impaired users; complement the sensory perception of DHH users with audio visualization techniques enhanced with tactile feedbacks; offer haptic-enhanced score representation modalities to DVI users.
Motion tracking technologies	Acquisition of performers' gestures	Enabling of local control of virtual musical instruments (for motor-impaired and DVI users), conveyance of local gestures to remote counterparts and visualization via avatars without need of real-time video streaming.
Actuators for haptic signals	Remote conveyance of auditory or gestural cues	Support non-verbal interaction modalities with remote users, e.g., the tactile conveyance of the metronome beat (for DHH users) or of the conductor's gestures (for DVI users).
Remote control of mixing functionalities	Full operational integration of remote audio engineers/technicians	Availability of remote support in case of reduced control capabilities of local interfaces by disabled users (or by users without advanced technological background, e.g., children).
3D audio rendering and visors for AR/VR	Navigation of virtual spaces	Enabling 3D audio environment rendering and 3D visualization of virtual music instruments (for DHH and mobility-impaired users), possibility of customizing AR/VR spaces and the acoustic scene for educational and/or performative activities specifically tailored for the individual needs of the users.
5G connectivity	Support of low-latency, high-reliability wireless Internet access	Increased flexibility of deployment at the user's premises, overcoming physical barriers that may hinder accessibility to the system by disabled subjects.

Web GUIs[1] are available, creating a fully accessible GUI requires considering several aspects. First of all, it should be easily navigable by a broad spectrum of

[1]https://www.w3.org/WAI/standards-guidelines/wcag/

users, including BVI and DHH subjects. To facilitate the navigation with assistive technologies such as screen readers, the GUI should be organized in a hierarchical structure that is logically coherent. This allows screen readers to parse and present the content in a well-defined structure, with the right level of importance to the various graphical elements, and allowing to skip recurring elements, such as navigation elements, to go directly to the main content of a page. Additionally, it must support keyboard-based interactions and provide meaningful descriptions for each element. Moreover, the GUI should be designed considering font sizes and colors contrast, to be distinguishable by subjects with color blindness or Color Vision Deficiency (CVD). Furthermore, it should adopt a terminology and implement/design interactions that ensure that any user is able to navigate and understand it, independently of his/her technological and educational background. All the above-mentioned aspects have been taken into account for the development of the GUI prototype of MEVO.

More specifically, MEVO's GUI allows the user to perform all the necessary actions to configure the system, create virtual rooms where users meet to play together, and control additional functionalities, such as remote mixing.

All the UI components on the web page are organized in a hierarchical structure, and each component was created following the web accessibility guidelines for UI components[2]. Following those guidelines ensures seamless integration of the GUI with assistive technologies, such as screen readers. We also made sure that color contrast was appropriate[3] for the various font sizes, to ensure readability by subjects with CVD.

The GUI presents a left sidebar, used for switching the various pages. The two main pages are: "Rooms" (to create virtual rooms) and "Settings" (to configure the various system settings). Furthermore, the room creation page allows for creating rooms with either selected participants or with a shareable link that anyone can use to join.

The creation of virtual rooms and the connection setup is handled and presented by the platform in a fashion similar to traditional videoconferencing applications. Moreover, any additional functionality to manipulate the audio stream, such as audio mixing, is presented with a layout that recalls the one adopted by traditional audio editing applications.

Once the users are connected to the same virtual room, each of them can individually customize the audio mixing of the various audio streams it receives from the other peers. If users are not able (or willing) to perform this action by themselves, it is possible to allow another participant (e.g., an assistant/audio engineer) to remotely control the audio mixing of each user. From a technical standpoint,

[2]https://www.w3.org/WAI/ARIA/apg/

[3]https://www.w3.org/WAI/WCAG21/Understanding/contrast-minimum

this is possible thanks to the presence of the central server, which enables the communication between a user and a music box. Thus, the control messages a given box receives are independent of the user that generated them. This allows us to seamlessly handle switching from a local control to a remote control of a music box by another participant to the session, as long as he/she has the required permissions to do so.

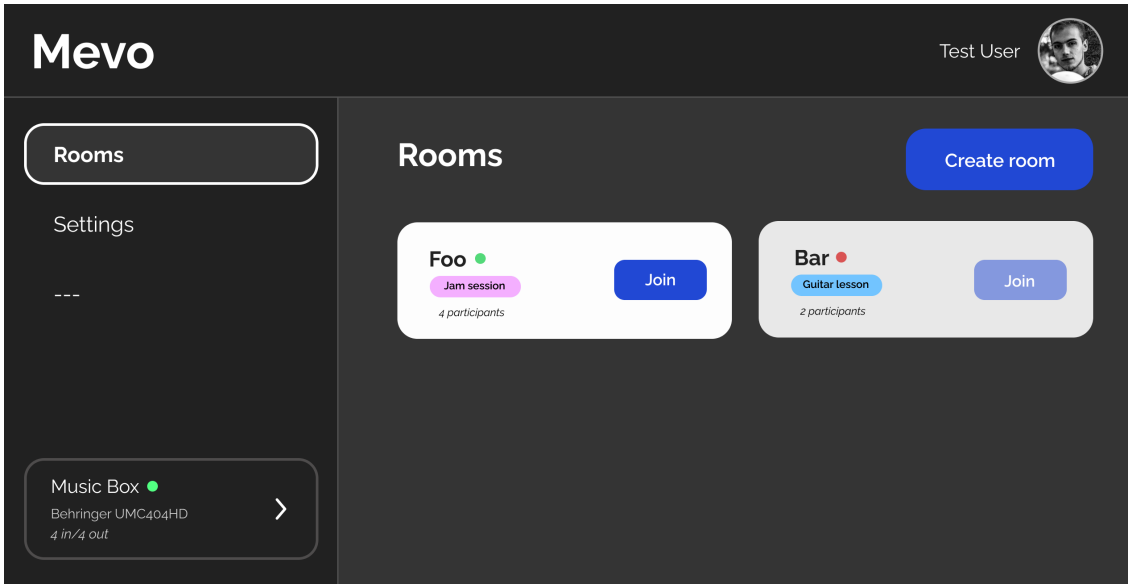


Figure 9.2: MEVO's Accessible GUI - Rooms page

9.5.2 Accessible Metronome

Metronomes can be easily integrated in NMP systems as done, e.g., in [18], even including adaptive capabilities to time-varying latency conditions. In turn, audio cues generated by the metronome can be complemented with visual and tactile feedback to offer a multisensory experience that caters to a broad spectrum of users, including BVI and DHH subjects.

More in detail, our prototype of accessible metronome is designed to operate in a multi-user scenario where every user is provided with equipment consisting of a software component running on a PC, that interacts with a wearable device (a bracelet in our setting). Such system offers three distinct methods for delivering metronome cues:

- Graphical: a user-friendly GUI facilitates the interaction and provides a clear visualization of the metronome's beats in real-time for DHH users.
- Tactile: the bracelet is equipped with vibrating motors activated synchronously with the metronome beats for BVI or DHH users;

- Auditory: both the bracelet (through buzzers), and the PC (through speakers/headset) can emit sounds associated to metronome clicks.

Two different GUIs were designed. One acts as orchestrator and is meant to be run by only one user at a time to set up the settings shared among all the participants to the session. The available settings can be expressed in forms of single values (or lists of values), each of which describes a group of adjacent bars with common characteristics. The possible settings are:

- Number of bars: it indicates the amount of bars in each group (leaving this field empty configures the metronome to continue operating indefinitely).
- Bpm per bar: it indicates the beats per minute value for bars belonging to each group.
- Tempo signature per bar: it indicates the numerator of the tempo signature (i.e., the number of beats per bar) for each group.

Conversely, the second GUI displays two circles, each one mapping respectively a strong and a weak beat. The blue circle indicates the strong beats of a bar, while the red circle shows the weak beats. This visual representation aids users in identifying the timing of the metronome's beats and the bars subdivision.

The custom hardware component of the metronome is a bracelet which utilizes an Arduino Nano (with ESP32 in our setting) to control two vibrating motors and two buzzers, with two potentiometers to independently control their volume. This configuration was designed so that one motor and one buzzer are associated with strong beats of a measure, in accordance with the chosen time signature, while the second motor and buzzer were assigned to signal the remaining weak beats of the measure.

The software not only takes care of preserving the synchronization between the graphical beats and the bracelet through periodic feedback from the latter, but also exchanges real-time data with the participants to the online music session with the aim of preserving synchronization with all the participants' devices, minimizing the effects of network jitter and clock drifts.

9.5.3 Tactile Feedback for Timbre Discrimination

MEVO supports visual representation of pre-recorded audio samples enhanced with tactile feedback, with the aim of facilitating discrimination of timbre and sonic grain by DHH users. As thoroughly described in [198], audio excerpts generated by different instruments are associated with:

- a visual representation of the instrument;

- a background image and a background color, which should be evocative of the emotions aroused in the listener[4];
- a texture pattern superimposed on the background, to render the tactile sensation associated with the background image.

Cross-modal associations among images, colors, sounds, and textures are thus exploited to offer a multisensorial perceptual experience to the users.

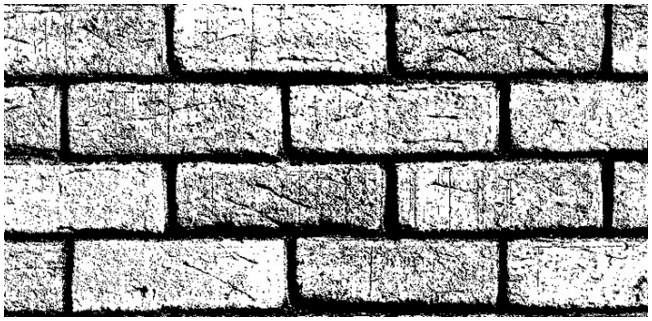


Figure 9.3: Example of black and white image rendering for haptic texture design of a brick wall [198]

Figure 9.4: Examples of visual representation of audio excerpts, enhanced with haptic feedback [198]

The current prototype leverages a TanvasTouch device[5], which enables a customized design of electroadhesion-based haptic feedback localized to the user's fingertip. Haptic textures are designed by means of black and white (B/W) images, where white corresponds to rough texture and black to smooth texture.

[4]The choice of colors and images to be associated with audio excerpts was taken based on the outcome of listening questionnaires, answered by 50 participants (see [198] for details)

[5]Tanvas. Surface Haptics. https://tanvas.co/technology

9.6 Technical Challenges

Based on the technology overview provided in Section 9.2, it emerges that accessible web-mediated musical interactions in real-time are a rather unexplored area, with only a few studies investigating collaborative music making [94], mainly due to the extremely low latency requirements necessary to maintain a synchronous musical interplay [191]. Therefore, how to successfully incorporate such inclusive technological solutions within NMP frameworks for remote music teaching and practices is a yet almost completely unexplored research direction, which urges for dedicated efforts by the scientific community. A roadmap to guide such efforts is provided in the remainder of this Section, which reviews the most significant technical challenges involved in the realization of an inclusive NMP system, discussing open research directions.

9.6.1 Latency Management and Packet Loss Concealment for Cross-Modal Data Streams

When heterogeneous data types (e.g., audio, gestural and haptic signals) need to be jointly transmitted (i.e., they cannot be processed and streamed separately due to low-latency and high-reliability application requirements), cross-modal transmission strategies must be adopted [259].

After acquisition and digitization, data that need to be streamed over a network are packetized before being transmitted, to adhere to the packet-oriented switching paradigm adopted in telecommunication networks. Then, every packet is routed individually through the telecommunication network infrastructure to be conveyed to its intended destination. Consecutive packets may experience different delays along their end-to-end route, since some latency components (e.g., the queueing delays introduced by routers traversed by the packet along its path) may vary over time. Variations of the interarrival times between consecutive packets (the so-called "jitter") and packet losses due to transmission errors or drops (actuated by routers to prevent excessive traffic congestion) may cause data portions to arrive too late to be reproduced in due time at the receiver side. In turn, this generates glitches (in the case of audio streams), deterioration of the rendered gestures (for motion data) or of the perceived haptic feedback (for tactile data).

State-of-the-art packet retransmission techniques, such as those implemented by the TCP protocol at transport layer [8], cannot be exploited by NMP applications, as they would further increase the mouth-to-ear latency. Therefore, Packet Loss Concealment (PLC) techniques specifically tailored for cross-modal streams should be devised [261, 46], while ensuring compatibility with the low-latency requirements of NMP. Additionally, dedicated solutions must be developed to effectively handle cross-modal coding, transmission and latency when synchronizing audio, visual, and haptic data streams in real-time. A recent survey on such techniques can be

found in [260].

Focusing on audio streaming, a further cause of artifacts is the so-called "drifting" effect caused by the imperfect alignment of local clock oscillators. Indeed, even when the clocks' nominal frequency is identical, small deviations may still occur due, e.g., to temperature variations or to fabrication imperfections. This leads to buffer over/underruns caused by the difference between the amount of audio samples generated by the sender and the amount of those reproduced by the receiver during a reference time window [262, 68]. Eventually, such differences cause either occasional drops of audio blocks or generate gaps in the audio stream. This side effect further motivates the need for adequate PLC mechanisms and/or for dedicated clock synchronization approaches (see e.g., [262, 68]).

9.6.2 Interoperability and Standardization

The need for interoperability among the heterogeneous components that are envisioned to be part of an inclusive NMP system has already emerged in [233], which addresses Internet of Musical Things (IoMusT) environments. The lack of standardization of interfaces and communication protocols further hinders from seamless integration of Musical Things (i.e., networked devices capable of generating, playing, or responding to musical content) in NMP frameworks. Though some preliminary steps towards standardization for IoMusT have recently been done [249], the specific use-case of NMP has been neglected so far.

9.6.3 Minimizing Network Contributions to Mouth-to-Ear Latency

The overall mouth-to-ear latency experienced by the users of NMP systems includes multiple contributions introduced by different stages of the audio acquisition, processing, transmission and reproduction chain. While acquisition, processing, buffering and playout delays can be controlled by NMP hardware/software designers, the latency component introduced by the network infrastructure is typically unknown and difficult to predict, as it is influenced by several factors such as the physical distance among performers, the network traffic congestion level (which exhibits variations over time), the traffic priority policies applied by network operators and ISPs, etc.

The exploitation of Software Defined Networking (SDN) [169], Network Function Virtualization (NFV) [95], and Multi-Access Edge Computing (MEC) [71] technologies is envisioned as a game-changer to ensure compliance to the QoS/QoE requirements of NMP systems. Such technologies should be incorporated in a back-end infrastructure dedicated to NMP applications, which should include tailored functionalities capable of dynamically adapting the routing of audio streams and the positioning of NVFs hosting NMP servers by jointly considering:

- the physical location of the involved users and of the candidate sites for NVFs placement, possibly exploiting MEC to ensure user proximity;
- the current network congestion level;
- predictions on the future evolution of traffic patterns.

Moreover, the involvement of ISPs and network operators in the design of commercial NMP services would enable the adoption of prioritization criteria for NMP-generated data (e.g., with a dedicated SDN slice), thus offering adequate QoS guarantees to the users in terms of latency and jitter.

To implement such functionalities, the backend portion of the NMP system should also include a Service Management and Orchestration (SMO) engine, responsible for instantiating NVFs hosting NMP servers within MEC nodes and to coherently allocate resources in the involved network segments. This would enable latency optimization by locating NVFs in strategic positions within the network infrastructure, depending on the users' physical location, bit-rate requirements, and on the traffic congestion level along the paths interconnecting users to servers. The SMO should also collect monitoring information from the users currently participating in the musical session (e.g., experienced latency and packet losses) to dynamically update the routing of audio flows as traffic conditions evolve. To this aim, the SMO could also exploit traffic prediction methods [139, 227] operating at various time scales. In turn, dedicated optimization algorithms implemented at the SMO premises will permit to reduce network latency by exploiting the outputs of such traffic predictors, which will enable proactive operations and dynamic adjustments of NVFs' locations depending on the current and forecasted network congestion conditions, thus ultimately improving the QoS/QoE perceived by the users.

Though a detailed discussion on such topics is out of the scope of this chapter, the interested reader is referred to [239] for a proposal of an NMP infrastructure supporting immersive audio rendering, which leverages NFVs and MEC technologies. The adoption of SDN technologies for NMP systems has been first envisioned in [131], which provides examples of possible interactions between a real-time network latency monitoring module and a NMP system. Despite such preliminary attempts, the potential of SDN integration in NMP contexts remains largely unexpressed.

9.7 Future Directions: Towards an Inclusive Musical Metaverse

As of today, the concept of Musical Metaverse (MM), i.e., a virtual, digital world where each user musically interacts through his/her own avatar with other users, starts being postulated in the research community. According to [232, p. 5],

the MM is "an interoperable persistent network of multiuser environments merging physical reality with digital reality, which serve a musical purpose. It is based on the convergence of Musical Extended Reality (Musical XR [236]) and Internet of Musical Things (IoMusT) technologies [242] that enable multisensory, networked musical interactions between musical stakeholders, as well as between such stakeholders and Musical XR environments and objects". The range of musical activities that could be performed in the MM encompasses composition, fruition of virtual performances, teaching, and collaborative content production.

In addition to IoMusT and Musical XR, the MM also builds upon a large pool of emerging technological advancements [232], including:

- *blockchain* and *Non-Fungible Tokens (NFTs)* to control, regulate and ensure transparency in copyright ownership of music-related material, as well as in royalties distribution;
- *digital twins* to create virtual musical entities replicating physical ones, such as music instruments, acoustic scenes etc.;
- *Artificial Intelligence and Machine Learning* to enhance user-experience by improving proactivity and context-awareness of musical services. To this aim, ML-based predictive algorithms leveraging user-gathered sensing data (e.g., musical gestures), can be exploited, as well as recognition/classification methods to categorize musical content.
- *5G-and-beyond communication infrastructure* to ensure seamless fruition of the MM by adhering to strict latency, bit-rate and reliability requirements while providing widespread access and supporting user mobility.

Among the opportunities that are envisioned to be unlocked by a widespread adoption of the MM, it is worth mentioning [232]:

- the growth of unprecedented forms of musical expressions and activities, entailing novel ways of engagement of the audience and fostering collaborative creation among performers;
- the constitution of new social communities and of novel music-based social interactions, with their own management and governmental rules;
- novel monetization opportunities for the music industry, related to music streaming and distribution in the MM.

It is also envisioned that inclusiveness will be one important characteristic of the MM, which will pave the way for novel opportunities for disabled musicians and audiences to access musical experiences [232]. For example, XR-based musical instruments could ease control and manipulation for visual/mobility-impaired users. Similarly, fruition and participation to music events in the MM could be extended

to disabled subjects for whom physical participation to in-presence performances is hindered or precluded.

We believe that inclusive NMP systems will constitute a pivotal building block for the realization of a disability-friendly MM, by providing the technological substrate upon which such a virtual world will be built. Indeed, based on the MM framework proposed in [232], NMP systems will act as middleman to bridge the physical layer (constituted by users, Musical Things, etc.) and the link layer, which encompasses the networking and communication substrate, as well storage and computational nodes in the cloud. However, current NMP systems will need to be significantly evolved and adapted for seamless incorporation in MM ecosystems, thus calling for dedicated research efforts by the scientific community. In particular, 6G communication technologies combined with MEC functionalities are envisioned for future integration in NMP infrastructures, as they are capable of supporting real-time streaming of huge volumes of cross-modal data, thus finally enabling VR delivery for fully-immersive experiences [266].

Chapter 10

Remote Orchestral Conduction via a Virtual Reality System

Standard NMP systems transmit audio but neglect the visual channel that orchestral musicians depend on for coordination. Video streaming can restore this channel but imposes bandwidth and latency costs that may exceed residential network capacity. This chapter explores an alternative: capturing conductor gestures through VR hand tracking and transmitting pose data to remote musicians who view the conductor as an avatar. Because pose data is orders of magnitude smaller than video, this approach can operate within typical Internet connection constraints. Within the broader context of the inclusive accessibility framework discussed in section 9.3, this system addresses the visual modality, enabling sighted musicians to perceive conducting cues remotely.

10.1 Introduction

As established in Chapter 1, musicians rely on visual cues to synchronize their performances and convey expressive intentions [209].

The most common form of visual feedback that a NMP system could exploit is a video stream, as typically done in videoconferencing platforms. Attempts to eliminate the encoding/decoding latency by streaming uncompressed video [58] resulted in a minimum network bandwidth requirement of 1 Gb/s (and even higher if the HD video configuration is used[1]).

[1] https://lola.conts.it/downloads/Lola_Manual_2.0.0_rev_001.pdf

In this study, we explore an alternative to video transmission in the context of a remote orchestra conducting scenario: we exploit a Virtual Reality (VR) headset to extract and convey the conductor's gestures to a geographically spread orchestra interconnected via a NMP system. Our proposed approach has the twofold goal of reducing *i*) the required transmission bit-rate, since we need to transmit only data related to the pose of hands and head of the conductor, and *ii*) the Motion-to-Photon (M2P) latency, i.e., the time taken for a user gesture to result in a corresponding visual change on a display [113], as pose tracking can operate at a higher frame rate than what is commonly supported by cameras.

Numerical results show that the average M2P latency achieved by our experimental setup is on average 140 ms and never exceeds 210 ms, which constitutes a promising starting point for future potential integration in a Musical Metaverse (MM) framework [232]. Indeed, the MM concept foresees the creation of a digital universe where users can interact musically via avatars, blending the physical and digital worlds by means of Musical Extended Reality (Musical XR) and Internet of Musical Things technologies [241]. In particular, we envision the adoption of our proposed system as a building block for MM-mediated orchestra rehearsals and performances, as already occurred in [150].

The remainder of the chapter is organized as follows: Section 10.2 reviews related studies, Section 10.3 provides an overview on the system with technical details on its implementation, Section 10.4 describes the methodology used to evaluate the performance of the system, Section 10.5 presents the obtained results, Section 10.6 outlines the identified limitations.

10.2 Related Work

Several studies tackled the assessment of the maximum acceptable asynchrony between audio and video in real-time streaming. In [224], the authors show that the perception of misalignment is deeply linked to the type of content considered. As a general outcome, the study highlights that when the lag between visual and audio streams is greater than 100 ms, more than a half of the subjects involved in the test campaign therein reported were able to perceive it, thus suggesting that lag should not exceed 100 ms.

In [172], the authors explored how different lags between audio and video impacted a musical performance involving remote musicians and a conductor. One interesting outcome of that study is that visualization of the conductor's gestural cues was considered extremely important by the musicians, as it helped mitigate the negative impact of audio latency on the performance quality. Furthermore, high audio latency was found to be much more impactful on the musicians' capability of maintaining a stable tempo w.r.t. video latency, thus suggesting that higher video delays may be tolerated, provided that they not exceed 100-200 ms.

A preliminary attempt to bypass video streaming in NMP practices can be found

in [35], where a conventional computer mouse was proposed as a replacement for the conductor's baton in a NMP scenario, with the aim of providing low bit-rate and low latency visual feedback while saving network resources for audio streaming. Results show that the proposed mechanism, though not user-friendly, proved viable for conducted music, with conductor and orchestra tolerating one-way transmission delays in the range from 35 to 75 ms.

More recently, several studies investigated real-time streaming of control and/or gestural data for music-related applications in the Metaverse. In [245, 105], extended reality (XR) platforms are proposed to support bidirectional polyrhythmic interactions between players, embodied by avatars, exploiting a virtual drum circle. Instead, our proposed scenario considers VR instead of XR and does not constrain the choice of the musicians' instruments to drums. In [109], an Extended Reality Environment (XRE) for immersive NMP is presented. The framework implements a position-dependent visual and auditory representation of the users within a shared virtual space, characterized by early reflections and diffuse reverberation. Our system could be integrated with a similar audio rendering framework to enhance the feeling of co-presence experienced by conductor and orchestra members. In [108, 32], upper body tracking is exploited for avatar representation of four remote musicians, coupled with immersive audio rendering of the virtual space. Differently, our system only tracks the conductor's head and hands movements.

10.3 System Overview

In this Section, we present our proposed framework for transmitting visual cues for remote orchestra conduction and explore the potential approaches for connecting a VR headset worn by the conductor to a web application used by the musicians in the NMP session.

The application running on the VR headset captures head and hand movements of the wearer and displays them in an immersive scenario. This scenario is shown from the conductor's point of view, with the musicians in front of him/her. Musicians are represented as avatars, with their name reported on top of them, and are positioned in the virtual space in a way that facilitates the conductor to identify and point to each musician individually. An example of the conductor's perspective is shown in Fig. 10.1.

On the musician's side, a receiver device obtains the pose data from the conductor and renders it on a screen. It displays a virtual scenario depicting the avatars of the conductor and of the other musicians in the same field of view. An example of the conductor's avatar seen from the perspective of four different musicians is shown in Fig. 10.2. The positioning of each individual musician in the scene visualized by the conductor through the VR headset is coherent with the positioning of the conductor in the personalized visualization offered to each musician.

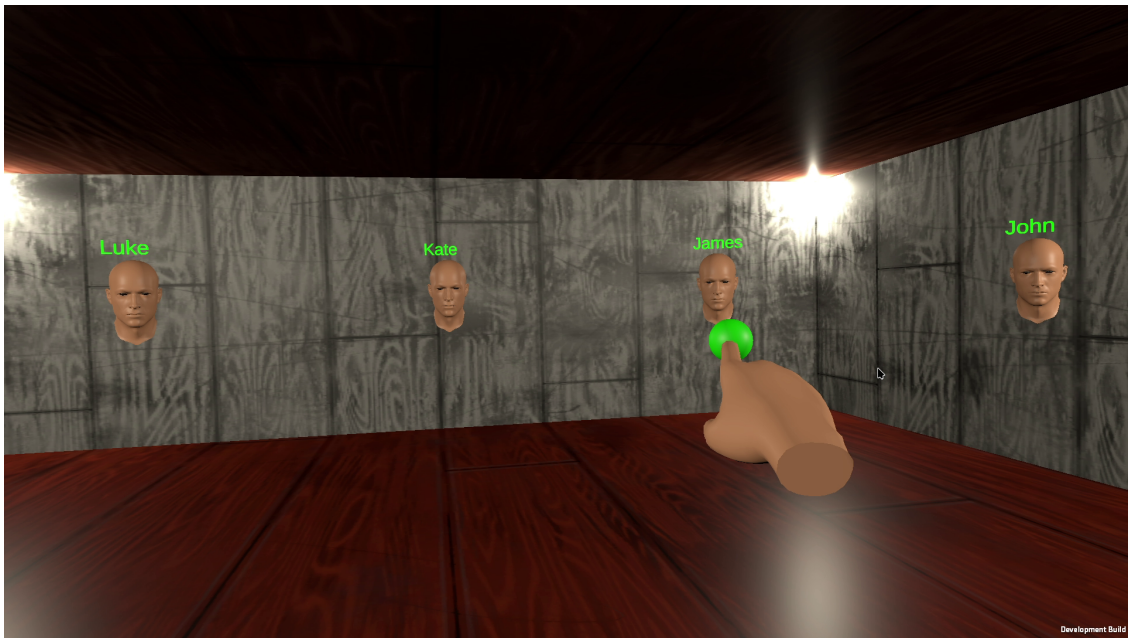


Figure 10.1: Conductor's avatar pointing at *James*, seen from the conductor's perspective.

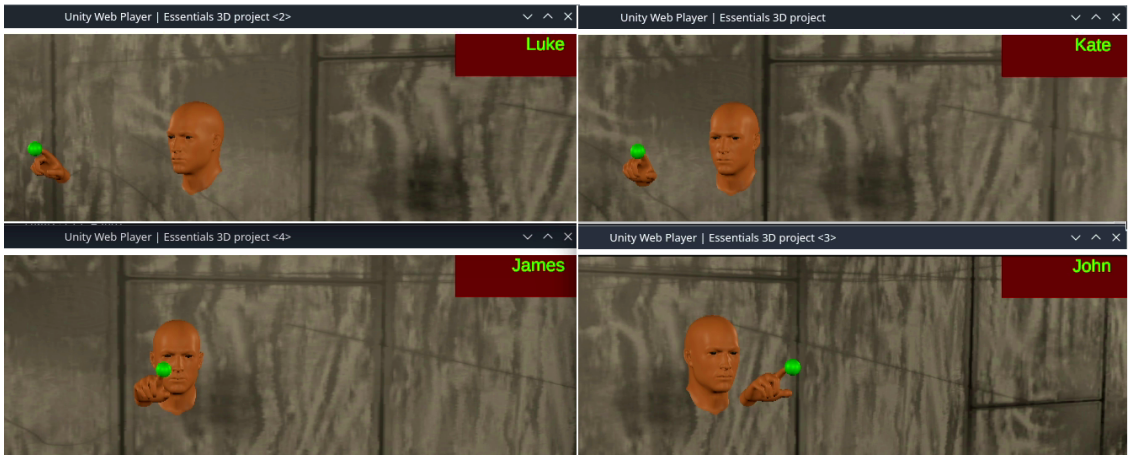


Figure 10.2: The four renders displayed by the musicians' web application in the same scenario of Fig. 10.1

10.3.1 Transmission Setup

The main challenge of this system is the streaming of real-time pose data from the VR headset to the remote musicians' devices. First, even though any wired or wireless transmission medium could be leveraged to convey pose data to the musicians' devices (equipped with a web browser to visualize the conductor's gestures), it is highly recommended to rely on a wired Internet connection, since latency and

jitter figures achievable by Wi-Fi and 4G technologies are hardly compatible with NMP requirements. Second, web applications cannot directly handle TCP connections or UDP packets. However, they can leverage WebRTC, that supports the exchange of messages in a peer-to-peer (P2P) fashion, as well ass the use of unreliable (UDP-like) data channels [121], which is the preferred choice when dealing with real-time data. Thus, our application uses WebRTC for the conductor's pose data streaming.

The application running on the VR headset is not strictly bound to any specific transport protocol. However, the VR headset has limited options when it comes to physical connections. It supports *i*) Wi-Fi, through an integrated Wi-Fi adapter; *ii*) a Wired Ethernet adapter over USB-C; *iii*) Android Debug Bridge (ADB)[2] on top of USB. During testing, we discovered that Ethernet support varies among VR headsets. At the time of writing, the Meta Quest 2 does not support it, while the Meta Quest 3 does. However, the graphical interface of the Meta Quest 3 does not provide consistent information about the connectivity status when using Ethernet, indicating that its integration is not yet stable. For such reasons, we opted for not leveraging an Ethernet connection, and relied instead on the ADB option since, as shown in Sec. 10.5, it yields comparable latency results while being more stable software-wise. The ADB option has the main drawback of requiring an additional device to enable the connectivity from the VR headset to the Internet. However, since in a realistic scenario the conductor needs to receive the audio streams produced by the remote musicians, we reckon on the presence of a device, which can also act as a streaming device for pose data, that may additionally run a NMP system for audio transmission (e.g., JackTrip [29]). Note that our system is agnostic to the number of audio channels streamed by the NMP system, i.e., both stereo and immersive (3D) audio streaming can be used.

In summary, in our setup, the VR headset is connected to a streaming device via USB. The application on the VR headset communicates with other peers and the server through a TCP socket forwarded by ADB reverse forwarding. The streaming device, i.e., a generic computer running the ADB tool, handles the connections. Messages in the TCP socket are synchronously produced and consumed by the application immediately after each graphic frame generation.

10.3.2 Framework Description

Our VR-based system prototype for remote orchestra conduction consists of four main components, as shown in Fig. 10.3:

- **VR headset**: worn by the orchestra conductor to track his/her hands and head movements and to visualize the remote performers via avatars. In our

[2] https://developer.android.com/tools/adb

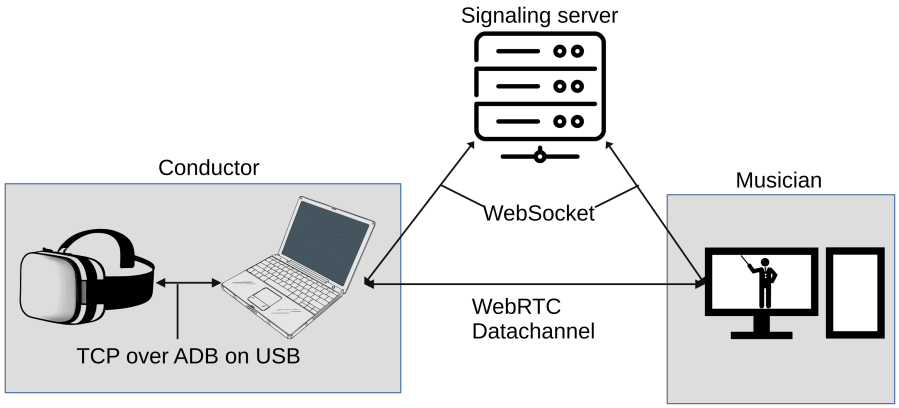


Figure 10.3: System overview

experiments, the VR headset used was the Meta Quest 3.

- **Streaming device**: a PC connected to the VR headset via USB and to the Internet through a wired connection. It takes care of receiving pose data from the VR headset through ADB and of streaming it to the remote musicians through WebRTC.
- **Signaling server**: a server that facilitates the setup of WebRTC P2P connections.
- **Receiver(s) device**: a PC connected to the Internet with a wired connection. It takes care of receiving and displaying the conductor's movements to the musicians by means of a web application.

Notably, the adoption of WebRTC is in line with recently proposed implementations of multi-user applications for the MM, e.g, in [25, 61, 60].

The application running on the VR headset is built with the Unity game engine[3] and the aid of the Oculus SDK[4]. The web application running on each receiver's device is also built with Unity and compiled as a WebGL target. It reuses the same scenario and layout of the application running on the VR headset, making the interaction and the positioning coherent among the two entities. The streaming device application and signaling server are Node.js applications.

10.4 Evaluation Methodology

The M2P latency can be decomposed in:

[3] https://unity.com/

[4] https://developer.oculus.com/downloads/unity/

Acquisition/Tracking latency time required by software and hardware components of the VR headset to track head movements and hand gestures.

Processing latency time taken by the application to incorporate the collected pose data in the message to be sent.

Network transmission latency time needed to convey messages from the VR headset to the musicians' application.

Rendering latency (receiver device) time required to graphically render the received data.

Video output latency (receiver device) time required by the driver and hardware to display the rendered frames.

We consider three different latency-related measurements:

- *Network packet jitter*: message generation happens at fixed intervals (once per frame generation, i.e., $d = 1000/f$, where d indicates the delay in milliseconds and f indicates the frame rate, measured in frames per seconds (fps), of the VR headset). At the receiver side, we measure the inter-arrival time between two consecutive messages.
- *Messages Round-Trip-Time (RTT)*: we measure the time difference between the dispatch of an "echo request" to the streaming device (or musician application), to the corresponding reception of an "echo reply" message. It is measured in terms of d, which represents our time granularity for every measure performed in the VR headset.
- *M2P latency of hand movement*: using a camera with a frame rate of 119 fps, we record the movement of a single hand executing an upward conducting gesture, close to the screen where the musicians' application is being displayed. We then manually annotate the video frames corresponding to the beginning and the end of a movement, for both the real hand and the rendered one. The time difference between the frame annotated as the start of a given movement of the real hand and the corresponding frame for the rendered hand represents the M2P latency. It is measured in number of frames and then converted in milliseconds.

10.5 Results

10.5.1 Testbed Description

The test environment consists of a single machine concurrently operating as streaming device, signaling server and receiver device. The machine is a laptop with an Intel i5-7300U, 12 GB of RAM and a Linux distribution (kernel 6.8.5 PREEMPT_DYNAMIC). The musicians' web-application runs in Chromium (version 123). The laptop is connected to an external monitor (HP e24m G4) through DisplayPort over USB-C. The monitor is used to display the video output visualized in the browser, based on which all M2P measurements were collected.

10.5.2 Transmission Jitter

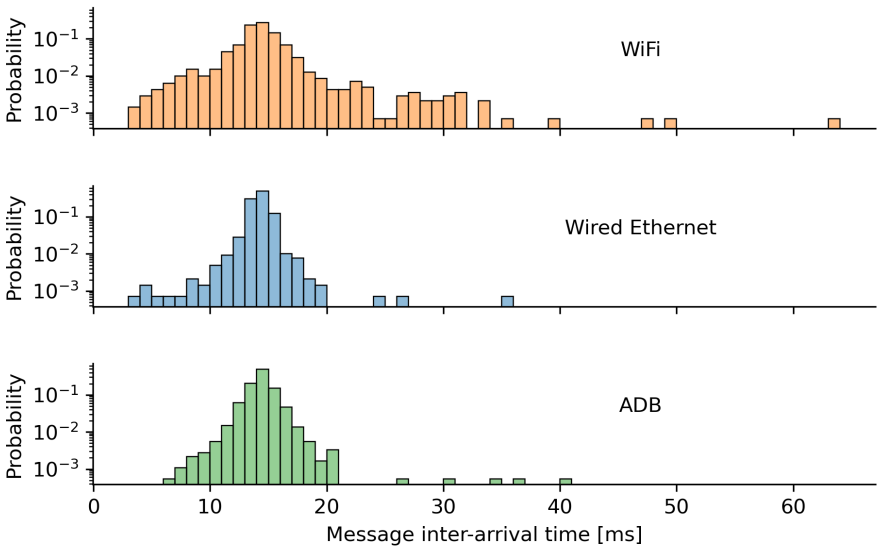


Figure 10.4: Distributions of message inter-arrival times for the three transmission stacks. The ideal value should be ~ 14 ms $= 1000/72$ fps.

Fig. 10.4 compares the jitter values achieved by the three different connections methods available on the VR headset. The wired-Ethernet connection (through the Ethernet to USB-C adapter) results to be the most stable one in terms of jitter, followed by ADB, whereas the Wi-Fi connection yields the worst jitter figures. While these results suggest that the best solution would be the use of the Ethernet connection through a USB-to-Ethernet adapter, we preferred the ADB option for the reasons described in Sec. 10.3.1.

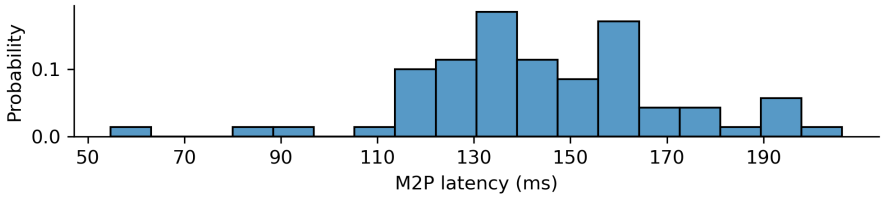


Figure 10.5: M2P latency of the whole chain.

10.5.3 Messages RTT Measurements

Regarding RTT, we performed two measurements:

VR headset to streaming device (short path) the VR headset application sends an "echo request" at time t_1, and the application in the streaming device replies with an "echo reply". The VR headset receives the answer at time t_2. A single measure of the RTT (R) is equal to $t_2 - t_1$.

VR headset to musician application (long path) the application in the VR headset runs in the same way as above. The application in the streaming device forwards the "echo request" to a single connected musician's application, which replies to it. The streaming device forwards back the "echo reply" to the VR headset through ADB.

The short path RTT measurements yielded 99.7% of the measurements exhibiting a value of $R = 1d$ and the remaining 0.3% exhibiting a value of $R = 2d$. Instead, the long path RTT experiment yielded 83.4% of the measures exhibiting a value of $R = 1d$, 13% with value $R = 2d$, 2.5% with value $R = 3d$, and the remaining 1.1% with values greater than $3d$. The RTT measured in terms of d can be converted in milliseconds by knowing the frame rate of the VR headset (f). The RTT expressed in ms (R_{ms}) is bound within the range $(R-1) <= \frac{R_{ms} \cdot f}{1000} <= R$. The total chain RTT (i.e., from the VR headset application to the musician's application) deserves further investigation but, assuming a symmetric RTT, we can infer that the one-way transmission latency is less than 21 ms in 99% of the cases (note that $f = 72$ fps on our VR headset). Pose data is rendered as it arrives, since no buffering is performed at the receiver's device. This has the effect of minimizing the latency at the cost of occasional graphical frame stuttering.

10.5.4 M2P Latency of Hand Movement

Fig. 10.5 depicts the measured M2P latency of hand movements. The camera used for recording had a frame rate of 119 fps. The median latency value is 17 frames, i.e., $\sim$ 143 ms. Based on the previously discussed RTT measurements,

we know that latency due to transmission is lower than 21 ms. According to Abdlkarim et al. [2], the hand tracking latency for the Meta Quest 2 VR headset should be in the order of 40 ms. Other sources [253] state that the Meta Quest 3 should have a slightly lower hand tracking latency than the of Meta Quest 2, i.e., around $\sim$ 30 ms. This would imply that at most $\sim$ 90 ms are introduced by the processing time on board of the VR headset plus the rendering and output at musician's application side.

10.5.5 Bit-rate Requirements

Bit-rate requirements can be computed deterministically, assuming that both hands are tracked and that messages are sent once per frame generation. For each message we track the 6D pose of 24 bones per hand, for both hands, plus one pose representing the head. Overall, 49 6D poses are tracked. Each 6D pose consists of the Cartesian position and rotation data, both represented as triplets of 4 bytes floating-point values, totaling $2 \cdot 3 \cdot 4 = 24$ bytes per bone. Therefore, each message is 1176 bytes long, with an overhead of 24 bytes for our protocol implementation, thus yielding 1200 bytes. It follows that the bit-rate required for each communication channel director-musician amounts to 86.4 KB/s, i.e., 691.2 Kb/s. It is worth noting that the communication is stateless, i.e., every message conveys the current state of the system. Thus, a single loss only affects the receiver's state until the next message is received, avoiding any error accumulation issues.

It should be noted that further reductions of the bit-rate requirements could be achieved by diminishing the bit-depth of position and rotation data representations, at the price of a small degradation of the accuracy of the rendered pose. However, such degradation would not worsen over time, due to the absence of error accumulation.

10.6 Limitations of the Proposed Approach

Although the results obtained are promising, the proposed prototype has some limitations in its setup. Firstly, the current prototype is not designed to accommodate more than one musician per device. Indeed, a single device is represented as one avatar from the conductor's perspective, making it impossible to distinguish between multiple musicians. Secondly, the prototype relies on embedded motion tracking features, which are based on a camera paired with deep-learning algorithms, both of which introduce substantial latency. Sensor-based alternatives could be investigated. Lastly, to the best of our knowledge, commercially available VR headsets offer limited customization options from both hardware and software standpoints. For instance, state-of-the-art NMP applications often rely on custom scheduling, as seen in JackTrip [29], and on the use of wired Ethernet connections, which was not feasible with the selected VR headset.

Chapter 11

Designing Haptic Feedback Stimuli to Convey Gestures of an Orchestra Conductor to Visually-Impaired Musicians

The VR-based conducting system of Chapter 10 benefits sighted musicians but provides no pathway for blind and visually impaired performers who cannot perceive avatar representations. This chapter investigates haptic feedback as an alternative modality for conveying conducting gestures, addressing the tactile component of the inclusive accessibility framework introduced in Chapter 9. Through interviews with both sighted and visually impaired musicians, design specifications are developed for wearable devices that translate conductor movements into vibrotactile patterns, with placement varying according to instrument-specific constraints.

11.1 Introduction

While haptic and tactile feedback systems have shown promise in compensating for visual information in various domains, the translation of complex conducting gestures (which simultaneously convey beat patterns, dynamics, phrasing, and emotional expression) into intuitive and interpretable haptic feedback presents unique challenges. Such systems must carefully balance both technological and perceptual design constraints to effectively bridge the gap between visual conducting cues and tactile perception.

Through interviews with sighted and BVI musicians, we identify key constraints and preferences for haptic feedback in musical contexts. We combine these findings with technical literature to propose design specifications for a wearable haptic system. Specifically, we present: (i) an analysis of body placement suitability across different instruments, (ii) technical requirements for vibrotactile actuators and signal design based on perceptual studies, and (iii) functional specifications for a prototype system that captures conductor gestures and translates them into structured and interpretable vibrotactile patterns known as *tactons* [28], delivered through wearable devices. Our findings lay the groundwork for future research into inclusive, embodied musical systems for BVI users.

The remainder of the chapter is organized as follows: Section 11.2 provides some background notions on musical conduction and haptic feedback technologies. Section 11.4 describes the questionnaire construction and reports the results of the conducted interviews, whereas the design guidelines derived from the analysis of the collected answers are presented and mapped onto functional and non-functional specifications for prototyping in Section 11.5. Conclusions are offered in the last Section.

11.2 Background

11.2.1 Notions on Musical Conduction

While conductors generally develop their own unique styles, conducting courses in conservatories and academies present a variety of techniques, both as historical knowledge and as practical tools that can be applied in different scenarios and with different ensembles [122].

In [82], a systematic framework for describing the gestural vocabulary of conductors is discussed, which proposes a lexicon organized into four main categories: Articulation, Dynamics, Attack, and Cut-off. Within each category, gestures are further classified into distinct classes according to their structural and expressive properties. The model highlights how movement shapes combine with hand configurations to produce gestures with recognizable meanings in rehearsal and performance contexts.

An important aspect of conducting practice is the asymmetry of conducting, which reflects the distinct roles of the two hands. The dominant hand, typically holding the baton, is primarily responsible for keeping time and marking the ictus (i.e., the precise point at which the beat occurs), whereas the non-dominant hand is more involved in expressive communication—indicating dynamics, phrasing, or cueing specific sections of the ensemble. This asymmetry plays a crucial role in the efficiency and clarity of communication between conductor and musicians [82, 127].

Beyond hand gestures, conducting also incorporates significant non-verbal and aural cues, such as breathing, facial expressions, and eye contact. These subtle

Table 11.1: List of common conduction gestures

Gesture	**Description**
Beat	Gestural indication of beat patterns
Accelerando	Quick, energic gesture indicating tempo increase
Rallentando	Soft, relaxed gesture indicating tempo decrease
Crescendo	Expanding gesture indicating volume increase
Diminuendo	Contracting gesture indicating volume decrease
Sustain pitch	Adjust to compensate falling pitch
Sharp entrance	Firm gesture indicating begin of musical phrase
Gentle entrance	Delicate gesture indicating begin of the piece/phrase
Sharp cut-off	Firm stop motion indicating end of a piece/phrase
Gentle cut-off	Delicate stop motion indicating end of a piece/phrase
Legato	Smooth, flowing motions to convey legato articulation
Staccato	Crisp, quick motions to convey staccato articulation

signals can strongly influence ensemble coordination and musical expressivity [246], particularly in entrances and phrase shaping, where the conductor's breath often acts as a shared preparatory cue analogous to that of a singer or wind player.

Finally, conducting differs significantly between rehearsals, teaching and performance. During rehearsals, conductors often interrupt, isolate sections, or exaggerate gestures to provide feedback and clarify musical intentions. In [144], it was found that conductors' gestures in rehearsals were more expressive and varied, while in teaching settings, gestures were more focused and instructional. In performance, however, gestures tend to be more compact, refined, and focused on maintaining flow and ensemble cohesion.

Recent developments have also explored telematic conducting, where conductors lead ensembles across physical distances using network technologies [34, 188, 217]. While technically feasible, such practices face challenges related to latency, limited transmission of subtle gestures, and the reduced effectiveness of non-verbal communication. Nevertheless, research in this area highlights the adaptability of conducting practice to new technological contexts.

In order to identify the gestures that are most widely adopted in practice, we interviewed an experienced choir and orchestra conductor (male, aged 76, with

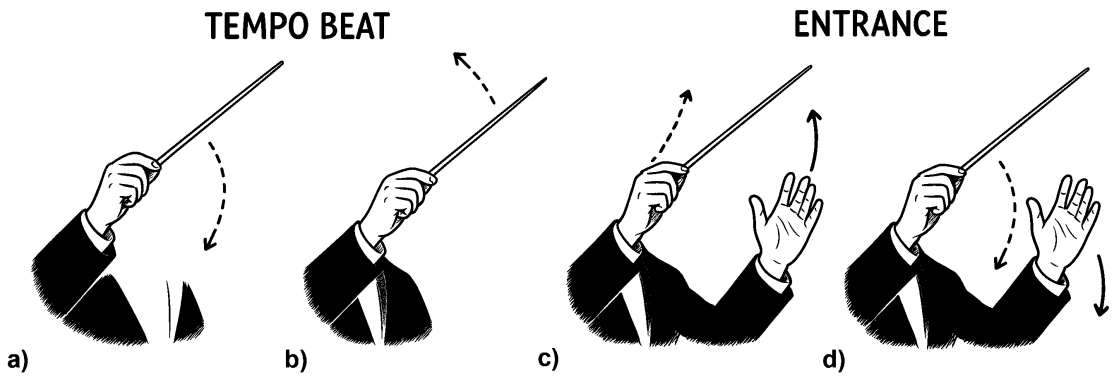


Figure 11.1: Examples of typical conduction gestures: tempo cues for a binary pattern (a-b) and entrance cue (c-d).

more than 40 years of conduction experience, mainly in classical repertoires). The outcome of the interview is the list of gestures reported in Table 11.1, which can be grouped into three different categories: tempo-related gestures (e.g., beat, *accelerando*, *rallentando*), dynamic gestures (e.g., *crescendo*, *diminuendo*, sustain pitch), and articulation or phrasing gestures (e.g., *legato*, *staccato*, sharp entrance, gentle entrance, gentle cut-off, sharp cut-off). These categories highlight the dual function of conducting gestures: on one hand, to provide precise technical information such as beat, tempo, or articulation, and on the other hand, to convey expressive intent and musical character.

An illustration of two fundamental types of gestural cues—respectively indicating tempo beat and entrance—is presented in Figure 11.1. More in detail, for what concerns beat indications, conductors typically trace standardized trajectories in the air with their baton or right hand. Each beat has an ictus, serving as the decisive timing cue for the ensemble. For instance, in a simple duple pattern (2/4 or 2/2), the first beat is shown with a downward motion (the strong beat, see Figure 11.1a), followed by an upward motion for the second beat (the weaker beat, preparing the next downbeat, see Figure 11.1b). Figures 11.1c-d show instead a typical entrance gesture: the conductor first prepares the players with a slight upward or lateral motion, comparable to giving a visual "breath," (Figure 11.1c) and then marks the exact moment of sound with a small, sharp ictus (Figure 11.1d). The quality of the gesture varies with the character of the attack—firm and energetic for a strong entrance, or smooth and delicate for a softer one. The left hand may reinforce the cue by opening for a broad sound or closing for a more focused entry.

11.2.2 Notions on Haptic Feedback

Numerous studies have demonstrated the potential of tactile feedback to compensate for visual or auditory information, particularly for users with sensory impairments. For example, haptic navigation aids for blind users [181, 124], obstacle detection systems using vibrotactile feedback [138], and tactile art experiences [81] all underscore the value of touch as an alternative communication channel. Prior studies have demonstrated the potential of haptic systems in enhancing spatial awareness of BVI subjects [181], guiding movement through vibrotactile cues [140], and improving accessibility in virtual environments [180].

A particularly relevant concept in tactile communication is the "tacton" (tactile icon) [28]. Tactons are structured, abstract tactile messages designed to communicate information non-visually, serving as the tactile equivalent of visual icons and auditory earcons. These messages can be constructed using various parameters, including frequency, amplitude, duration, rhythm, and spatial location of tactile stimulation. Tactons enable the encoding of complex information into distinguishable tactile patterns, making them particularly valuable for situations where visual displays are unavailable, overloaded, or inaccessible to users with visual impairments.

In the musical domain, haptic technologies have been leveraged to support a variety of tasks, including vibrotactile metronomes for deaf or hard-of-hearing musicians [192, 84], tactile systems for the conveyance of pitch, timbre, melody [186] or musical emotions [240], for assistance in instrument learning [147], or for providing real-time feedback for interactive performances [189]. However, these systems focus on reinforcing time-based cues or on conveying some musical features of an audio track, but they rarely address the expressive, multidimensional qualities of conducting gestures.

11.3 Related Works

The DIAMI (Distributed Intelligent Environment for Blind Musicians) architecture presented by Bajo et al. [15] provides an early example of using motion capture and tactile feedback for BVI musicians in orchestral settings. Their system employed WiiMote technology to capture conductor gestures via an infrared sensor on the baton, with movements interpreted by a central computer using GlovePie software and transmitted wirelessly via Arduino Bluetooth to a bracelet worn by the blind musician. The system encoded conducting information by analyzing directional changes in baton movements and mapping these to four strategically placed vibrators on the musician's arm. In testing with five blind musicians and two conductors, the system achieved a 98% acceptance rate among blind musicians, with error rates decreasing from 20% to 2% as users gained experience with the system. However, the systems limits the motion capture to the dominant hand and imposes

the usage of the baton to the conductor.

11.4 Questionnaire Structure and Results

11.4.1 Questionnaire Structure

To gather user-centered insights for the design of a system that conveys conducting gestures via haptic feedback, we conducted a qualitative questionnaire targeting both sighted and BVI musicians. The questionnaire aimed to understand musicians' experiences with haptic devices, identify suitable body placements for wearable systems, and gather perspectives on the role of conductors in orchestral performance.

The questionnaire was administered by means of oral interviews. Through open-ended questions, participants were asked to reflect on:

- Their familiarity with tactile sensory feedback (vibrations, touch) and contexts of use;
- Which body parts they consider most suitable for perceiving tactile signals, considering their own musical instrument and tactile acuity;
- Areas of the body where they would not want haptic devices placed;
- Preferences regarding haptic device characteristics (fixed vs. adjustable, intensity levels of vibrational feedback);
- Which conductor cues are most effective, and at what moments is it essential to visually engage with the conductor (for sighted musicians);
- Challenges encountered in orchestral settings and methods used to follow conductors' cues (for BVI musicians).

11.4.2 Questionnaire Outcomes

A total of 16 sighted musicians, aged 20 to 28, and 10 BVI musicians, aged 38 to 79, participated in the study. In Table 11.2, we present the participants' profiles, the type of visual impairment (for BVI musicians), the instrument(s) played, the reported years of experience (when available) and level of expertise, and the main music genre played. Additionally, Table 11.3 summarizes, by instrument group, the body parts that participants indicated as preferred or to be avoided for the placement of vibrational motors used to deliver haptic feedback.

In the following, we summarize the main survey outcomes regarding the role of conductors in orchestral performance, preferred modalities for non-visual feedback, body placement preferences, and potential usage concerns.

Table 11.2: Profiles of Interviewees

ID	Age	Sex	Visual Impairment	Instrument (years of experience)	Expertise	Music Genre
S1	23	F	-	Harp (13)	Amateur	Classical/ Contemporary
S2	21	M	-	Flute (10)	Amateur	Classical
S3	28	M	-	Trombone (8)	Amateur	Classical
S4	21	F	-	Violin (14)	Amateur	Classical
S5	21	M	-	Trombone (12)	Amateur	Classical
S6	21	M	-	Trombone (13)	Amateur	Classical
S7	25	F	-	Flute (12)	Amateur	Classical
S8	24	M	-	Cello (18)	Amateur	Classical
S9	23	F	-	Harp (10)	Amateur	Classical
S10	23	M	-	Guitar (13)	Amateur	Classical
S11	24	F	-	Piano (7)	Amateur	Classical
S12	24	F	-	Piano (10)	Amateur	Classical/ Contemporary
S13	22	F	-	Flute (10)	Amateur	Classical
S14	23	M	-	Trumpet (15)	Amateur	Classical/ Marching Band
S15	23	M	-	Violin (15)	Amateur	Classical
S16	20	F	-	Cello (14)	Amateur	Classical
B1	65	M	Shadow after 45	Trumpet (- before shadow)	Beginner	Marching Band
B2	38	F	Shadow since birth	Vocals (5), Piano (1), Accordion (-)	Beginner	Classical/Opera
B3	54	M	Light and contrast perception only, after 5	Guitar (-)	Amateur	Contemporary
B4	63	M	Blind after 21	Accordion (-)	Beginner	Classical
B5	54	M	Blind since birth	Piano (50)	Professional	Various
B6	57	M	Blind after 8	Piano (50), Accordion (-)	Professional	Various
B7	59	F	Blind after 44	Vocals (-), Tom Tom (-)	Beginner	Traditional
B8	51	M	Light and contrast perception only, since birth	Guitar (-)	Beginner	Classical
B9	79	F	Light and contrast perception only, since birth	Piano (-)	Professional	Various
B10	76	F	Light and contrast perception only, after 62	Percussions (-)	Amateur	Marching Band

Role of the Conductor

Based on interview responses, participants identified the conductor's essential functions in orchestral performance:

- **Temporal coordination**: Establishing and maintaining the beat throughout the piece, signaling tempo changes, and ensuring rhythmic unity across the orchestra.
- **Expressive guidance**: Conveying dynamics, phrasing, and emotional interpretation through gestures, shaping the overall sound and artistic dimension of the performance.
- **Structural navigation**: Providing clear cues for entrances of different sections, indicating the start of musical phrases, leading climaxes, and managing transitions between sections.

Participants noted that, even during silences, the conductor's presence remains essential for shaping duration and maintaining ensemble cohesion. For BVI musicians specifically, the visual nature of these cues presents the primary barrier to

Table 11.3: Instrument compatibility with haptic feedback

Instrument	Participant	Feet	Ankles	Legs	Knees	Thighs	Belt	Belly	Chest/Torso	Back	Hands	Wrists	Arms	Shoulders	Neck	Face	Head	Ear
Harp	S1					✗					✔	✔	✔					
	S9	✔	✔					✗		✗	✗		✔		✗		✗	
Piano	S11	✔										✔					✗	
	S12			✔									✔				✗	
	B2				✗		✔				❍							
	B5						✗			✗	❍		✔	✔				
	B6		✔									✔					✗	✗
	B9																	
Trumpet	S14		✔						✔			✔			✔			
	B1																	✔
Trombone	S3								✔		❍		❍					
	S5									✗	❍					✔		
	S6						✔				❍			✗	✗		✔	
Flute	S2			✗			✔		✗			✔						
	S7			✔		✗					✗		✔				✗	
	S13			✔					✔	✔	✗		✗					
Accordion	B2				✗		✔				❍							
	B4									✔	❍							
	B6		✔									❍					✗	✗
Violin	S4			✗					✔				✗					
	S15	✔	✔	✔	✔	✔	✔											
Cello	S8		✔						✗		✗	✗	✗				✔	
	S16			✔				✗		✗				❍				
Percussions	B7	✔										✔	✔				✔	
	B10										✗		❍		✔			
Guitar	S10			✔			✔		✔		✗		❍					
	B3		✗				✗				❍		✔					
	B8						✔		✔									
Vocals	B2				✗		✔				❍							
	B7	✔										✔	✔				✔	

✔ Suitable, ❍ Partially suitable, ✗ Unsuitable

orchestral participation.

Preferred Modalities

Although the focus was on haptic communication, several respondents advocated for a multimodal approach, combining tactile and auditory cues to enrich the interpretability of the information.

Wearability and Use Contexts

Most respondents expressed enthusiasm about using haptic systems in educational or rehearsal settings, but raised concerns about their practicality in live performance contexts. The interviews revealed several essential requirements for such devices to be truly useful for musicians. Comfort and unobtrusiveness were paramount, the device must be lightweight and discreet enough not to interfere with body movements or instrument handling, even during extended use. Respondents emphasized that vibrations should be clearly perceptible yet carefully calibrated to avoid creating distraction or stress through excessive strength or frequency, therefore continuous vibrations must be avoided.

Practical considerations emerged as equally important: the system should be wireless (preferably using Bluetooth) to preserve freedom of movement, and must be designed for independent use; musicians should be able to put on and remove the device without assistance, a particularly crucial requirement for BVI users. Additionally, respondents stressed that the device should be intuitive enough to use without extensive training, making it accessible to musicians regardless of their technical expertise. While participants recognized the potential value of haptic feedback systems, they consistently emphasized that successful adoption would depend on balancing functionality with the practical demands of musical performance.

11.4.3 Per-Instrument Discussion

In this section we discuss the interview outcomes, summarizing, for each instrument, the body parts that were found suitable/unsuitable for the placement of a haptic feedback device, providing motivations derived from the answers by the interviewees. Individual answers are reported in Table 11.3.

Harp (2 sighted musicians)

The two harp players mentioned that freedom of hand and finger movement is essential, particularly for actions such as hand closure and the performance of harmonics. Thus, any haptic device must not interfere with fine motor skills. Participant S1 identified wrists and the back of the hand as acceptable locations, as long as they do not restrict hand movement. Conversely, the hand was considered unsuitable by participant S9, as it may be too distracting and impede finger movements. Both S1 and S9 agreed that the forearms and upper arms are viable, provided that the system remains in place. Certain zones are unsuitable due to the posture and movement involved in harp performance, including the thighs, as identified by S1, where placement would be too intrusive, and the head and back, as identified by S9, which could be uncomfortable during seated playing. For S9, the ankles and the feet were considered suitable as they do not interfere with the instrument.

Piano (2 sighted, 4 BVI musicians)

Piano players did not find consensus on the identification of a suitable location for haptic feedback placement. Two of them (S11 and B6) considered the wrist as one of the suitable areas, as it does not interfere with hand/finger movement essential to performance. On the other hand, S12 and B5 identified the arms as acceptable, due to their proximity to the hands. However, all of them explicitly mentioned that any device placed near the hands must avoid restricting movement. The back and the belt were considered unsuitable by B5, as placing a wearable device in those areas may cause discomfort during seated playing. It is worth noting that some

interviewed piano players (B2 and B6) also play additional instruments: the areas deemed suitable by those participants hold for all instruments they play.

Trumpet (1 sighted, 1 BVI musician)

For trumpet players, the body movement while playing is quite static. None of the two participants mentioned any unsuitable area. However, depending on the playing position, some areas may be perceived as uncomfortable. For example, feet may not be recommended, as the player can either stand or sit depending on the orchestral settings, as well as knees and thighs. Here, no clear preference emerged, with S14 mentioning ankles, torso, wrists, and neck as most suitable areas, while B1 mentioned only ears.

Trombone (3 sighted musicians)

For trombone players, participants highlighted that haptic placement must account for the asymmetrical use of the arms during performance: the trombone is posed on the left shoulder, whereas the right one remains free. The right arm and right hand, which are in constant motion to operate the slide, are deemed unsuitable by all the participants. On the other hand, the left arm and left hand, which are more stationary, can be considered as a suitable location, particularly the hand or forearm. The back was considered unsuitable by S5 due to incompatibility with sitting, whereas shoulders and neck were considered unsuitable by S6, as they may interfere with supporting or manipulating the instrument.

Flute (3 sighted musicians)

For flute players, no clear preference emerged, as the participants showed contrasting opinions. S2 and S7 identified the wrist and the forearm as suitable locations for receiving haptic feedback, as long as it does not interfere with the delicate hand positioning required for playing. Conversely, S12 considered arms as too involved in the performance to receive additional complex stimuli. On the other hand, S7 and S12 considered legs as suitable, as they would not interfere with the performance. Differently, S2 considered legs (in particular thighs), as unsuitable due to the seated posture typical in orchestral settings. Other suitable positions were identified, such as the belt, chest, and back. However, S12 explicitly mentioned during the interview that those are valid only if vibrations remain very subtle, so as not to interfere with the playing.

Accordion (3 BVI musicians)

All accordion players agree on the right arm and the back of the right hand as suitable positions for haptic placement, as playing mainly involves fingers only,

whereas the hand position remains relatively still. In contrast, the left hand is too involved in controlling the bellows and thus requires full freedom of movement, making it unsuitable for the placement of haptic devices. Placement near the belt or the back is also considered viable by some of the players, since it does not obstruct the instrument. Instead, B2 considered the knees as not recommended, particularly when playing standing up, as physical tension in that area may reduce sensitivity. Additionally, B6 excluded both head and ears, as placing the haptic device on those areas could be too distracting during the performance.

Violin (2 sighted musicians)

Among violinists, S4 considered the central area of the chest as a possible location for haptic feedback, as long as the device does not interfere with movement. Conversely, she mentioned that arms and shoulders must remain entirely free to allow for the precise movements required in playing, making these areas unsuitable. S4 also considered the legs as unsuitable, since they could cause imbalance or discomfort during performance. On the other hand, S15 considered the whole bottom part of the body as suitable, since it is less involved in the performance as more static, as long as placement of the haptic feedback device does not interfere with sitting.

Cello (2 sighted musicians)

Cello players agreed that haptic feedback is best suited on the lower part of the body, with ankles and legs identified as compatible areas that do not interfere with performance. Conversely, they considered vibrations in the upper body, particularly the arms, chest, wrists, and hands, as highly disruptive, since these areas are actively engaged in playing. The participants reported difficulty when playing while wearing a watch on their wrist, highlighting the need to keep the upper limbs completely free. The chest and back were also considered unsuitable areas, since haptic feedback placement would interfere with breathing and posture. S16 mentioned shoulders as potentially suitable areas, but clarified that they could also be problematic due to their involvement in instrument support.

Percussion (2 BVI musicians)

For percussionists, both hands are involved in the performance, with different movement amplitudes and speeds, thus they were considered unsuitable by B10. The arms, in particular the left forearm, were considered as a suitable location by both participants. However, they mentioned that, if placed in this location, the device should be designed to avoid restrictions of the movements of the arms.

Guitar (1 sighted, 2 BVI musicians)

For guitar players, haptic feedback is best placed on the fretting arm's forearm (typically the left arm for right-handed players), as indicated by S12, since it remains free during performance and the device would not interfere with playing. In contrast, the strumming arm rests against the guitar body, making it unsuitable due to the risk of transferring unwanted vibrations. The hands and fingers are also deemed unsuitable, as they are essential for precision and tactile sensitivity, with only potentially the back of the hands being a valid location, as suggested by B3. The chest/torso may be compatible, though its proximity to the instrument could affect its usability. Hips or waistline are mentioned as possible locations when seated, while the legs may be less suitable due to reduced sensitivity.

Vocals (2 BVI musicians)

For singers, the belt area (especially on the side) is considered by B2 to be a favorable location for haptic feedback, as it does not interfere with vocal performance. However, vibrations should not occur during singing to avoid disruption. The arms and feet are also identified as acceptable by B7, though the usability of such parts would heavily depend on the device's design. The knees are considered unsuitable by B2, particularly when standing, due to physical tension that reduces sensitivity.

11.4.4 Final Observations

From the interviews it emerged that, while for some instruments players had clear consensus on preferred body areas for device placement, in numerous instances the choice of body parts depended strongly on individual user preference. For musicians who play multiple instruments, versatility also influenced their decisions, as they often favored locations that could work across different instruments. This suggests that the device should be designed to be easily customizable and adaptable to various body parts.

Another aspect to consider is that there are several body parts for which the participants did not specify their opinion on suitability. These unexplored areas may still hold potential for device placement. They could be evaluated based on both practical convenience, as suggested in prior studies [254, 193], and tactile acuity, with preference given to regions more sensitive to vibrational stimuli [148].

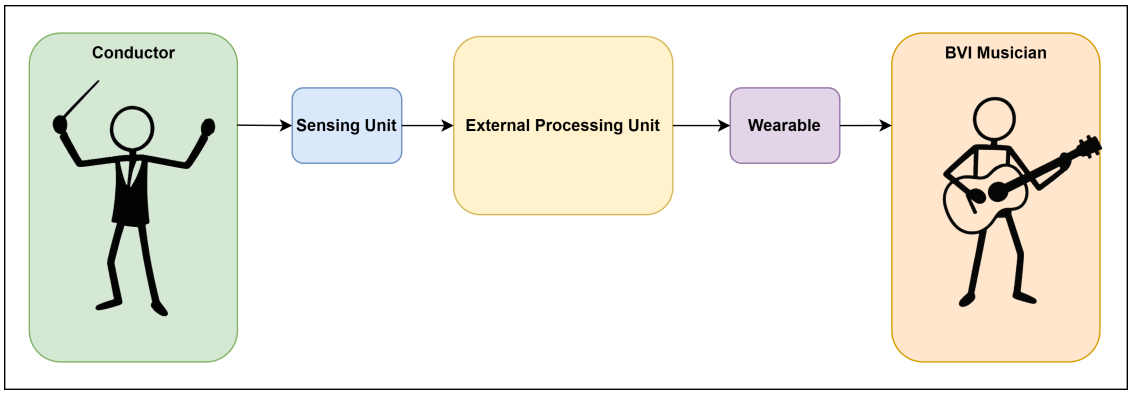


Figure 11.2: The image illustrates a conceptual framework where the modular system architecture can take place. The conductor's gestures are first captured through a sensing unit and transmitted to an external processing unit. There, gesture recognition algorithms interpret the input and generate control messages. These messages are then encoded and sent wirelessly to a wearable device worn by the BVI musician, which delivers corresponding tactons to support synchronized musical interaction.

11.5 From Requirements to Specifications for Prototype Design

Based on the insights gathered from interviews with musicians and an analysis of the literature regarding orchestra conduction and haptic systems, this section explores a range of potential design solutions to translate conductor gestures into tactons, enabling BVI musicians to perceive and respond to conducting cues in real-time.

A set of requirements is therefore defined to provide a foundation for the system's technical development and to support its future implementation and validation. For each requirement, multiple design options are presented and analyzed in terms of advantages and limitations. The exploration of possible design solutions is guided by three core functional requirements:

- Motion data acquisition and processing;
- Gesture-to-tacton feedback via wearable devices;
- User customization.

These functional requirements are considered alongside a set of non-functional requirements essential for real-time and wearable use. In particular, system latency must be minimized to ensure effective temporal alignment with musical gestures.

Moreover, since the system involves both a conductor and a musician, the physical footprint of any body-worn components must be kept as small as possible for both roles. This ensures that the conductor's movement remains unobstructed and that the musician's comfort is preserved during performance. Additionally, the power consumption of the wearable unit should be optimized to support portable, untethered operation over extended periods.

To accommodate these requirements, the design space points toward a modular architecture (see Figure 11.2). This comprises an external processing unit, which handles gesture acquisition and interpretation, and a wearable feedback device, which receives control messages and delivers tactile stimulation. This division allows for powerful and flexible processing without compromising the size, comfort, or energy efficiency of the wearable device.

11.5.1 Motion Data Acquisition and Processing

The first requirement addresses the acquisition and interpretation of conductor gestures in real time, with the goal of translating them into structured tactons that can be perceived and understood by BVI musicians. This process involves two main stages: the acquisition of gesture data through appropriate sensing technologies, and the subsequent processing and classification of those signals into meaningful control commands.

Two sensing approaches are particularly relevant in this context, each offering distinct trade-offs. The first option involves the use of wearable inertial sensors, such as Inertial Measurement Units (IMUs), which can be embedded in gloves, wristbands, armbands, or mounted on a lightweight headpiece worn by the conductor [65]. In the proposed setup, these sensors are connected to an external processing unit, where raw motion data is transmitted for analysis. This configuration keeps the sensors extremely lightweight and power-efficient, and minimizes local processing requirements, which aligns well with the design constraints of low energy consumption and reduced wearable footprint. On the other hand, requiring the conductor to wear multiple IMUs can introduce physical encumbrance and may interfere with natural expressivity, particularly in performance settings. The presence of cables also demands careful setup to avoid restricting movement.

Alternatively, gesture acquisition can be performed using stereo RGB-D cameras, placed on a stationary stand positioned in front of the conductor. This configuration eliminates the need for body-worn sensing hardware altogether, allowing for natural, unrestricted motion. The camera captures detailed spatial and temporal information about the conductor's movements and streams this data directly to the processing unit. While this setup supports vision-based recognition pipelines and avoids distributing electronics on the conductor's body, it introduces higher latency compared to inertial sensors. In particular, RGB-D cameras typically exhibit end-to-end latencies in the range of 25–35 ms due to frame acquisition [111],

internal processing, and data transfer delays, whereas IMUs can achieve latencies well below 10 ms [10]. Moreover, the functional reliability of camera-based systems is influenced by environmental factors such as lighting and visual occlusion, which may affect the quality of gesture capture.

Once gestural data has been acquired, it must be processed to extract discrete symbolic information suitable for controlling haptic feedback. This extraction of symbolic information is crucial to avoid a direct, continuous mapping of conductor movements into vibrations, which, as discussed in Sec. 11.4.2, would create the distracting continuous stimulation that musicians seek to avoid. Instead, the processing stage must intelligently interpret the conductor's gestural intent, translating meaningful musical cues into discrete tactile events. The nature of this processing stage depends on the chosen sensing modality.

When using wearable IMUs, traditional signal processing techniques are typically sufficient for detecting well-defined gestures with limited variability. These include time-series segmentation, thresholding, and classification based on handcrafted features such as peak acceleration, angular velocity, or trajectory patterns. Algorithms such as Dynamic Time Warping (DTW) [97, 141] have been successfully used to recognize rhythmic and temporal cues with low computational overhead, which is particularly beneficial for real-time applications. However, when greater generalization or robustness is required, machine learning approaches can be employed to process IMU data. Since computation is handled off-board, such models can be implemented without impacting the wearable's energy efficiency.

Unlike IMUs, data from RGB-D streams provide a richer and more detailed representation of both movements and facial expressions, making them well-suited for advanced deep learning techniques for multimodal recognition, such as combining gesture recognition and emotion recognition. To identify conducting gestures in real time, adaptations of state-of-the-art techniques, such as those described in [206, 43, 87, 267], may be adopted. These models can capture complex spatio-temporal dynamics and are more robust to variation across users or environments. While such methods require more computational resources, they are compatible with the proposed architecture, where the processing is performed on an external general-purpose computing unit.

In both sensing scenarios, the output of the processing stage is a compact symbolic representation of the detected gesture, such as a cue identifier, tempo value, or intensity level, which is subsequently transmitted to the wearable system for real-time vibrotactile feedback.

11.5.2 Real-Time Gesture-to-Tacton Feedback via Wearable Devices

To enable real-time tactile feedback in response to conductor gestures, the system requires a suitable communication protocol and a wearable embedding a processing

Table 11.4: Comparison of Design Alternatives for the Haptic Feedback System

Requirement	**Aspect**	**Component**	**Advantages**	**Limitations**
Motion data acquisition and processing	Sensing modality	IMU	Low latency (<10 ms); precise motion tracking	Requires wearable sensors; may constrain gestures
		RGB-D Camera	No need for worn sensors; natural conductor movement	High latency (25–35 ms); sensitive to lighting
	External processing unit	General-purpose PC	Balanced performance and cost; supports ML pipelines	Lower latency requires more specialized and costly hardware
Gesture-to-tacton feedback via wearable devices	Communication protocol	BLE	Wireless; low latency (∼10 ms); low power consumption; widely supported	Higher latency than wired options (e.g., USB)
	Wearable	MCU	Good performance, easy to integrate, compact	Limited computational capacity; low flexibility
		SoC / FPGA	High performance and flexibility	High power consumption; complex; large size
	Actuator type	LRA	Good trade-off in terms of response time (∼25 ms) and power consumption	Fixed frequency; limited expressiveness
		ERM	Low cost	Slow response time(>100 ms); high power consumption
		Piezoelectric	Fast response time (1 to 4 ms); low power consumption	Needs high-voltage drivers; more complex integration
User customization (Conductor and BVI Musicians)	Configuration method	Predefined profiles	Easy to activate; performance-ready	Requires external UI; Limited flexibility
		Adjustable parameters	High personalization (intensity, timing, mapping)	Requires external UI; more setup effort

unit able to drive the actuators delivering distinct and timely vibrotactile patterns to the musician.

The first step involves the transmission of the compact, encoded message representing the detected gesture (e.g., tempo, beat intensity, or cue type) from the external processing unit to the wearable system. For this purpose, Bluetooth Low Energy (BLE) is a practical choice due to its wireless operation, low power consumption, sufficient data rate for control signals, and widespread hardware support in embedded platforms. BLE provides adequate responsiveness, typically 10 ms latency [230], and simplifies integration with consumer-grade microcontrollers.

Once received, the message is handled by the wearable. Among the possible technologies such as Field-Programmable Gate Array (FPGA), System on Chip (SoC), or Microcontroller Units (MCUs), the most feasible choice is an MCU. Compared to FPGA or SoC-based solutions, MCUs offer a superior trade-off between computational capability and system complexity. Alternative architectures such as FPGAs and SoCs can offer higher performance and flexibility. However, their increased power consumption, development complexity, and larger physical size make them poorly suited for wearable applications, where compactness and energy efficiency are critical. Microcontrollers, on the other hand, are well-suited to the task.

The final component is the actuator, which physically conveys the tacton to the user. Several technologies are available, each offering distinct trade-offs in terms

of responsiveness and power consumption. These characteristics have been extensively compared in recent literature, which reports typical rise times and integration constraints for common actuator classes used in haptic systems [44].

Linear Resonant Actuators (LRA) produce clean vibrations at a fixed resonant frequency and typically offer fast response times (around 25 ms), combined with relatively low power usage. Due to their compact size and consistent performance, they are well suited for integration into wearable elements such as straps, bands, or garments, where space and power constraints are critical. However, their fixed operating frequency can limit the range of tactile effects that can be delivered.

Eccentric Rotating Mass (ERM) motors represent a cost-effective and widely adopted alternative. However, their mechanical structure leads to significantly slower response times, often exceeding 100 ms, and less precise control of vibration onsets. As a result, they are generally less suitable for applications requiring timely and well-defined cues, such as beat alignment in musical performance.

Piezoelectric actuators provide the fastest response among the three technologies, with rise times typically ranging from 1 to 4 milliseconds. They are capable of generating highly detailed and sharp tactile sensations with a wide bandwidth. Nevertheless, these actuators require high-voltage drivers and more sophisticated control electronics, which can limit their practical deployment in low-power and compact wearable systems, especially where safety and energy efficiency are concerns.

Taking all constraints into account, including responsiveness, power budget, and physical integration, the combination of BLE communication, a low-power MCU, and LRA actuators represents a balanced solution for real-time, body-worn tacton delivery.

11.5.3 User Customization

To guarantee effectiveness of use, the proposed system design also requires a degree of customization for both the conductor and the BVI musicians. For the conductor, customization is necessary at the gesture recognition stage, since conducting styles vary widely across individuals and the system must allow adaptation of mapped gestures, interpretation rules, and detection thresholds in order to preserve each conductor's personal expressivity. For the BVI musicians, customization is required at the haptic feedback stage, since preferences in tactons intensity, duration, or actuators mapping differ according to instrument, tactile sensitivity, and performance context.

For both user types, two complementary levels of customization can be foreseen. Predefined profiles would allow for a quick setup through ready-to-use modes optimized for typical conducting styles or for specific instruments, minimizing configuration effort during rehearsals and performances. Adjustable parameters, on

the other hand, would provide deeper personalization, enabling conductors to fine-tune gesture mapping and BVI musicians to refine vibrotactile feedback according to individual needs.

In all cases, customization requires the support of an external user interface, which should remain simple enough to be managed by conductors or orchestra staff without specialized technical skills.

The design options discussed across the three subsections are summarized in Table 11.4, which provides a comparative overview of key components, trade-offs, and suitability for wearable haptic feedback systems.

Chapter 12

Conclusions and Future Work

This thesis investigated whether NMP systems could meet professional audio requirements while remaining accessible to non-technical users. The research pursued this question along three complementary directions: system design, network resilience, and accessibility. While each contribution addresses distinct technical problems, they collectively advance a unified vision: making high-quality NMP accessible to diverse users across varied technical and physical circumstances. This concluding chapter synthesizes the main findings, reflects on their implications, and outlines directions for future research.

12.1 Contributions and Reflection

12.1.1 System Performance and Network Resilience

A central challenge in NMP system design is the apparent trade-off between technical performance and ease of use. Existing systems prior to this work generally fell into two categories: professional tools offering low latency and high audio quality but requiring substantial technical expertise, and consumer videoconferencing applications offering simplicity but inadequate performance for musical interaction. The work presented in Chapters 4 and 5 demonstrates that this dichotomy is not inevitable. Chapter 4 articulated a vision for combining web accessibility with professional audio performance, while Chapter 5 showed the MEVO platform and its deployment in actual performance contexts. By implementing a web interface familiar to any browser user alongside native audio handling that meets professional requirements, the system achieves both accessibility and performance. The key architectural insight is the separation of concerns: the user interface layer prioritizes simplicity while the audio transport layer prioritizes latency, allowing

each to be optimized independently. The distributed concert connecting musicians in Turin and Wrocław demonstrated that these technologies can support actual musical performance across more than a thousand kilometers, not merely laboratory experiments. Building on this systems-level approach, Chapter 3 explored the fundamental limits of local processing delay when general-purpose computing is abandoned in favor of dedicated hardware. The FPGA-based coprocessor, built on a Transport Triggered Architecture, achieved sub-millisecond audio processing latency, confirming that specialized hardware can reserve nearly the entire tolerable delay budget for network traversal. This work establishes a performance ceiling that informs realistic expectations for what software-only approaches can achieve. However, achieving low local latency is only part of the challenge. Real-world networks introduce impairments such as packet loss or jitter, that laboratory conditions may not reveal. At the same time commercial hardware is subject to imperfections, like clock drift that must be considered during the design phase. To address this gap, Chapter 6 introduced DUST, a dataset of UDP audio traces collected with MEVO under varied network conditions. A distinguishing feature is the inclusion of local playback status information alongside packet arrival timestamps, enabling researchers to simulate scenarios where late packets still contain partially usable audio data. The dataset provides a foundation for studying network phenomena under realistic conditions, capturing not only packet arrival patterns but also the playback system state that determines whether late packets retain utility. When network losses do occur, the quality of the musical experience depends critically on how missing audio is reconstructed. Chapter 8 addressed this challenge through a sparse linear prediction algorithm that demonstrates effective mitigation need not require extensive computational resources. By exploiting the periodic structure common in musical signals, the algorithm achieves reconstruction quality comparable to traditional autoregressive models while running in sub-millisecond time on ARM hardware. This efficiency enables deployment on resource-constrained devices such as the Raspberry Pi, broadening both the scalability and the range of hardware that can support high-quality NMP. While the preceding chapters focused on audio transmission, many musical interactions also rely on symbolic control data. Chapter 7 laid the groundwork for managing such data in NMP contexts. The evaluation of recovery journal-based techniques for MIDI streaming over Real-Time Protocol quantified the trade-offs between bandwidth overhead and resilience to packet loss, providing guidance for system designers who must balance these competing concerns.

12.1.2 Accessibility and Inclusive Design

Traditional NMP systems implicitly assume users who can see, hear, and operate conventional interfaces. The accessibility-focused contributions of this thesis challenge these assumptions, demonstrating that the design space for NMP extends

beyond audio transmission to encompass the full range of communicative modalities that musicians employ. Chapter 9 offered a comprehensive overview of technological solutions that could address the needs of musicians with visual, auditory, or mobility impairments in remote music education and performance contexts. The chapter envisioned the integration of haptic feedback, motion tracking, and immersive audio rendering within NMP frameworks, identifying the technical challenges that must be tackled to realize an inclusive musical environment. Moving from vision to concrete design, Chapter 11 explored how haptic feedback can convey conducting gestures to blind and visually impaired musicians. Through interviews with sighted and visually impaired performers, design specifications were derived for wearable devices that translate conductor movements into vibrotactile patterns. A critical insight emerged from these interviews: device placement must vary according to instrument-specific constraints to ensure that the feedback does not interfere with playing technique. This attention to embodied musical practice distinguishes the approach from generic assistive technologies that ignore the physical demands of instrumental performance.

12.1.3 Gestural Communication Beyond Audio

Chapter 10 investigated Virtual Reality as a means to transmit conductor gestures to remote musicians. By tracking hand movements and head orientation through a VR headset and rendering them as an avatar, the system restores visual communication without the bandwidth costs of video streaming, achieving substantially lower bandwidth requirements while maintaining acceptable Motion-to-Photon latency. This approach addresses a fundamental limitation of traditional NMP systems, which focus on audio transmission while neglecting the visual cues essential to orchestral coordination.

The VR-based conducting system and haptic feedback specifications represent initial explorations rather than complete solutions. The VR system requires validation with actual conductors and orchestras in rehearsal contexts; the haptic specifications have motivated prototype development and initial user testing. Nevertheless, these contributions establish that NMP can accommodate diverse sensory modalities and physical abilities, expanding the boundaries of who can participate in distributed musical experiences.

12.2 Future Directions and Ongoing Work

The research presented in this thesis opens several avenues for continued investigation.

12.2.1 Packet Loss Concealment

The sparse linear prediction algorithm was evaluated primarily through objective metrics. Extensive subjective testing with musicians and expert listeners would validate its perceptual effectiveness across different musical genres, instruments, and loss patterns. Such evaluation should include comparison with deep learning-based approaches to characterize the trade-off frontier between quality and efficiency.

The algorithm's hyperparameter such as the model order or the lag selection strategy, were chosen through systematic but limited exploration. Adaptive approaches that adjust parameters based on signal characteristics or detected loss patterns could improve robustness across diverse musical material.

12.2.2 Dataset Expansion and Buffer Management

The DUST dataset currently comprises traces collected over wired network connections. WiFi in home and venue environments, mobile networks (especially 5G, which promise low latency and may be suitable for NMP) introduce distinct impairment patterns that the current dataset does not capture. Expanding DUST to include wireless traces would enable research on NMP under the connectivity conditions that many users actually experience.

Beyond dataset expansion, the collected traces enable research into automated receiver buffer management strategies. The primary objective of DUST was to provide realistic loss traces for testing packet loss concealment methods, but the inclusion of playback status information opens the possibility of developing adaptive algorithms that dynamically adjust receiver buffer sizes. Such algorithms could intelligently navigate the trade-off between latency and robustness by analyzing network jitter patterns and clock drift characteristics in real time, adapting to time-varying network conditions without requiring manual buffer configuration. This would address both jitter and drift challenges simultaneously, making NMP systems more robust across diverse network environments.

The dataset also captures complex temporal behaviors like network conditions that change over time, non-constant clock drift, and non-linear relationships between sender and receiver audio clocks, that highlight the complexity of required statistical modeling. Future work could explore traffic prediction algorithms or time-series forecasting methods to model these dynamics, potentially enabling proactive buffer adjustment before degradation becomes audible.

12.2.3 VR Conducting System

The VR-based conducting system requires validation in actual rehearsal contexts with professional conductors and ensembles. Such evaluation would reveal how residual latency affects musical coordination and whether musicians can reliably interpret avatar-mediated gestures.

Alternative approaches to gesture capture merit exploration alongside VR-based tracking. Camera-based systems using pose estimation frameworks such as MediaPipe could provide conductor tracking without requiring specialized headsets, potentially lowering barriers to adoption. RGB-D cameras offer depth information that could improve tracking robustness under varied lighting conditions, while Inertial Measurement Units (IMUs) attached to the conductor's hands or baton could provide high-frequency motion data with minimal computational overhead. Each approach presents distinct trade-offs between cost, accuracy, user comfort, and deployment complexity that warrant systematic comparison.

Integration with audio streaming presents both technical and perceptual challenges. Synchronizing gesture display with audio playback under varying network conditions requires careful buffer management. The current system captures hand and head movement but not facial expression; face tracking capabilities in newer VR hardware could enable richer avatar representation.

A recent Turin-Wrocław concert provided an opportunity to deploy the VR conducting system in a real performance context [223]. Feedback from this experience has been collected and awaits systematic analysis. This evaluation will inform refinements to the tracking pipeline, avatar rendering, and synchronization mechanisms based on the actual needs of conductors and musicians in distributed performance settings.

12.2.4 Haptic Feedback Development

The specifications derived from musician interviews in Chapter 11 provided a foundation for prototype development. A subsequent study, not included in this thesis, implemented and assessed these specifications through a wearable haptic device incorporating five vibrating motors arranged in a cross-shaped constellation on a neoprene sleeve. The prototype was evaluated through user studies assessing spatial localization, intensity discrimination, and interpretation of complex vibrotactile patterns corresponding to conducting gestures such as entrances, cut-offs, beats, and dynamic changes. Results demonstrated that participants could reliably distinguish spatial locations and intensities of vibrotactile stimuli, and achieved above-chance recognition accuracy for gesture-based patterns after brief familiarization, confirming the viability of the proposed approach.

Further development will focus on studies with blind and visually impaired musicians in actual performance contexts, which would reveal requirements not apparent from controlled experiments. Iterative design based on user feedback could refine the mapping of conducting gestures to tactile patterns and optimize actuator placement for different instruments.

Bibliography

[1] *A high quality low-complexity algorithm for packet loss concealment with G.711*. ITU-T Rec. G.711 Appendix I. Sept. 1999.

[2] D. Abdlkarim et al. "A methodological framework to assess the accuracy of virtual reality hand-tracking systems: a case study with the Meta Quest 2". In: *Behavior Research Methods* 56.2 (Feb. 2024), pp. 1052–1063. ISSN: 1554-3528. DOI: 10.3758/s13428-022-02051-8.

[3] J. Abramo. "Disability in the classroom: Current trends and impacts on music education". In: *Music Educators Journal* 99.1 (2012), pp. 39–45. DOI: 10.1177/0027432112448824.

[4] Acoustical Society of America. *Sound*. Accessed: 2025-12-14. 2025. URL: https://asastandards.org/terms/sound-2/.

[5] P. Adenot and H. Choi. *Web Audio API*. W3C. June 2020. URL: https://www.w3.org/TR/webaudio-1.1/.

[6] A. Afonso-Jaco and B. F. G. Katz. "Spatial knowledge via auditory information for blind individuals: spatial cognition studies and the use of audio-VR". In: *Sensors* 22.13 (June 2022). ISSN: 1424-8220. DOI: 10.3390/s22134794.

[7] C. Alexandraki et al. "Towards the implementation of a generic platform for networked music performance: the DIAMOUSES approach" (Aug. 2008).

[8] M. Allman, V. Paxson, and E. Blanton. *RFC 5681: TCP congestion control*. 2009.

[9] H. T. Alvestrand. *Overview: real-time protocols for browser-based applications*. RFC 8825. Jan. 2021. DOI: 10.17487/RFC8825. URL: https://www.rfc-editor.org/info/rfc8825.

[10] Analog Devices, Inc. *ADIS16507: precision, small form factor, six degrees of freedom inertial sensor*. Analog Devices. URL: https://www.analog.com/media/en/technical-documentation/data-sheets/ADIS16507.pdf.

[11] D. Antón et al. "Real-time communication for Kinect-based telerehabilitation". In: *Future Generation Computer Systems* 75 (Oct. 2017), pp. 72–81. ISSN: 0167739X. DOI: 10.1016/j.future.2017.05.006.

[12] A. Arfaoui et al. “Designing interactive and immersive multimodal installations for people with disability”. In: *Virtual Reality and Its Application in Education*. Ed. by D. Cvetković. IntechOpen, Jan. 2021. DOI: 10.5772/intechopen.90678.

[13] *Arty A7 Reference Manual*. [Accessed 13-10-2023]. URL: https://digilent.com/reference/programmable-logic/arty-a7/reference-manual.

[14] X. Bai. “Remote music learning based on wireless sensors supporting 6G and CPS”. In: *Wireless Personal Communications* (May 2024). ISSN: 0929-6212, 1572-834X. DOI: 10.1007/s11277-024-11147-7.

[15] J. Bajo et al. “A distributed architecture for facilitating the integration of blind musicians in symphonic orchestras”. In: *Expert Systems with Applications* 37.12 (Dec. 2010), pp. 8508–8515. ISSN: 09574174. DOI: 10.1016/j.eswa.2010.05.025.

[16] G. Baltas and G. Xylomenos. “Ultra low delay switching for networked music performance”. In: *IISA 2014, The 5th International Conference on Information, Intelligence, Systems and Applications*. IEEE, July 2014, pp. 70–74. DOI: 10.1109/IISA.2014.6878798.

[17] A. Barmpoutis et al. “Assessing the role of virtual reality with passive haptics in music conductor education: a pilot study”. In: *Virtual, Augmented and Mixed Reality. Design and Interaction*. Ed. by J. Y. C. Chen and G. Fragomeni. Vol. 12190. Springer International Publishing, 2020, pp. 275–285. DOI: 10.1007/978-3-030-49695-1_18.

[18] R. Battello et al. “Experimenting With Adaptive Metronomes in Networked Music Performances!” In: *Journal of the Audio Engineering Society* 69.10 (Oct. 2021), pp. 737–747. ISSN: 15494950. DOI: 10.17743/jaes.2021.0034.

[19] *BCM2711 datasheet*. [Accessed 13-10-2025]. URL: https://datasheets.raspberrypi.com/bcm2711/bcm2711-peripherals.pdf.

[20] D. Bert. “FPGA-based implementation of audio effects for ultralow-latency Networked Music Performance applications”. MA thesis. Politecnico di Torino, 2023.

[21] D. Bert et al. “FPGA-based Low-Latency Audio Coprocessor for Networked Music Performance”. In: *2023 4th International Symposium on the Internet of Sounds*. IEEE, Oct. 2023, pp. 1–8. DOI: 10.1109/IEEECONF59510.2023.10335333.

[22] F. Berthaut. “3D interaction techniques for musical expression”. In: *Journal of New Music Research* 49.1 (Jan. 2020), pp. 60–72. ISSN: 0929-8215, 1744-5027. DOI: 10.1080/09298215.2019.1706584.

[23] Bialystok University of Technology and K. Tomaszewska. "XR technology in manufacturing – exploring of practical applications". In: *Scientific Papers of Silesian University of Technology. Organization and Management Series* 2025.217 (2025), pp. 573–591. ISSN: 16413466. DOI: 10.29119/1641-3466.2025.217.37.

[24] J. R. Blum, M. Bouchard, and J. R. Cooperstock. "What's around me? spatialized audio augmented reality for blind users with a smartphone". In: *Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Ed. by A. Puiatti and T. Gu. Vol. 104. Springer Berlin Heidelberg, 2012, pp. 49–62. DOI: 10.1007/978-3-642-30973-1_5.

[25] A. Boem and L. Turchet. "Musical Metaverse Playgrounds: Exploring the Design of Shared Virtual Sonic Experiences on Web Browsers". In: *2023 4th International Symposium on the Internet of Sounds*. IEEE, Oct. 2023, pp. 1–9. DOI: 10.1109/IEEECONF59510.2023.10335297.

[26] M. Bosi et al. "Experiencing Remote Classical Music Performance Over Long Distance: A JackTrip Concert Between Two Continents During the Pandemic". In: *Journal of the Audio Engineering Society* 69.12 (Dec. 2021), pp. 934–945. ISSN: 15494950. DOI: 10.17743/jaes.2021.0056.

[27] H. Boström, J.-I. Bruaroey, and C. Jennings. *WebRTC 1.0: real-time communication between browsers*. W3C. Dec. 2020. URL: https://www.w3.org/TR/webrtc/.

[28] S. Brewster and L. M. Brown. "Tactons: structured tactile messages for non-visual information display". In: *Proceedings of the 5th Australasian User Interface Conference (AUIC 2004)*. Ed. by A. Cockburn. Vol. 28. CRPIT. Australian Computer Society, 2004, pp. 15–23.

[29] J.-P. Cáceres and C. Chafe. "JackTrip: Under the Hood of an Engine for Network Audio". In: *Journal of New Music Research* 39.3 (Nov. 2010), pp. 183–187. DOI: 10.1080/09298215.2010.481361.

[30] J.-P. Cáceres et al. "To the edge with China: explorations in network performance". In: *ARTECH 2008: Proceedings of the 4th International Conference on Digital Arts*. ARTECH US. 2008, pp. 61–66.

[31] P. Cairns, H. Daffern, and G. Kearney. "Parametric evaluation of ensemble vocal performance using an immersive network music performance audio system". In: *Journal of the Audio Engineering Society* 69.12 (Dec. 2021), pp. 924–933. ISSN: 15494950. DOI: 10.17743/jaes.2021.0040.

[32] P. Cairns et al. "Evaluation of Metaverse Music Performance With BBC Maida Vale Recording Studios". In: *Journal of the Audio Engineering Society* 71.6 (June 2023), pp. 313–325. ISSN: 15494950. DOI: 10.17743/jaes.2022.0086.

[33] D. Calderón-Garrido and J. Gustems-Carnicer. "Adaptations of music education in primary and secondary school due to COVID-19: the experience in Spain". In: *Music Education Research* 23.2 (Mar. 2021), pp. 139–150. ISSN: 1461-3808, 1469-9893. DOI: 10.1080/14613808.2021.1902488.

[34] A. Carôt and J. Reizner. "A telematic approach for mass music ensembles". In: *Proc. Re-New Conf. Digit. Arts Festival*. 2013, pp. 170–174.

[35] A. Carôt and G. Schuller. "Towards a telematic visual-conducting system". In: *Audio Engineering Society Conference: 44th International Conference: Audio Networking*. Nov. 2011. URL: https://www.aes.org/e-lib/browse.cfm?elib=16137.

[36] A. Carôt and C. Werner. "Distributed network music workshop with Soundjack". In: *Proceedings of the 25th Tonmeistertagung*. 2008.

[37] A. Carôt and C. Werner. "Fundamentals and principles of musical telepresence". In: *Journal of Science and Technology of the Arts* 1.1 (May 2009), pp. 26–37. DOI: 10.7559/citarj.v1i1.6.

[38] D. Carvalho, J. Barroso, and T. Rocha. "Multisensory experience for people with hearing loss: a preliminary study using haptic interfaces to sense music". In: *HCI International 2022 – Late Breaking Papers: HCI for Health, Well-being, Universal Access and Healthy Aging*. Ed. by V. G. Duffy et al. Vol. 13521. Springer Nature Switzerland, 2022, pp. 292–306. DOI: 10.1007/978-3-031-17902-0_21.

[39] F. Catania et al. "Boris: a spoken conversational agent for music production for people with motor disabilities". In: *CHItaly 2021: 14th Biannual Conference of the Italian SIGCHI Chapter*. ACM, July 2021, pp. 1–5. DOI: 10.1145/3464385.3464713.

[40] M. Centenaro, P. Casari, and L. Turchet. "Towards a 5G communication architecture for the Internet of Musical Things". In: *2020 27th Conference of Open Innovations Association (FRUCT)*. IEEE, Sept. 2020, pp. 38–45. DOI: 10.23919/FRUCT49677.2020.9210980.

[41] C. Chafe and S. Oshiro. "Jacktrip on Raspberry Pi". In: *Proceedings of the 17th Linux Audio Conference (LAC-19), CCRMA, Stanford University, USA*. 2019.

[42] J. Chappell. *SonoBus*. https://sonobus.net/. Accessed: 15-07-2023.

[43] G.-F. Chen. "Recognition of Beat-Motion Gestures of Orchestra Conductor Using DTW and Nearest Neighbor Method". In: *Proceedings of the 2023 3rd International Conference on Artificial Intelligence, Automation and Algorithms*. ACM, July 2023, pp. 59–65. DOI: 10.1145/3611450.3611459.

[44] J. Chen, E. H. T. Teo, and K. Yao. "Electromechanical actuators for haptic feedback with fingertip contact". In: *Actuators* 12.3 (Feb. 2023). ISSN: 2076-0825. DOI: 10.3390/act12030104.

[45] W. Chen et al. "A Survey on Hand Pose Estimation with Wearable Sensors and Computer-Vision-Based Methods". In: *Sensors* 20.4 (Feb. 2020). ISSN: 1424-8220. DOI: 10.3390/s20041074.

[46] Y. Chen et al. "Toward Low-Latency Cross-Modal Communication: A Flexible Prediction Scheme". In: *IEEE Transactions on Mobile Computing* 23.12 (Dec. 2024), pp. 13310–13324. ISSN: 1536-1233, 1558-0660, 2161-9875. DOI: 10.1109/TMC.2024.3425733.

[47] Y. Chen et al. "Wearable actuators: an overview". In: *Textiles* 1.2 (Aug. 2021), pp. 283–321. ISSN: 2673-7248. DOI: 10.3390/textiles1020015.

[48] M. Cognolato, M. Atzori, and H. Müller. "Head-mounted eye gaze tracking devices: an overview of modern devices and recent advances". In: *Journal of Rehabilitation and Assistive Technologies Engineering* 5 (Jan. 2018). ISSN: 2055-6683. DOI: 10.1177/2055668318773991.

[49] A. Collins and R. Halverson. *Rethinking education in the age of technology: the digital revolution and schooling in America*. Teachers College Press, 2018.

[50] R. G. Crespo, J. P. Espada, and D. Burgos. "Social4all: definition of specific adaptations in web applications to improve accessibility". In: *Computer Standards & Interfaces* 48 (Nov. 2016), pp. 1–9. ISSN: 09205489. DOI: 10.1016/j.csi.2016.04.001.

[51] CultureHub. *LiveLab*. URL: https://www.culturehub.org/livelab.

[52] J. C. Da Silva Junior and W. Germanovix. "Development of a sip-and-puff interface for communication and control of devices". In: *VIII Latin American Conference on Biomedical Engineering and XLII National Conference on Biomedical Engineering*. Ed. by C. A. González Díaz et al. Vol. 75. Springer International Publishing, 2020, pp. 1137–1146. DOI: 10.1007/978-3-030-30648-9_148.

[53] A.-A. Darrow. "Music and the hearing impaired: a review of the research with implications for music educators". In: *Update: Applications of Research in Music Education* 7.2 (Mar. 1989), pp. 10–12. ISSN: 8755-1233, 1945-0109. DOI: 10.1177/875512338900700205.

[54] D. S. E. Deering and B. Hinden. *Internet Protocol, Version 6 (IPv6) Specification*. RFC 8200. July 2017. DOI: 10.17487/RFC8200. URL: https://www.rfc-editor.org/info/rfc8200.

[55] J. A. Deja et al. "Vitune: A visualizer tool to allow the deaf and hard of hearing to see music with their eyes". In: *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Apr. 2020, pp. 1–8. DOI: 10.1145/3334480.3383046.

[56] L. Diener et al. *The ICASSP 2024 Audio Deep Packet Loss Concealment Challenge*. 2024. DOI: 10.48550/ARXIV.2402.16927.

[57] N. Domini. "Ultralow latency audio processing via FPGA for Networked Music Performance applications". MA thesis. Politecnico di Torino, 2022.

[58] C. Drioli and N. Buso. "Networked performances and natural interaction via LOLA: low latency high quality A/V streaming system". In: *Lecture Notes in Computer Science* 7990 (Jan. 2013), pp. 240–250. DOI: 10.1007/978-3-642-40050-6_21.

[59] J. Dürre et al. "In-depth latency and reliability analysis of a networked music performance over public 5G infrastructure". In: *Audio Engineering Society Convention 153*. Audio Engineering Society. 2022.

[60] D. Dziwis and H. von Coler. "The entanglement: volumetric music performances in a virtual metaverse environment". In: *Journal of Network Music and Arts* 5.1 (2023).

[61] D. Dziwis, H. Von Coler, and C. Pörschmann. "Orchestra: a toolbox for live music performances in a web-based metaverse". In: *Journal of the Audio Engineering Society* 71.11 (Nov. 2023), pp. 802–812. ISSN: 15494950. DOI: 10.17743/jaes.2022.0096.

[62] W. Eddy. *Transmission Control Protocol (TCP)*. RFC 9293. Aug. 2022. DOI: 10.17487/RFC9293. URL: https://www.rfc-editor.org/info/rfc9293.

[63] Elk. *Elk Audio*. URL: https://www.elk.audio/start.

[64] European Music School Union. *European music schools in times of coronavirus*. 2020. URL: http://www.musicschoolunion.eu/wp-content/uploads/2020/05/EMU-Survey-Coronavirus.pdf.

[65] B. Fang et al. "Development of a Wearable Device for Motion Capturing Based on Magnetic and Inertial Measurement Units". In: *Scientific Programming* 2017 (2017), pp. 1–11. ISSN: 1058-9244, 1875-919X. DOI: 10.1155/2017/7594763.

[66] S. Farner et al. "Ensemble hand-clapping experiments under the influence of delay and various acoustic environments". In: *Journal of the Audio Engineering Society* 57 (12 Dec. 2009), pp. 1028–1041.

[67] Farplay. *Farplay*. URL: https://farplay.io/.

[68] P. Ferguson, C. Chafe, and S. Gapp. "Trans-Europe Express Audio: testing 1000 mile low-latency uncompressed audio between Edinburgh and Berlin using GPS-derived word clock, first with jacktrip then with Dante". In: *Audio Engineering Society Convention 148*. Audio Engineering Society. May 2020.

[69] B. Fg and L. Picinali. "Spatial audio applied to research with the blind". In: (Apr. 2011). Ed. by P. Strumillo. DOI: 10.5772/15206.

[70] R. T. Fielding, M. Nottingham, and J. Reschke. *HTTP Semantics*. RFC 9110. June 2022. DOI: 10.17487/RFC9110. URL: https://www.rfc-editor.org/info/rfc9110.

[71] A. Filali et al. "Multi-access edge computing: a survey". In: *IEEE Access* 8 (2020), pp. 197017–197046. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3034136.

[72] V. Fischer. "Case study: performing band rehearsals on the Internet with Jamulus" (2015). URL: https://jamulus.io/PerformingBandRehearsalsontheInternetWithJamulus.pdf.

[73] P. J. Fitzgibbons, S. Gordon-Salant, and J. Barrett. "Age-related differences in discrimination of an interval separating onsets of successive tone bursts as a function of interval duration". In: *The Journal of the Acoustical Society of America* 122.1 (July 2007), pp. 458–466. ISSN: 0001-4966. DOI: 10.1121/1.2739409.

[74] M. D. Fletcher. "Can haptic stimulation enhance music perception in hearing-impaired listeners?" In: *Frontiers in Neuroscience* 15 (Aug. 2021). ISSN: 1662-453X. DOI: 10.3389/fnins.2021.723877.

[75] A. Flores Ramones and M. S. del-Rio-Guerra. "Recent developments in haptic devices designed for hearing-impaired people: a literature review". In: *Sensors* 23.6 (Mar. 2023). ISSN: 1424-8220. DOI: 10.3390/s23062968.

[76] D. Fober, Y. Orlarey, and S. Letz. "Real time musical events streaming over Internet". In: *Proceedings First International Conference on WEB Delivering of Music. WEDELMUSIC 2001*. 2001, pp. 147–154. DOI: 10.1109/WDM.2001.990170.

[77] C. Frauenberger and M. Noistering. "3D audio interfaces for the blind". Georgia Institute of Technology. 2003.

[78] E. Frid. "Accessible digital musical instruments—a review of musical interfaces in inclusive music practice". In: *Multimodal Technologies and Interaction* 3.3 (July 2019). ISSN: 2414-4088. DOI: 10.3390/mti3030057.

[79] E. Frid and A. Ilsar. "Reimagining (accessible) digital musical instruments: a survey on electronic music-making tools". In: *NIME 2021*. PubPub, June 2021. DOI: 10.21428/92fbeb44.c37a2370.

[80] M. Frigo and S. Johnson. "The Design and Implementation of FFTW3". In: *Proceedings of the IEEE* 93.2 (Feb. 2005), pp. 216–231. ISSN: 0018-9219. DOI: 10.1109/JPROC.2004.840301.

[81] M. J. García Vizcaíno. "Access for the blind in the art setting: tactile paintings as touching experiences?" In: *Museum Management and Curatorship* 41.2 (Mar. 2026), pp. 303–317. ISSN: 0964-7775, 1872-9185. DOI: 10.1080/09647775.2024.2408243.

[82] S. Gibet. "A Grammar of Expressive Conducting Gestures". In: *Sonic Design*. Ed. by A. R. Jensenius. Vol. 12. Springer Nature Switzerland, 2024, pp. 67–91. DOI: 10.1007/978-3-031-57892-2_5.

[83] M. Giordano, J. Sullivan, and M. M. Wanderley. "Design of vibrotactile feedback and stimulation for music performance". In: *Musical Haptics*. Ed. by S. Papetti and C. Saitis. Springer International Publishing, 2018, pp. 193–214. DOI: 10.1007/978-3-319-58316-7_10.

[84] M. Giordano and M. M. Wanderley. "Follow the tactile metronome: vibrotactile stimulation for tempo synchronization in music performance". In: *Proceedings of the Sound and Music Computing Conference. Maynooth, Ireland*. Vol. 2015. 7. 2015.

[85] O. Glauser et al. "Interactive hand pose estimation using a stretch-sensing soft glove". In: *ACM Trans. Graph.* 38.4 (July 2019), 41:1–41:15. ISSN: 0730-0301. DOI: 10.1145/3306346.3322957.

[86] Google LLC. *Google Meet*. URL: https://meet.google.com/.

[87] B. Greinke et al. "An Interactive Garment for Orchestra Conducting: IoT-enabled Textile & Machine Learning to Direct Musical Performance". In: *Proceedings of the Fifteenth International Conference on Tangible, Embedded, and Embodied Interaction*. ACM, Feb. 2021, pp. 1–6. DOI: 10.1145/3430524.3442451.

[88] T. Grosshauser and T. Hermann. "Augmented haptics - an interactive feedback system for musicians". In: *Haptic and Audio Interaction Design*. Ed. by M. E. Altinsoy, U. Jekosch, and S. Brewster. Vol. 5763. Springer Berlin Heidelberg, 2009, pp. 100–108. DOI: 10.1007/978-3-642-04076-4_11.

[89] G. Guennebaud, B. Jacob, et al. *Eigen v3*. 2010. URL: http://eigen.tuxfamily.org.

[90] E. Gunduzhan and K. Momtahan. "Linear prediction based packet loss concealment algorithm for PCM coded speech". In: *IEEE Transactions on Speech and Audio Processing* 9.8 (Nov. 2001), pp. 778–785. ISSN: 10636676. DOI: 10.1109/89.966081.

[91] Guoqiang Zhang and W. B. Kleijn. "Autoregressive model-based speech packet-loss concealment". In: *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, Mar. 2008, pp. 4797–4800. DOI: 10.1109/ICASSP.2008.4518730.

[92] A. Hadjakos. "Pianist motion capture with the Kinect depth camera". In: *Proceedings of the Sound and Music Computing Conference*. 2012, pp. 303–310.

[93] A. Hadjakos, T. Großhauser, and W. Goebl. "Motion analysis of music ensembles with the Kinect". In: *Conference on New Interfaces for Musical Expression*. 2013, pp. 106–110.

[94] R. Hamilton and C. Platz. "Gesture-based collaborative virtual reality performance in carillon". In: *Proceedings of the 2016 international computer music conference*. 2016, pp. 337–340.

[95] B. Han et al. "Network function virtualization: challenges and opportunities for innovations". In: *IEEE Communications Magazine* 53.2 (Feb. 2015), pp. 90–97. ISSN: 0163-6804. DOI: 10.1109/MCOM.2015.7045396.

[96] N. Hanes and S. Lundberg. "E-Learning as a regional policy tool: principles for a cost-benefit analysis". In: *RUSC. Universities and Knowledge Society Journal* 5.1 (Apr. 2008). ISSN: 1698-580X. DOI: 10.7238/rusc.v5i1.325.

[97] B. Hartmann and N. Link. "Gesture recognition with inertial sensors and optimized DTW prototypes". In: *2010 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, Oct. 2010, pp. 2102–2109. DOI: 10.1109/ICSMC.2010.5641703.

[98] P. Hash. "Teaching instrumental music to deaf and hard of hearing students". In: *Research & Issues in Music Education* (2003).

[99] P. M. Hash. "Remote learning in school bands during the COVID-19 shutdown". In: *Journal of Research in Music Education* 68.4 (2021), pp. 381–397.

[100] C. Hawthorne et al. "Enabling factorized piano music modeling and generation with the MAESTRO dataset" (2018). DOI: 10.48550/ARXIV.1810.12247.

[101] A. Haynes et al. "FeelMusic: enriching our emotive experience of music through audio-tactile mappings". In: *Multimodal Technologies and Interaction* 5.6 (May 2021). ISSN: 2414-4088. DOI: 10.3390/mti5060029.

[102] Herzen State Pedagogical University of Russia et al. "Music computer technologies as a means of teaching the musical art for visually-impaired people". In: *ICONI* 1 (2019), pp. 42–53. ISSN: 26584824, 27133095. DOI: 10.33779/2658-4824.2019.1.042-053.

[103] I. J. Hirsh. "Auditory Perception of Temporal Order". In: *The Journal of the Acoustical Society of America* 31.6 (June 1959), pp. 759–767. ISSN: 0001-4966. DOI: 10.1121/1.1907782. eprint: https://pubs.aip.org/asa/jasa/article-pdf/31/6/759/11746722/759_1_online.pdf.

[104] C. Hoene, I. Howell, and A. Carôt. "Networked Music Performance: Developing Soundjack and the Fastmusic Box During the Coronavirus Pandemic". In: *AES International Audio Education Conference*. July 2021.

[105] T. Hopkins et al. "AR drum circle: real-time collaborative drumming in AR". In: *Frontiers in Virtual Reality* 3 (Aug. 2022). ISSN: 2673-4192. DOI: 10.3389/frvir.2022.847284.

[106] I. Howell et al. *Preliminary Report: Comparing the Audio Quality of Classical Music Lessons Over Zoom, Microsoft Teams, VoiceLessonsApp, and Apple FaceTime*. May 2020. URL: https://www.ianhowellcountertenor.com/preliminary-report-testing-video-conferencing-platforms.

[107] *HPSJAM*. URL: https://github.com/hselasky/hpsjam/.

[108] A. Hunt, H. Daffern, and G. Kearney. "Avatar representation in extended reality for immersive networked music performance". In: *Audio Engineering Society Conference: AES 2023 International Conference on Spatial and Immersive Audio*. Audio Engineering Society. 2023.

[109] R. Hupke, S. Preihs, and J. Peissig. "Immersive room extension environment for networked music performance". In: *Audio Engineering Society Convention 153*. Audio Engineering Society. 2022.

[110] E. D. Innocenti et al. "Mobile virtual reality for musical genre learning in primary education". In: *Computers & Education* 139 (Oct. 2019), pp. 102–117. ISSN: 03601315. DOI: 10.1016/j.compedu.2019.04.010.

[111] Intel RealSense Documentation Team. *High-speed capture mode of Intel RealSense Depth Camera D435*. URL: https://dev.intelrealsense.com/docs/high-speed-capture-mode-of-intel-realsense-depth-camera-d435%5C#45-latency.

[112] *Internet Protocol*. RFC 791. Sept. 1981. DOI: 10.17487/RFC0791. URL: https://www.rfc-editor.org/info/rfc791.

[113] B. Iribe. *Virtual reality - a new frontier in computing*. Oculus VR. 2013.

[114] P. Jääskeläinen et al. "HW/SW Co-design Toolset for Customization of Exposed Datapath Processors". In: *Computing Platforms for Software-Defined Radio*. Ed. by W. Hussain et al. Springer International Publishing, 2017, pp. 147–164. DOI: 10.1007/978-3-319-49679-5_8.

[115] JackTrip Foundation. *Building Your Own JackTrip Bridge: An In-Depth Guide*. URL: https://support.jacktrip.com/building-your-own-jacktrip-bridge-an-in-depth-guide (visited on 04/21/2026).

[116] JackTrip Foundation. *JackTrip Lab*. URL: https://www.jacktrip.com/.

[117] JamKazam Inc. *JamKazam*. URL: https://jamkazam.com/.

[118] Q. Jarvis-Holland, C. Cortez, F. Botello, et al. "Expanding access to music technology–rapid prototyping accessible instrument solutions for musicians with intellectual disabilities". In: *arXiv preprint arXiv:2011.09143* (2020).

[119] N. S. Jayant and P. Noll. *Digital coding of waveforms. Principles and applications to speech and video*. Elsevier, 1985.

[120] J. A. Jellison and D. M. Taylor. "Attitudes toward inclusion and students with disabilities: A review of three decades of music research". In: *Bulletin of the Council for Research in Music Education* (2007), pp. 9–23.

[121] R. Jesup, S. Loreto, and M. Tüxen. *WebRTC data channels*. RFC 8831. RFC Editor, Jan. 2021. URL: https://datatracker.ietf.org/doc/html/rfc8831.

[122] G. Johannsen and T. Nakra. "Conductors' gestures and their mapping to sound synthesis". In: *Musical Gestures: Sound, Movement, and Meaning*. Routledge, 2010, pp. 264–298.

[123] S. K. Jones. "Teaching students with disabilities: a review of music education research as it relates to the Individuals with Disabilities Education Act". In: *Update: Applications of Research in Music Education* 34.1 (Nov. 2015), pp. 13–23. ISSN: 8755-1233, 1945-0109. DOI: 10.1177/8755123314548039.

[124] A. M. L. Kappers et al. "Hand-held haptic navigation devices for actual walking". In: *IEEE Transactions on Haptics* 15.4 (Oct. 2022), pp. 655–666. ISSN: 1939-1412, 2329-4051, 2334-0134. DOI: 10.1109/TOH.2022.3209350.

[125] J. Kim, S. Ananthanarayan, and T. Yeh. "Seen music: ambient music data visualization for children with hearing impairments". In: *Proceedings of the 14th International Conference on Interaction Design and Children*. ACM, June 2015, pp. 426–429. DOI: 10.1145/2771839.2771870.

[126] A. King, H. Prior, and C. Waddington-Jones. "Connect Resound: using online technology to deliver music education to remote communities". In: *Journal of Music, Technology & Education* 12.2 (Sept. 2019), pp. 201–217. ISSN: 1752-7066, 1752-7074. DOI: 10.1386/jmte_00006_1.

[127] P. Kolesnik and M. M. Wanderley. "Recognition, analysis and performance with expressive conducting gestures". In: *ICMC*. 2004.

[128] K. Kondo and K. Nakagawa. "A packet loss concealment method using recursive linear prediction". In: *Interspeech 2004*. ISCA, Oct. 2004, pp. 2633–2636. DOI: 10.21437/Interspeech.2004-700.

[129] J. Kreimeier and T. Götzelmann. "Two decades of touchable and walkable virtual reality for blind and visually impaired people: a high-level taxonomy". In: *Multimodal Technologies and Interaction* 4.4 (Nov. 2020). ISSN: 2414-4088. DOI: 10.3390/mti4040079.

[130] T. Laes and H. Westerlund. "Performing disability in music teacher education: moving beyond inclusion through expanded professionalism". In: *International Journal of Music Education* 36.1 (Feb. 2018), pp. 34–46. ISSN: 0255-7614, 1744-795X. DOI: 10.1177/0255761417703782.

[131] E. Lakiotakis, C. Liaskos, and X. Dimitropoulos. "Improving networked music performance systems using application-network collaboration". In: *Concurrency and Computation: Practice and Experience* 31.24 (Dec. 2019). ISSN: 1532-0626, 1532-0634. DOI: 10.1002/cpe.4730.

[132] O. Lartillot and P. Toiviainen. "A Matlab toolbox for musical feature extraction from audio". In: *International conference on digital audio effects*. Vol. 237. Bordeaux. 2007.

[133] C. Lauri and J. Malmgren. "Synchronization of streamed audio between multiple playback devices over an unmanaged IP network". In: *Department of Electrical and Information Technology, Faculty of Engineering, LTH, Lund University, Sweden* (2015).

[134] J. Lazzaro and J. Wawrzynek. *An Implementation Guide for RTP MIDI*. RFC 4696. RFC Editor, Nov. 2006. URL: https://www.rfc-editor.org/info/rfc4696.

[135] J. Lazzaro and J. Wawrzynek. *RTP Payload Format for MIDI*. RFC 4695. RFC Editor, Nov. 2006.

[136] J. Lazzaro and J. Wawrzynek. "A case for network musical performance". In: *Proceedings of the 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video*. Association for Computing Machinery, 2001, pp. 157–166. DOI: 10.1145/378344.378367.

[137] S. Lederer, C. Müller, and C. Timmerer. "Dynamic adaptive streaming over HTTP dataset". In: *Proceedings of the 3rd Multimedia Systems Conference*. ACM, Feb. 2012, pp. 89–94. DOI: 10.1145/2155555.2155570.

[138] J. Y. F. Lee, N. Rajeev, and A. Bhojan. "Goldeye: enhanced spatial awareness for the visually impaired using mixed reality and vibrotactile feedback". In: *ACM Multimedia Asia*. ACM, Dec. 2021, pp. 1–7. DOI: 10.1145/3469877.3495636.

[139] K. Lee et al. "Short-term traffic prediction with deep neural networks: a survey". In: *IEEE Access* 9 (2021), pp. 54739–54756. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3071174.

[140] M. Lee and H. In. "Novel wrist-worn vibrotactile device for providing multi-categorical information for orientation and mobility of the blind and visually impaired". In: *IEEE Access* 11 (2023), pp. 111860–111874. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3321802.

[141] H. Li and H. Hu. "Head gesture recognition combining activity detection and dynamic time warping". In: *Journal of Imaging* 10.5 (2024). DOI: 10.3390/jimaging10050123.

[142] H. B. Lima, C. G. R. D. Santos, and B. S. Meiguins. "A survey of music visualization techniques". In: *ACM Computing Surveys* 54.7 (Sept. 2022), pp. 1–29. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3461835.

[143] S. P. Lipshitz and J. Vanderkooy. "Pulse-code modulation — an overview". In: *Journal of the Audio Engineering Society* 52.3 (Mar. 2004), pp. 200–215.

[144] P. Litman. "The relationship between gesture and sound: a pilot study of choral conducting behaviour in two related settings". In: *Visions of Research in Music Education* 8.1 (2006).

[145] A. Logeswaran et al. "The Role of Extended Reality Technology in Healthcare Education: Towards a Learner-Centred Approach". In: *Future Healthcare Journal* 8.1 (Mar. 2021), e79–e84. ISSN: 25146645. DOI: 10.7861/fhj.2020-0112.

[146] B. Loveridge. "Networked music performance in virtual reality: current perspectives". In: *Journal of Network Music and Arts* 2.1 (2020).

[147] L. Lu et al. "Playing with feeling: exploring vibrotactile feedback and aesthetic experiences for developing haptic wearables for blind and low vision music learning". In: *The 25th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, Oct. 2023, pp. 1–16. DOI: 10.1145/3597638.3608397.

[148] F. Mancini et al. "Whole-body mapping of spatial acuity for pain and touch". In: *Annals of Neurology* 75.6 (June 2014), pp. 917–924. ISSN: 0364-5134, 1531-8249. DOI: 10.1002/ana.24179.

[149] M. Mandanici, R. Di Filippo, and S. Delle Monache. "The discovery of interactive spaces: learning by design in high school music technology classes". In: *Technology, Knowledge and Learning* 26.4 (Dec. 2021), pp. 1131–1151. ISSN: 2211-1662, 2211-1670. DOI: 10.1007/s10758-020-09464-4.

[150] G. F. Martín. "Social and psychological impact of musical collective creative processes in virtual environments; the Avatar Orchestra Metaverse in Second Life". In: *Music Technol.* 75 (2018), pp. 75–87. ISSN: 1974-0050.

[151] V. F. Martins, L. Gomez, and A. G. Dionísio Corrêa. "Teaching children musical perception with MUSIC-AR". In: *EAI Endorsed Transactions on e-Learning* 2.5 (Mar. 2015). ISSN: 2032-9253. DOI: 10.4108/el.2.5.e3.

[152] T. B. McHugh et al. "Towards Inclusive Streaming: Building Multimodal Music Experiences for the Deaf and Hard of Hearing". In: *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, May 2021, pp. 1–7. DOI: 10.1145/3411763.3451690.

[153] A. P. McPherson, R. H. Jack, G. Moro, et al. "Action-sound latency: Are our tools fast enough?" (2016).

[154] A. P. McPherson and V. Zappi. "An Environment for Submillisecond-Latency Audio and Sensor Processing on BeagleBone Black" (2015).

[155] A. I. Mezza and A. Bernardini. *The IEEE-IS2 2024 Music Packet Loss Concealment Challenge*. 2024. DOI: 10.48550/ARXIV.2409.18564.

[156] A. I. Mezza et al. "Hybrid packet loss concealment for real-time networked music applications". In: *IEEE Open Journal of Signal Processing* 5 (2024), pp. 266–273. ISSN: 2644-1322. DOI: 10.1109/OJSP.2023.3343318.

[157] Microsoft Corp. *Skype*. URL: https://www.skype.com.

[158] MIDI Manufacturers Association (MMA). *Complete MIDI 1.0 Detailed Specification Document*. 1996. URL: https://www.midi.org/specifications-old/item/the-midi-1-0-specification.

[159] *MIDI specification page*. [Accessed 13-10-2025]. URL: https://midi.org/specs.

[160] M. M. Mohamed, M. A. Nessiem, and B. W. Schuller. "On deep speech packet loss concealment: A mini-survey" (2020). DOI: 10.48550/ARXIV.2005.07794.

[161] L. Mohjazi et al. "The journey toward 6G: a digital and societal revolution in the making". In: *IEEE Internet of Things Magazine* 7.2 (Mar. 2024), pp. 119–128. ISSN: 2576-3180, 2576-3199. DOI: 10.1109/IOTM.001.2300119.

[162] V. J. K. Morinigo, S. Benatti, and L. Benini. "A high SNR, low-latency dry EMG acquisition system for unobtrusive HMI devices". In: *2022 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, Oct. 2022, pp. 544–548. DOI: 10.1109/BioCAS54905.2022.9948679.

[163] W. A. Munson and M. B. Gardner. "Standardizing auditory tests". In: *Journal of the Acoustical Society of America* 22.4 (1950), pp. 675–675. DOI: 10.1121/1.1917190.

[164] B. Naderi et al. "VCD: a video conferencing dataset for video compression". In: *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Apr. 2024, pp. 3970–3974. DOI: 10.1109/ICASSP48485.2024.10448484.

[165] National Center for Education Statistics, U.S. Department of Education, Institute of Education Sciences. *Students with disabilities. Condition of education*. 2022. URL: https://nces.ed.gov/programs/coe/indicator/cgg.

[166] J. R. Nelson, A. J. Heidorn, and R. J. Cox. *Reliable real-time transmission of musical sound control data over wireless networks*. US Patent 9,601,097 B2. Mar. 2017. URL: https://patents.google.com/patent/US9601097B2/en.

[167] NetworkSound. *JackStreamer*. URL: http://www.networksound.com/index.php/products/jrpbox (visited on 04/21/2026).

[168] D. Nie et al. "E-Learning Financing Models in Russia for Sustainable Development". In: *Sustainability* 12.11 (May 2020). ISSN: 2071-1050. DOI: 10.3390/su12114412.

[169] K. Nisar et al. "A survey on the architecture, application, and security of software defined networking: challenges and open issues". In: *Internet of Things* 12 (Dec. 2020). ISSN: 25426605. DOI: 10.1016/j.iot.2020.100289.

[170] E. J. O'Leary and J. K. Bannerman. "Technology integration in music education and the COVID-19 Pandemic". In: *Bulletin of the Council for Research in Music Education* 238 (Oct. 2023), pp. 23–40. ISSN: 0010-9894, 2162-7223. DOI: 10.5406/21627223.238.02.

[171] Ohidujjaman et al. "Packet loss compensation for VoIP through bone-conducted speech using modified linear prediction". In: *IEEJ Transactions on Electrical and Electronic Engineering* 18.11 (Nov. 2023), pp. 1781–1790. ISSN: 1931-4973, 1931-4981. DOI: 10.1002/tee.23907.

[172] A. Olmos et al. "Exploring the role of latency and orchestra placement on the networked performance of a distributed opera". In: *12th Annual International Workshop on Presence*. Vol. 10. IWP Los Angeles. 2009, pp. 1–9.

[173] N. C. Onuora-Oguno and D. S. Umeojiaka. "The dominance of e-learning and Information Communication Technology (ICT) in nigerian music education: problems and prospects". In: *Journal of African Studies and Sustainable Development* 3.1 (2020).

[174] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition". In: *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*. IEEE Comput. Soc. Press, 1993, pp. 40–44. DOI: 10.1109/ACSSC.1993.342465.

[175] P. Paul et al. "Infrastructure-Technology, Socio-Economical Issues and Challenges in ICT-based Education and Digital Education with Possible Solutions-A Scientific Observation". In: *SSRN Electronic Journal* (2023). ISSN: 1556-5068. DOI: 10.2139/ssrn.4491058.

[176] W. C. Payne et al. "Empowering blind musicians to compose and notate music with SoundCells". In: *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, Oct. 2022, pp. 1–14. DOI: 10.1145/3517428.3544825.

[177] *PCM1803 reference page*. [Accessed 29-07-2023]. URL: https://www.ti.com/product/PCM1803.

[178] R. Pelzer et al. "Head-related transfer function recommendation based on perceptual similarities and anthropometric features". In: *The Journal of the Acoustical Society of America* 148.6 (Dec. 2020), pp. 3809–3817. ISSN: 0001-4966, 1520-8524. DOI: 10.1121/10.0002884.

[179] C. Perkins, O. Hodson, and V. Hardman. "A survey of packet loss recovery techniques for streaming audio". In: *IEEE network* 12.5 (Sept. 1998), pp. 40–48. DOI: 10.1109/65.730750.

[180] M. Piçarra, A. Rodrigues, and J. Guerreiro. "Evaluating accessible navigation for blind people in virtual environments". In: *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Apr. 2023, pp. 1–7. DOI: 10.1145/3544549.3585813.

[181] M. Pielot et al. "A Tactile Compass for Eyes-Free Pedestrian Navigation". In: *Human-Computer Interaction – INTERACT 2011*. Vol. 6947. 2011, pp. 640–656. DOI: 10.1007/978-3-642-23771-3_47.

[182] M. Popoff et al. "Towards an FPGA-based compilation flow for ultra-low latency audio signal processing". In: *SMC-22-Sound and Music Computing*. Zenodo, June 2022. DOI: 10.5281/ZENODO.6798303.

[183] L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*. Prentice-Hall, Inc., 1993.

[184] *Raspberry Pi 4 Model B datasheet*. [Accessed 13-10-2025]. URL: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf.

[185] B. Redman and J. Impett. "The potential of videoconferencing and low-latency (LoLa) technology for instrumental music teaching". In: *Music & Practice* 6 (July 2020). ISSN: 1893-9562. DOI: 10.32063/0610.

[186] B. Remache-Vinueza et al. "Audio-Tactile Rendering: A Review on Technology and Methods to Convey Musical Information through the Sense of Touch". In: *Sensors* 21.19 (Sept. 2021). ISSN: 1424-8220. DOI: 10.3390/s21196575.

[187] T. Risset et al. "faust2fpga for ultra-low audio latency: Preliminary work in the syfala project". In: *IFC 2020-Second International Faust Conference*. 2020, pp. 1–9.

[188] M. Rofe, E. Geelhoed, and L. Hodsdon. "Experiencing Online Orchestra: Communities, Connections and Music-Making through Telematic Performance". In: *Journal of Music, Technology & Education* 10.2 (Sept. 2017), pp. 257–275. ISSN: 1752-7066, 1752-7074. DOI: 10.1386/jmte.10.2-3.257_1.

[189] C. Rottondi. "Inclusiveness in Remote Music Teaching and Networked Music Performances: Vision, Technological Enablers and Design Strategies". In: *IEEE Communications Magazine* 62.12 (Dec. 2024), pp. 34–40. ISSN: 0163-6804, 1558-1896. DOI: 10.1109/MCOM.001.2400156.

[190] C. Rottondi, M. Buccoli, and M. Zanoni. "Feature-Based Analysis of the Effects of Packet Delay on Networked Musical Interactions". In: *Journal of the Audio Engineering Society* 63.11 (Dec. 2015), pp. 864–875. ISSN: 15494950. DOI: 10.17743/jaes.2015.0074.

[191] C. Rottondi et al. "An overview on networked music performance technologies". In: *IEEE Access* 4 (2016), pp. 8823–8843. DOI: 10.1109/ACCESS.2016.2628440.

[192] C. Rottondi et al. "Toward an Inclusive Framework for Remote Musical Education and Practices". In: *IEEE Access* 12 (2024), pp. 173836–173849. DOI: 10.1109/ACCESS.2024.3501414.

[193] C. Rousseau et al. "Assessing posture while playing in musicians — a systematic review". In: *Applied Ergonomics* 106 (2023). ISSN: 0003-6870. DOI: 10.1016/j.apergo.2022.103883.

[194] *RTP-MIDI or MIDI over Networks*. URL: https://www.midi.org/midi-articles/rtp-midi-or-midi-over-networks.

[195] M. Sacchetto et al. "Pristine Quality Networked Music Performance System for the Web". In: *Proceedings of the 10th Convention of the European Acoustics Association Forum Acusticum 2023*. European Acoustics Association, Jan. 2024, pp. 2957–2964. DOI: 10.61782/fa.2023.0556.

[196] M. Sacchetto, C. Rottondi, and A. Bianco. "Implementation and optimization of Burg's method for real-time packet loss concealment in networked music performance applications". In: *Personal and Ubiquitous Computing* 28.5 (Oct. 2024), pp. 727–743. ISSN: 1617-4909, 1617-4917. DOI: 10.1007/s00779-024-01806-8.

[197] M. Sacchetto, A. Servetti, and C. Chafe. "JackTrip WebRTC: networked music experiments in a web browser". In: *Proc. of the 6th International Web Audio Conference (WAC)*. July 2021.

[198] M. Sacchetto et al. "Collection of Design Directions for the Realization of a Visual Interface with Haptic Feedback to Convey the Notion of Sonic Grain to DHH Students". In: *2023 4th International Symposium on the Internet of Sounds*. IEEE, Oct. 2023, pp. 1–7. DOI: 10.1109/IEEECONF59510.2023.10335449.

[199] M. Sacchetto et al. "HiFiReM: a unified platform, both web-based and native, for remote music education". In: *Proc. XXIII Colloqui di Informatica Musicale*. Oct. 2022.

[200] M. Sacchetto et al. "Using autoregressive models for real-time packet loss concealment in networked music performance applications". In: *AudioMostly 2022*. ACM, Sept. 2022, pp. 203–210. DOI: 10.1145/3561212.3561226.

[201] M. Sacchetto et al. "Web-Based Networked Music Performances via WebRTC: A Low-Latency PCM Audio Solution". In: *Journal of the Audio Engineering Society* 70.11 (Dec. 2022), pp. 926–937. ISSN: 15494950. DOI: 10.17743/jaes.2022.0021.

[202] W. Safi et al. "Blind navigation of web pages through vibro-tactile feedbacks". In: *25th ACM Symposium on Virtual Reality Software and Technology*. ACM, Nov. 2019, pp. 1–1. DOI: 10.1145/3359996.3364758.

[203] A. Saha, T. B. McHugh, and A. M. Piper. "Tutoria11y: enhancing accessible interactive tutorial creation by blind audio producers". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Apr. 2023, pp. 1–14. DOI: 10.1145/3544548.3580698.

[204] M. G. Salimon et al. "E-learning satisfaction and retention: A concurrent perspective of cognitive absorption, perceived social presence and technology acceptance model". In: *Journal of Systems and Information Technology* 23.1 (June 2021), pp. 109–129. ISSN: 1328-7265, 1328-7265. DOI: 10.1108/JSIT-02-2020-0029.

[205] A. D. Samala et al. "Unveiling the landscape of generative artificial intelligence in education: a comprehensive taxonomy of applications, challenges, and future prospects". In: *Education and Information Technologies* 30.3 (Feb. 2025), pp. 3239–3278. ISSN: 1360-2357, 1573-7608. DOI: 10.1007/s10639-024-12936-0.

[206] A. Sarasua. "Context-aware gesture recognition in classical music conducting". In: *Proceedings of the 21st ACM International Conference on Multimedia*. ACM, Oct. 2013, pp. 1059–1062. DOI: 10.1145/2502081.2502216.

[207] S. Schulze and E. J. King. "Sparse pursuit and dictionary learning for blind source separation in polyphonic music recordings". In: *EURASIP Journal on Audio, Speech, and Music Processing* 2021.1 (Jan. 2021). ISSN: 1687-4722. DOI: 10.1186/s13636-020-00190-4.

[208] H. Schulzrinne et al. *RTP: A Transport Protocol for Real-Time Applications*. STD 64. RFC Editor, July 2003. URL: http://www.rfc-editor.org/rfc/rfc3550.txt.

[209] F. A. Seddon. "Modes of communication during jazz improvisation". In: *British Journal of Music Education* 22.1 (2005), pp. 47–61. DOI: 10.1017/S0265051704005984.

[210] S. S. Sefati and S. Halunga. "Ultra-reliability and low-latency communications on the internet of things based on 5G network: literature review, classification, and future research view". In: *Transactions on Emerging Telecommunications Technologies* 34.6 (June 2023). ISSN: 2161-3915, 2161-3915. DOI: 10.1002/ett.4770.

[211] S. Serafin et al. "Considerations on the use of virtual and augmented reality technologies in music education". In: *2017 IEEE Virtual Reality Workshop on K-12 Embodied Learning through Virtual & Augmented Reality (KELVAR)*. June 2017, pp. 1–4. DOI: 10.1109/KELVAR.2017.7961562.

[212] X. Serra and J. Smith. "Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic Plus Stochastic Decomposition". In: *Computer Music Journal* 14.4 (1990). ISSN: 01489267. DOI: 10.2307/3680788. JSTOR: 3680788.

[213] R. Setiawan et al. "Tech-Driven Transformation: Innovative Pricing Strategies for E-Learning". In: *IEEE Access* 12 (2024), pp. 59063–59078. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3392489.

[214] *Setting up a virtual MIDI network*. URL: https://help.ableton.com/hc/en-us/articles/209071169-Setting-up-a-virtual-MIDI-network.

[215] L. Severi and C. Rottondi. "Sparse Linear Prediction for Packet Loss Concealment in Networked Music Performances". In: *2025 IEEE 6th International Symposium on the Internet of Sounds (IS2)*. IEEE, Oct. 2025, pp. 1–5. DOI: 10.1109/IS264627.2025.11284650.

[216] L. Severi et al. "Assessment of Recovery Journal-Based Packet Loss Concealment Techniques for Low-Latency MIDI Streaming". In: *Journal of the Audio Engineering Society* 71.12 (Dec. 2023), pp. 872–885. ISSN: 15494950. DOI: 10.17743/jaes.2022.0115.

[217] L. Severi et al. *Demonstration of a Networked Music Performance Experience with MEVO*. 2024. DOI: 10.48550/arXiv.2404.09665. arXiv: 2404.09665 [cs.NI].

[218] L. Severi et al. "Introducing DUST: A Dataset of Real-Time UDP Sound Packet Traces". In: *2024 IEEE 5th International Symposium on the Internet of Sounds (IS2)*. IEEE, Sept. 2024, pp. 1–4. DOI: 10.1109/IS262782.2024.10704190.

[219] L. Severi et al. "Remote orchestral conduction via a virtual reality system". In: *2024 IEEE 5th International Symposium on the Internet of Sounds (IS2)*. IEEE, Sept. 2024, pp. 1–6. DOI: 10.1109/IS262782.2024.10704182.

[220] L. Shtrepi, S. Di Blasio, and A. Astolfi. "Listeners sensitivity to different locations of diffusive surfaces in performance spaces: the case of a shoebox concert hall". In: *Applied Sciences* 10.12 (2020). ISSN: 2076-3417. DOI: 10.3390/app10124370. URL: https://www.mdpi.com/2076-3417/10/12/4370.

[221] A. Skuse and S. Knotts. "Creating an online ensemble for home based disabled musicians: why disabled people must be at the heart of developing technology". In: *Proceedings of the 2020 Conference on New Interfaces for Musical Expression*. 2020.

[222] M. Soto Ramos, E. Băutu, and M. Dorin Popovici. "Exploring the potential of extended reality (XR) in education: insights from a survey and case studies". In: *16th International Conference on Education and New Learning Technologies*. July 2024, pp. 1028–1037. DOI: 10.21125/edulearn.2024.0351.

[223] SOUNDMIT. *International Faust Conference — 21 November 2024 — Torino*. Timestamp: 4:34:00. SOUNDMIT, Nov. 21, 2024. URL: https://www.youtube.com/watch?v=zli5sFc5dlE&t=16440s (visited on 01/26/2026).

[224] R. Steinmetz. "Human perception of jitter and media synchronization". In: *IEEE Journal on Selected Areas in Communications* 14.1 (1996), pp. 61–72. DOI: 10.1109/49.481694.

[225] Syneme. *Artsmesh*. URL: https://www.artsmesh.com/.

[226] M. S. Taat and A. Francis. "Factors Influencing the Students' Acceptance of E-Learning at Teacher Education Institute: An Exploratory Study in Malaysia." In: *International Journal of Higher Education* 9.1 (2020), pp. 133–141.

[227] D. A. Tedjopurnomo et al. "A survey on modern deep neural network for traffic prediction: Trends, methods and challenges". In: *IEEE Transactions on Knowledge and Data Engineering* (2020), pp. 1–1. ISSN: 1041-4347, 1558-2191, 2326-3865. DOI: 10.1109/TKDE.2020.3001195.

[228] S. Thorn. "Telematic wearable music: remote ensembles and inclusive embodied education". In: *Audio Mostly 2021*. ACM, Sept. 2021, pp. 188–195. DOI: 10.1145/3478384.3478386.

[229] R. Tibshirani. "Regression shrinkage and selection via the Lasso". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 58.1 (Jan. 1996), pp. 267–288. ISSN: 1369-7412, 1467-9868. DOI: 10.1111/j.2517-6161.1996.tb02080.x.

[230] J. Tosi et al. "Performance evaluation of Bluetooth Low Energy: a systematic review". In: *Sensors* 17.12 (Dec. 2017). ISSN: 1424-8220. DOI: 10.3390/s17122898.

[231] K. Tsioutas and G. Xylomenos. "Audio Delay in Web Conference Tools" (June 2022). DOI: 10.5281/ZENODO.6768273.

[232] L. Turchet. "Musical metaverse: vision, opportunities, and challenges". In: *Personal and Ubiquitous Computing* 27.5 (Oct. 2023), pp. 1811–1827. ISSN: 1617-4917. DOI: 10.1007/s00779-023-01708-1.

[233] L. Turchet and F. Antoniazzi. "Semantic web of musical things: achieving interoperability in the internet of musical things". In: *Journal of Web Semantics* 75 (Jan. 2023). ISSN: 1570-8268. DOI: 10.1016/j.websem.2022.100758.

[234] L. Turchet and M. Barthet. "Co-design of musical haptic wearables for electronic music performer's communication". In: *IEEE Transactions on Human-Machine Systems* 49.2 (Apr. 2019), pp. 183–193. ISSN: 2168-2291, 2168-2305. DOI: 10.1109/THMS.2018.2885408.

[235] L. Turchet and P. Casari. "Latency and reliability analysis of a 5G-enabled internet of musical things system". In: *IEEE Internet of Things Journal* 11.1 (Jan. 2024), pp. 1228–1240. ISSN: 2327-4662, 2372-2541. DOI: 10.1109/JIOT.2023.3288818.

[236] L. Turchet, R. Hamilton, and A. Çamci. "Music in extended realities". In: *IEEE Access* 9 (Jan. 2021), pp. 15810–15832. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3052931.

[237] L. Turchet and E. Rinaldo. "Technical performance assessment of the Ableton Link protocol over Wi-Fi". In: *Journal of the Audio Engineering Society* 69.10 (Oct. 2021), pp. 748–756. ISSN: 15494950. DOI: 10.17743/jaes.2021.0014.

[238] L. Turchet and M. Tomasetti. "Immersive networked music performance systems: identifying latency factors". In: *2023 Immersive and 3D Audio: From Architecture to Automotive (I3DA)*. IEEE, Sept. 2023, pp. 1–6. DOI: 10.1109/I3DA57090.2023.10289169.

[239] L. Turchet et al. "5G-Enabled Internet of Musical Things Architectures for Remote Immersive Musical Practices". In: *IEEE Open Journal of the Communications Society* 5 (2024), pp. 4691–4709. DOI: 10.1109/OJCOMS.2024.3407708.

[240] L. Turchet et al. "Exposure to vibrotactile music improves audiometric performances in individuals with cochlear implants". In: *Scientific Reports* 15.1 (June 2025). ISSN: 2045-2322. DOI: 10.1038/s41598-025-02946-4.

[241] L. Turchet et al. "Internet of musical things: vision and challenges". In: *IEEE Access* 6 (2018), pp. 61994–62017. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2872625.

[242] L. Turchet et al. “The internet of musical things ontology”. In: *Journal of Web Semantics* 60 (Jan. 2020). ISSN: 1570-8268. DOI: 10.1016/j.websem.2020.100548.

[243] *User Datagram Protocol*. RFC 768. Aug. 1980. DOI: 10.17487/RFC0768. URL: https://www.rfc-editor.org/info/rfc768.

[244] J.-M. Valin et al. *Real-time packet loss concealment with mixed generative and predictive model*. 2022. DOI: 10.48550/ARXIV.2205.05785.

[245] B. Van Kerrebroeck et al. “The virtual drum circle: polyrhythmic music interactions in mixed reality”. In: *Journal of New Music Research* 0.0 (Apr. 2024), pp. 1–21. DOI: 10.1080/09298215.2024.2339244.

[246] K. Van Weelden. “Relationships between Perceptions of Conducting Effectiveness and Ensemble Performance”. In: *Journal of Research in Music Education* 50.2 (July 2002), pp. 165–176. ISSN: 0022-4294, 1945-0095. DOI: 10.2307/3345820.

[247] W. Verhelst and M. Roelands. “An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 1993. DOI: 10.1109/icassp.1993.319366.

[248] J. Vetter. “WELLE-a web-based music environment for the blind”. In: *Proceedings of the International Conference on New Interfaces for Musical Expression. Birmingham, United Kingdom*. 2020, pp. 701–705.

[249] R. Vieira and F. Schiavoni. “Sunflower: an environment for standardized communication of IoMusT”. In: *Proceedings of the 16th International Audio Mostly Conference*. AM ’21. Association for Computing Machinery, Oct. 2021, pp. 175–181. DOI: 10.1145/3478384.3478414.

[250] E. O. Vieux et al. “Designing Haptic Feedback Stimuli to Convey Gestures of an Orchestra Conductor to Visually-Impaired Musicians”. In: *2025 IEEE 6th International Symposium on the Internet of Sounds (IS2)*. IEEE, Oct. 2025, pp. 1–10. DOI: 10.1109/IS264627.2025.11284557.

[251] J. Virolainen and P. Laine. *Method and apparatus for enabling music error recovery over lossy channels*. US Patent 2004/0154460 A1. Aug. 2004. URL: https://patents.google.com/patent/US20040154460A1/en.

[252] J. Virolainen and P. Laine. *Methods and apparatus for transmitting MIDI data over a lossy communications channel*. US Patent 6,898,729 B2. May 2005. URL: https://patents.google.com/patent/US6898729B2/en.

[253] *Vision Pro and Quest 3 hand-tracking latency compared*. Accessed on May 01, 2024. URL: https://www.roadtovr.com/apple-vision-pro-meta-quest-3-hand-tracking-latency-comparison/.

[254] E. Volta and N. Di Stefano. "Using Wearable Sensors to Study Musical Experience: A Systematic Review". In: *Sensors* 24.17 (Sept. 2024). ISSN: 1424-8220. DOI: 10.3390/s24175783.

[255] K. Vos. *A fast implementation of Burg's method*. OPUS codec. 2013.

[256] B. W. Wah, X. Su, and D. Lin. "A survey of error-concealment schemes for real-time audio and video transmissions over the Internet". In: *Proceedings International Symposium on Multimedia Software Engineering*. IEEE Comput. Soc, 2000, pp. 17–24. DOI: 10.1109/MMSE.2000.897185.

[257] M. Warburton et al. "Measuring Motion-to-Photon Latency for Sensorimotor Experiments with Virtual Reality Systems". In: *Behavior Research Methods* 55.7 (Oct. 2022), pp. 3658–3678. ISSN: 1554-3528. DOI: 10.3758/s13428-022-01983-5.

[258] C. Wegener, S. Stang, and M. Neupert. "FPGA-accelerated real-time audio in pure data". In: *Proc. Int. Conf. in Sound and Music Computing, SMC-22*. Zenodo, June 2022. DOI: 10.5281/ZENODO.6572969.

[259] X. Wei, M. Zhang, and L. Zhou. "Cross-modal transmission strategy". In: *IEEE Transactions on Circuits and Systems for Video Technology* 32.6 (June 2022), pp. 3991–4003. ISSN: 1051-8215, 1558-2205. DOI: 10.1109/TCSVT.2021.3105130.

[260] X. Wei et al. "Cross-modal communication technology: A survey". In: *Fundamental Research* (2023). ISSN: 2667-3258. DOI: 10.1016/j.fmre.2023.08.002.

[261] X. Wei et al. "Perception-aware cross-modal signal reconstruction: from audio-haptic to visual". In: *IEEE Transactions on Multimedia* 25 (2023), pp. 5527–5538. ISSN: 1520-9210, 1941-0077. DOI: 10.1109/TMM.2022.3194309.

[262] C. Werner and R. Kraneis. "UNISON: a novel system for ultra-low latency audio streaming over the internet". In: *2021 IEEE 18th Annual Consumer Communications Networking Conference (CCNC)*. 2021, pp. 1–4. DOI: 10.1109/CCNC49032.2021.9369466.

[263] Wikipedia. *Comparison of remote music performance software — Wikipedia, The Free Encyclopedia*. [Online; accessed 13-October-2025]. 2025. URL: http://en.wikipedia.org/w/index.php?title=Comparison%5C%20of%5C%20remote%5C%20music%5C%20performance%5C%20software&oldid=1290093290.

[264] M. Wright. "Open Sound Control: An Enabling Technology for Musical Networking". In: *Organised Sound* 10.3 (Dec. 2005), pp. 193–200. ISSN: 1355-7718, 1469-8153. DOI: 10.1017/S1355771805000932.

[265] G. Xylomenos et al. “Reduced switching delay for networked music performance”. In: *Packet Video Workshop (Poster Session)*. 2013.

[266] J. Yu et al. *6G Mobile-Edge Empowered Metaverse: Requirements, Technologies, Challenges and Research Directions*. June 2023. DOI: 10.48550/arXiv.2211.04854.

[267] W. Zhang. “Dynamic pose recognition based on deep learning: developing a CNN model for choral conductor pose recognition”. In: *Journal of Computational Methods in Sciences and Engineering* (2025).

[268] Zoom Video Communications, Inc. *Zoom*. URL: https://www.zoom.us.

This Ph.D. thesis has been typeset by means of the TEX-system facilities. The typesetting engine was pdfLATEX. The document class was `toptesi`, by Claudio Beccari, with option `tipotesi=scudo`. This class is available in every up-to-date and complete TEX-system installation.