Submission to the International Graduate Residency in Ubiquitous Music (GIM 2026)

A Multi-Modal Adaptive Microtuning Architecture: Bridging MTS, MPE, and MIDI 2.0 for Real-Time Just Intonation in Ubiquitous Music

Rui Su[1,3], Joseph Timoney[1,3], Damián Keller[2,3]
[1] Department of Computer Science, Maynooth University, Ireland
[2] Federal University of Acre, Brazil
[3] Ubiquitous Music Group

Abstract

Real time adaptive Just Intonation (JI), as it pertains to use with MIDI keyboard instruments, has been limited historically due to trade-offs between the accuracy at which pitch can be tuned and the degree of compatibility that exists with hardware capable of transmitting MIDI data. While the MIDI Tuning Standard (MTS) provides for greater resolution of pitch through the transmission of System Exclusive (SysEx) messages, these are generally poorly supported in consumer equipment. Conversely, while Polyphonic Pitch Bend methods (such as MIDI Polyphonic Expression or MPE) provide for broad compatibility they often suffer from being less precise. Lastly, deployment to web browsers introduces another constraint: that SysEx is blocked by default for equipment connection security purposes. This paper discusses this specific part of the work from Instant Harmonies, a browser native multi-protocol MIDI architecture designed to address the three constraints outlined above, specifically: (1) precision, (2) compatibility, and (3) browser permissions. These constraints have been addressed through fallback mechanisms among three layers of the proposed system including: (1) MTS Scale/Octave 2-byte message communication protocols when SysEx permission and receiver capabilities are present; (2) an optimised version of MPE using +/- 2 semitone RPN 0 pitch bend sensitivity; and (3) a proof-of-concept layer utilising a MIDI 2.0 Universal MIDI Packet (UMP) Export protocol. The work described here falls under the umbrella of the ubiquitous music (ubimus) research agenda. Specifically, this work frames zero installation, adaptive JI as a mechanism to enable access to creative participation in struck string interaction and in pedagogical residency settings (such as GIM). An empirical assessment was conducted across two contrasting piano compositions (Mozart's Piano Sonata No. 11, K. 331/III; Bach's Prelude in C Major, BWV 846) where both operating modes were shown to achieve resolutions over two orders of magnitude finer than a conservative 3 cent engineering retuning reference. In addition to achieving higher resolutions, both operating modes had an estimated total processing and MIDI transmission latency of sub-5 milliseconds. Large effect sizes (Cohen's d = 1.54 – 1.71) were also observed.

Keywords: Ubiquitous music; Adaptive Just Intonation; Web MIDI API; MIDI Tuning Standard; MPE; MIDI 2.0; Struck-string interaction; Accessibility.

1. Introduction

Ubiquitous Music (ubimus) represents the intersection of sound and music computing, human-computer interaction, studies of creativity, and music education, and includes an explicitly social/communal basis (Keller, et al., 2019). Keller's approach to ubimus draws upon Weiser's (1991) idea of embedding technology into the fabric of everyday life. Therefore, ubimus views pervasive use of music in modern society as an opportunity to creatively participate in music-making. Keller, Jagwani, and Lazzarini (2025) describe struck-string interaction as fostering generative techniques that explore boundaries between tuning and sonic qualities.. Since tuning, assigning specific frequencies to specific pitches, is not a secondary consideration but rather one of the primary aesthetic dimensions of the ubimus project, it follows that issues related to tuning will need to be resolved at the outset.

The majority of struck-string music today is played using some form of Equal Temperament (12-TET). The 12-TET is a compromise tuning in which the octave is divided into twelve logarithmically equal semitones. Beginning with the earliest modern periods – Muzzulini (2021), e.g., describes Newton's circular-diagram extensions to Descartes' Compendium Musicae (1650) that extend through the octave in 53 and 612 equal divisions to deal specifically with this issue, digital instrument builders today are revisiting this line in light of limitations imposed by the use of fixed pitch MIDI equipment. This method produces acoustic approximations of the simple integer ratios perceived as being the most consonant by the human auditory system. For example, while a pure perfect fifth is defined by a frequency ratio of 3:2 (701.96 cents), the 12-TET fifth is approximately 700 cents. Similarly, a pure major third is defined by a frequency ratio of 5:4 (386.31 cents), but in 12-TET, it is widened to 400 cents—a difference of 14 cents that results in audible beating in the upper partials (Vos, 1982). While Just Intonation (JI) does eliminate this roughness by defining intervals based on exact integer ratios, there exists no single JI scale that will include all possible key signatures (Mohrlok, 2003), thereby necessitating adaptive retuning when changes occur in the harmonic environment.

It is this problem-solving challenge that gives rise to the current research. Extending access to real-time adaptive JI to a broad class of MIDI-compatible keyboards and external synthesisers, subject to protocol support and configuration, directly serves the ubimus objective of accessible creative participation available to a broader audience and align with the educational objectives of programmes such as the International Graduate Residency in Ubiquitous Music (GIM), where microtonal practice may be examined collaboratively among students of different levels of experience. Nevertheless, translating microtonal pitch variation into fixed-pitch electronic instruments still poses practical problems. Typically developers have three choices: (1) the standard MIDI Tuning Standard (MTS), which permits exact control via its System Exclusive (SysEx) message but has only limited hardware support (Moussa, 2006); (2) Polyphonic Pitch Bend methods, which are supported on

many platforms, but lack sufficient resolution-headroom and require users to carefully manage channels (Celik, 2016); or (3) for browser-based deployments: SysEx messages are blocked from transmission by default as a security precaution. In this paper we propose a unified architecture that provides a solution to the three challenges mentioned above, precision, compatibility, and browser permission issues, utilizing a hierarchical fallback strategy implemented on top of the Web MIDI API (Baratè & Ludovico 2022; Wilson & Wilson, 2025).

2. Background and Related Work

2.1 Ubimus Framing

The prior paper – Su, Timoney & Keller (2024) described the initial ubimus toolset to deploy Just Intonation (JI) in struck string interaction and introduced a web based prototype to convert MIDI to JI. In addition, Su, Timoney & Keller (2025) explored the application of adaptive tuning techniques in the context of the ubimus framework and recognised the necessity of creating a browser based platform that could provide both real time tuning capabilities and interface with external hardware. Keller, Jagwani & Lazzarini (2025) have also developed a ubimus plug-in framework using Field Programmable Gate Array (FPGA) components to allow for reconfiguration of hardware interfaces. This project builds upon the reconfigurable nature of the FPGA hardware, but introduces a software level hierarchical fallback mechanism that will enable the same browser session to adapt to various instrument types encountered in classroom, residency and every day environments.

2.2 Psychoacoustic Basis and Resolution Requirements

Adaptive Just Intonation (AJI) relies on psychoacoustic principles. Plomp and Levelt (1965) illustrated that maximum consonance for harmonic spectral configurations occur at simple integer ratios. Dissonance increases as the separation between partials approaches about 25% of a critical bandwidth. Vos (1982, 1984) demonstrated that discrimination between pure and tempered intervals are primarily due to differences in beat perception between near-coincident partials. For example, the just intonation fifth differs from the tempered version by only about 2 cents, whereas the just intonation major third differs from the tempered version by about 14 cents. Schwär et al. (2021) experimentally confirmed Vos' empirical results in multi-track chorale recordings. Hasegawa (2013) summarised Tenney's tolerance principle: the degree of tolerance to an interval inversely relates to the complexity of the ratio: simpler intervals are recognizable over larger tuning errors than more complex intervals. Mohrlok (2003) provided evidence that for clear timbres (i.e. church organ), the upper bound of tolerable step size for retuning is about 3–4 cents.

Based on these findings we employed 3 cents as an engineering reference point for quantisation in retuning and recognise that the ability to distinguish pitches may vary depending on timbre, frequency register, listener and task. Therefore, a system's quantisation error should be at least one order of magnitude lower than this reference value. As such, both operation modes of the proposed system far exceed this requirement. The two byte format (Scale/Octave) of the MTS provides:

$$R_{MTS} = 200 / 2^{14} = 200 / 16384 \approx 0.0122 \text{ cents} \quad (1)$$

The RPN negotiated MPE mode, which limits pitch-bending sensitivity to ±2 semitones, provides:

$$R_{MPE} = 400 / 2^{14} = 400 / 16384 \approx 0.0244 \text{ cents} \quad (2)$$

As expected both values are much lower than the 3 cent threshold. The standard MPE allows a pitch-bending range of ±48 semitones on member channels; if one were to utilise the full 9600 cent range then the 14 bit resolution would be 9600 / 16384 ≈ 0.59 cents per step, providing significantly more "head room" than required. The proposed MPE mode utilises one twenty fourth of that range while being compliant with MPE standards through RPN 0 parameter negotiation, thus providing a 24x increase in resolution.

2.3 Prior Adaptive Tuning Systems

Recent browser-native microtonal tools have established the viability of in-browser microtonal practice. Leimma and Apotome (Allami, Diggins & Parviainen, 2021) provide browser-based environments for exploring transcultural tunings and for generating music over user-defined scales, including MIDI export to external synthesisers; their design emphasis is on decolonised, scale-as-substrate creative practice rather than on adaptive retuning under key change. Paulick (2022) demonstrated microtonal harmony with the Web Audio and Web MIDI APIs at WAC 2022, but again for fixed user-chosen scales. Neither system performs the per-note, key-anchored retuning under modulation that defines adaptive Just Intonation; the present work addresses this complementary gap.

Sethares (2002) created a Max system that generates minimum sensory dissonance in real-time by means of gradient descents. Code (2002) produced Groven.Max where keyboard inputs were mapped to a 36 tone JI scale using three MIDI channels. Hermode Tuning (Mohrlok, 2003) was incorporated into Logic Pro and Cubase and limited adjustments away from the reference equal temperament to +/-20 cents (+/-30 cents per pitch). Stange, Wick & Hinrichsen (2018) modelled adaptive tuning as a network of springs and generated MIDI output using fifteen separately pitch-bended channels. Volkov (2023) published Pivotuner, a VST3/AU plug-in with built-in MPE support and adjustable Key Determiners. Celik (2016)

produced Micro-MIDI: a real-time dynamic microtonal channel manager in Max/MSP. OddSound[1] MTS-ESP produced a software-only solution utilizing interprocess communications to share tuning data internally within a Digital Audio Workstation (DAW). Moussa (2006) identified MTS, manufacturer-specific SysEx, and polyphonic pitch bend as MIDI microtuning transport methods, noting that MTS has high nominal resolution (~0.0061 cents) but low hardware adoption. Indicated that although MTS possessed the highest nominal resolution (approximately .0061 cents), its hardware adoption remained minimal.

3. System Architecture

The system's initial response when activated is to confirm if the user has allowed the use of SysEx, the browser-imposed safety check that enables sending MTS messages. If SysEx permission is granted, the system can transmit MTS messages; audible success still depends on receiver support. Due to MIDI 1.0's unidirectional nature, there was no pre-existing way to determine whether an external device supported MTS; hence, SysEx messages were always treated as "fire and forget." Therefore, since users may find that the MPE fallback is necessary to achieve audible results from their instrument, users can select the MPE fallback manually in the interface and it will persist as a user preference. The permission-based design of the system provides a solution to both the Precision–compatibility tradeoff issue and the potential misuse of SysEx to send arbitrary payloads (i.e., firmware upgrades or factory reset commands) to connected devices. The MPE fallback utilises only Standard Channel Voice Messages that do not require SysEx permission, thus providing both a compatibility bridge and a security compliant option.

3.1 Primary Method: MIDI Tuning Standard (MTS)

For key-based adaptive just intonation, the system determines the current tonal centre and tunes the twelve pitch classes of the chromatic scale to pure ratio definitions relative to this keynote (stange et al., 2018). Because JI defines its pure ratios based on a specific keynote, a recalculation of these mappings must occur every time the key changes. Hence, the method employed here differs from approaches that resolve multiple dissonances simultaneously by changing individual note pitches independently (sethares, 2002). Since the entire chromatic scale is tuned as a unit, the most efficient MTS message type for this purpose would be the Scale/Octave tuning 2-byte message (sub-id#2 = 0x09), which retunes all twelve pitch classes across one or more specified channels within a single SysEx packet. Within MTS mode, scale/octave updates are used for lower-frequency key changes, while single-note messages are utilised for event-level, non-zero offset adjustments applied immediately prior to note-on events. While scale/octave updates provide encoded 14-bit cent offsets for each pitch class over a +/-100-cent range. Single-note updates utilise encoded data representing absolute note frequencies via the MTS Single Note Tuning Change format.

3.2 Fallback Method: Precision-Optimised MPE

If access to MTS is unavailable, the system defaults to utlising precision optimised MPE. As described in the MPE specification (MMA/AMEI, 2022, section 2.2.5), by default MPE Devices shall set member Channel Pitch Bend Sensitivity to 48 semitones. The system reduces this by use of registered parameter number (RPN) 0 (pitch bend sensitivity; see MIDI 1.0 detailed specification, 1996/2014, section registered parameter numbers) to 2 semitones; however, this results in a loss of expressiveness in favor of improved tuning accuracy of .0244 cents per step (a twenty-four fold increase). Utilising a single Lower Zone, under the MPE specification, Channel 1 serves as the manager channel for zone-wide messages while member channels 2 through 16 serve as individual note-bearing channels. The RPN 0 message is sent to all member channels with a data-entry MSB of 2 semitones to lock the sensitivity.

Since MPE allocates each sounding note to a dedicated member channel, polyphony is limited to fifteen voices and an allocation policy is required. The implementation employs a free pool of member channels and an activity queue. Upon receipt of a new note event, the next available member channel in the free pool is allocated for the new note event; if all member channels are occupied, the least recently activated note is voice-stolen. The LRU (least recently used) strategy is akin to traditional cache replacement strategies and applies a previously proposed multi-channel pitch bend allocation problem for browser native MPE(Moussa, 2006) to this application environment.

3.3 Proof-of-Concept: MIDI 2.0 Integration

A MIDI 2.0 proof-of-concept tuning engine has been developed that generates Universal MIDI Packet (UMP) structures for file export. However, because at present (Baratè & Ludovico, 2022), UMP packets cannot be sent by the Web MIDI API, this module operates presently in File Export Mode. The engine uses registered per-note controller #3 (Pitch 7.25) as defined in the MIDI 2.0 protocol specification; Pitch 7.25 represents a thirty-two bit number where seven bits represent an integer semitone and twenty-five bits represent the fractional portion thereof. Thus, theoretically, this method achieves a resolution of approximately .000003 cents – roughly a million times finer than the three-cent engineering reference value used in this study.

3.4 Internal Synthesis Fallback (Web Audio)

When neither external MTS nor MPE-capable hardware is configured — the common case for first-time residency or classroom use — the system falls back to an internal sample-based synthesiser implemented on the Web Audio API. The engine loads a multi-velocity Salamander Grand Piano sample set (~3-semitone spacing across A0–C8) and applies per-note JI cent offsets by modulating the *AudioBufferSourceNode.playbackRate* (effective ratio = 2^(semitones/12) × 2^(cents/1200)). This couples pitch and timbre because the playback-rate model shifts all partials together, so the internal path is intended for demonstration, pedagogy and zero-installation residency contexts rather than for research-grade tuning evaluation, for which external hardware synthesis remains the preferred output. Together with the external-MIDI paths described in §3.1–§3.3, this

[1] Cash, O., Gamble, D., & Hancock, D. (2021). MTS-ESP Mini user guide (Version 1.08). OddSound.

synthesis layer yields three operational deployment tiers — external MTS, external MPE and internal Web Audio — under a single browser session.

4. Comparative Evaluation

To confirm our design, we have compared the resolution of each tuning protocol (MTS and Optimised MPE) relative to a conservative 3-cent engineering reference for quantising retuning, as suggested by Vos (1982) and Mohrlok (2003). The data presented in able 1 reports the Engineering Margin as a multiple of the 3-cent reference (for example, 123x represents a resolution that is 123 times finer than the margin). We also show this same data on a logarithmic scale in Figure 1.

Table 1. Resolution analysis of the proposed tuning modes.

Method	Bit Depth	Range	Resolution (cents)	Engineering Margin
MPE (Standard)	14 bit	±48 semitones	~0.59	~5×
MPE (Optimised)	14 bit	±2 semitones	0.0244	~123×
MTS (Scale/Oct)	14 bit	±1 semitone	0.0122	~246×
MIDI 2.0	32 bit	Per-note absolute	0.000003	~1,000,000×

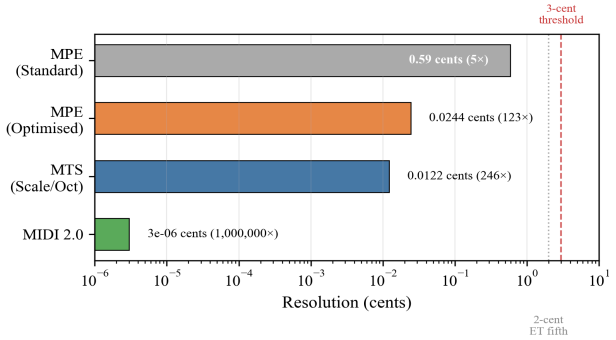

Figure 1. Logarithmic comparison of tuning resolution across MIDI protocols. The 3-cent perceptual threshold (dashed red line) and the 2-cent ET fifth deviation (dotted grey line) are shown for reference. All proposed modes substantially exceed the threshold.

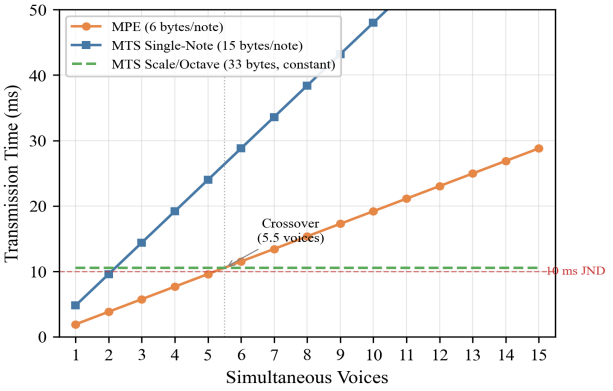

Figure 2. Transmission time as a function of simultaneous voices. MPE (6 bytes/note) has lower overhead for sparse textures, while MTS Scale/Octave (33 bytes, constant) becomes more efficient beyond ~5.5 voices. MTS Single-Note (15 bytes/note) scales linearly. The 10 ms percussive JND is shown for reference.

While both MTS and Optimised MPE provide quantisation steps that are far finer than the 3-cent reference, they do impose differing bandwidth costs. Given that Serial MIDI 1.0 operates at 31,250 bits per second, there exists an effective throughput of 3,125 bytes per second; therefore, each byte will take 0.32 ms to send. The summary table (Table 2) provides details regarding the composition and transmission timing of the messages used.

Table 2. MIDI message byte counts and transmission times.

Message Type	Structure	Bytes	Time (ms)	Source
MTS Scale/Octave 2-Byte	F0 7F id 08 09 ff gg hh [ss tt]×12 F7	33	10.56	CA-021
MTS Single-Note	F0 7F id 08 02 tt ll [kk xx yy zz] F7	12	3.84	CA-020
MPE Pitch Bend	En ll mm	3	0.96	MIDI 1.0
Note On/Off	9n/8n kk vv	3	0.96	MIDI 1.0

We can see clearly defined architectural trade-offs within these two approaches. MTS Scale/Octave allows for constant-time retuning. A single message consisting of 33 bytes can retune all 12 possible pitch classes regardless of the number of notes being played. This makes it suitable for use in adaptive Just Intonation applications that require key-based adaptation. On the other hand, MPE provides per-note granularity and lower latency in sparsely populated textures. It is capable of retuning individual notes independently of the other notes. However, as demonstrated in Figure 2, the overall MPE transmission time increases linearly based on polyphony.

Measurements were made empirically during performances by the first author of two disparate works. They include Bach's Well-Tempered Clavier, Book I, Prelude in C Major, BWV 846 (n=54 per mode), and Mozart's Rondo alla Turca, K.331/III (n=215 for MTS, n=190 for MPE). Each mode was evaluated using the same performance session for each piece. Timing was

achieved through the use of the W3C High Resolution Time API performance.now() method (W3C, 2019) and the Web MIDI API message.timeStamp property (Wilson & Wilson, 2025) for hardware timestamps. Because latency distributions were non-normal, Mann-Whitney U tests (Mann & Whitney, 1947) were used for non-parametric between-mode comparisons for testing non-parametric hypotheses, while employing Welch's t-tests (Welch, 1947) as secondary robustness checks.

Table 3. Empirical latency analysis.

Metric	MTS Single-Note	MPE Pitch Bend
Bach, Prelude in C, BWV 846 (n = 54 per mode)		
M (95% CI)	4.32 ms [3.83, 4.81]	2.31 ms [2.20, 2.42]
SD	1.81 ms	0.40 ms
Median (IQR)	5.10 [3.52, 5.20]	2.22 [2.12, 2.22]
Min / Max	1.06 / 7.50 ms	2.02 / 3.62 ms
Bytes per Note	11.9 (74% at 15, 26% at 3)	6.0 (100% at 6)
Callback Latency	+0.25 ms	+0.23 ms
Mozart, Rondo alla Turca, K. 331/III (n = 215 MTS, 190 MPE)		
M (95% CI)	4.21 ms [3.99, 4.42]	2.16 ms [2.13, 2.20]
SD	1.63 ms	0.24 ms
Median	5.00	2.12
Min / Max	0.96 / 6.40 ms	1.92 / 3.62 ms
Skewness	−1.31	3.77
Shapiro–Wilk	p < .0001 (non-normal)	p < .0001 (non-normal)
Bytes per Note	12.4 (78% at 15, 22% at 3)	6.0

Note. Total estimated latency is a composite value combining measured JavaScript processing latency (performance.now()), and calculated MIDI transmission latency (byte count x 0.32 ms/byte). Callback latency (message.timeStamp) is shown as an additional overhead value. All latency distributions were found to be non-normal (Shapiro-Wilk p < .0001). The first ten samples per mode in BWV 846 were discarded due to warm-up. This metric does not account for device input scanning, OS driver latency, synthesiser response, audio buffering or acoustic output.

Inferred from our analysis of BWV 846, we found Mann-Whitney U = 662, p<.001, and Welch's t(58.1)= –7.986, p<.0001, Cohen's d = 1.537, and a difference 95% confidence interval of [-2.51,-1.51]. Similarly for K.331/III we obtained Welch's t(224.6)= –18.18, p<.0001, Cohen's d = 1.706, and Mann-Whitney U = 8745, p<.0001. The software processing latency remained less than one millisecond for both modes and both pieces, suggesting that the LRU channel allocation logic necessary for MPE has no measurable overhead in this benchmark. Furthermore, combining the software processing time and wire transmission time, we determined that MPE exhibited approximately a 47-49% lower total estimated latency than MTS Single-Note. MTS Single-Note would theoretically require 15 bytes per note (12 bytes SysEx + 3 bytes Note On), but since the system only sends a Single-Note SysEx when a note needs to be tuned with a non-zero offset value; therefore, observed averages in terms of bytes transmitted per note were reduced relative to the 15-byte worst case for MTS Single-Note. Specifically for BWV 846(C major) 26% of note events resulted in a note being placed upon the tonic (a 1:1 ratio and zero cent offset), resulting in a per-note average of ~11.9 bytes (~74% at 15 bytes and ~26% at 3); similarly for K.331/III(A major/minor), where 22% of the note events occurred on the tonic producing a ~12.4 bytes per note average; whereas MPE consistently required only 6 bytes (3-byte Pitch Bend + 3-byte Note On). As both per-note modes remain below generally recognised action-sound latency thresholds (although the current measurement does not consider device input scanning, OS driver latency, synthesiser response, audio buffering, or acoustic output), these findings are depicted visually in Figures 3-6.

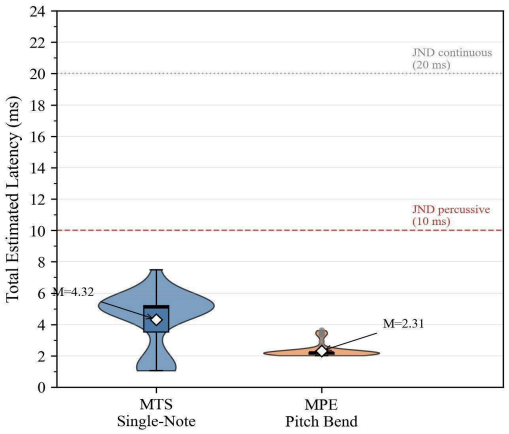

Figure 3. Violin plot of total estimated latency for BWV 846. MTS Single-Note (M = 4.32 ms) vs. MPE Pitch Bend (M = 2.31 ms). Both modes fall below the 10 ms and 20 ms reference latency lines used for comparison; these lines are not perceptual validation of the complete system.

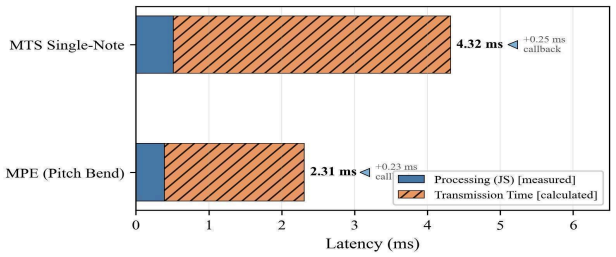

Figure 4. Latency breakdown for BWV 846 showing processing (JS) and calculated transmission time components. Software processing stays below 0.3 ms in both modes; the latency difference is dominated by transmission time due to differing byte counts.

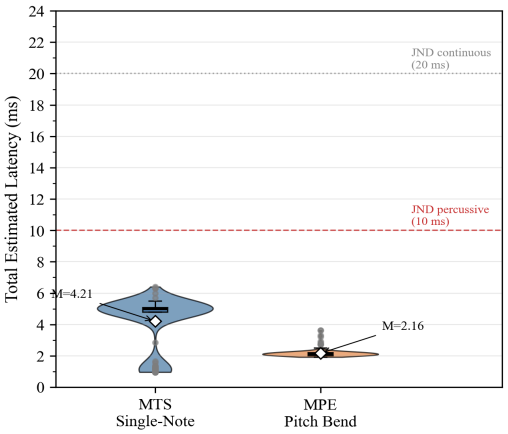

Figure 5. Violin plot of total estimated latency for Mozart, Rondo alla Turca, K. 331/III. MTS Single-Note (M = 4.21 ms) vs. MPE Pitch Bend (M = 2.16 ms). The bimodal MTS distribution and tight MPE clustering observed in BWV 846 are replicated.

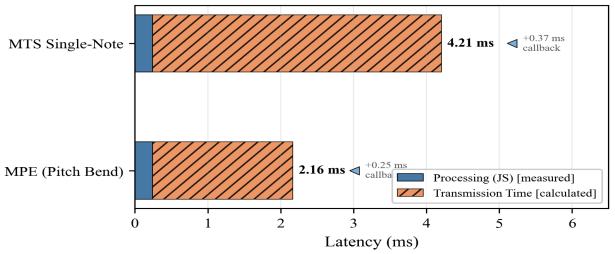

Figure 6. Latency breakdown for K. 331/III showing processing (JS) and calculated transmission time components. The pattern mirrors BWV 846: software processing remains sub-millisecond, with the latency difference driven by transmission overhead.

Table 4. Comparison with existing adaptive tuning systems.

System	Protocol	Environment	HW Output	Fallback	Resolution
Adaptun (Sethares, 2002)	Multi-ch PB	Max/MSP	Yes	No	~1.56 cents
Groven.Max (Code, 2002)	Multi-ch routing	Max/MSP	Yes	No	Fixed 36-tone
Hermode (Mohrlok, 2003)	Proprietary	DAW	No	No	Proprietary
Micro-MIDI (Celik, 2016)	Multi-ch PB	Max/MSP	Yes	No	~3.12 cents
JI App (Stange et al., 2018)	Multi-ch PB	Standalone	Yes	No	~1.56 cents (ext.)
MTS-ESP[2]	IPC/Shared Mem	DAW Plugins	No	No	0.0061 cents
Pivotuner (Volkov, 2023)	MPE	DAW Plugins	No	No	~0.59 cents
Proposed System	MTS/MPE/MIDI 2.0	Browser (Web MIDI)	Yes	Yes	0.012 / 0.024 cents
Leimma/Apotome (Allami et al., 2021)	Web MIDI (fixed scales)	Browser	Yes	n/a (non-adaptive)	User-defined scale resolution

Table 4 illustrates that previous systems like Sethares (2002) and Stange et al. (2018) provided more advanced algorithmic tuning capabilities and OddSound MTS-ESP[2] removed polyphony constraints through inter-process communication. The architectural innovation of the current study lies in providing a combination of three features:

[2] Cash, O., Gamble, D., & Hancock, D. (2021). MTS-ESP Mini user guide (Version 1.08). OddSound.

1. A Hierarchical fallback between two standardised MIDI tuning protocols,
2. Direct external hardware output from inside a web-browser utilising the Web MIDI API and,
3. A native deployment model running from a web-browser requiring no installation.

5. Discussion

Both methods achieve protocol quantisation greater than the engineering reference used here, along with sub-5 ms estimated processing plus transmission latency. We point out three design factors related to the ubimus agenda and three limitations.

Hierarchical fallback for heterogeneous settings. Moussa (2006) demonstrated the limitation of assigning channels to MIDI ports and introduced an architecture based upon routing to three separate physical MIDI output channels — one for instruments capable of producing polyphonic after touch, one for MTS-enabled instruments, and one for all other instruments. As stated earlier, the proposed system has analogous graceful degradation through software level protocol switch in a single browser session utilising SysEx permissions as the fall back criteria. This creates a new security layer in addition to being compatible with desktop systems because Browsers block SysEx by default. Therefore, the precision optimised MPE mode can function as a compatibility bridge while providing a secure compliant solution. Both modes have achieved protocol quantisation much better than the engineering reference utilised in this research and there is little measurable difference between the two modes when compared to widely accepted action/sound latency values (Mäki-Patola & Hämäläinen, 2004; Jack et al., 2018). In light of cross-piece consistencies found between BWV 846 and K. 331/III (Cohen's d = 1.54 – 1.71), we believe that the absolute latency difference is small relative to commonly cited action-sound latency ranges, although the between-mode statistical effect is large. Therefore, the software-based fallback procedure compliments the hardware based flexibility provided by Keller et al. (2025)'s ubimus plugging framework.

Accessibility and Participatory Creativity. Because we are using a model that involves zero installations for the deployment, this aligns well with the ubimus goal of increasing the capacity of many diverse participants to create and participate in many music-making activities (Keller et al., 2019). We provide adaptive JI capabilities across all types of MIDI compatible keyboard and synthesizer devices via protocol compatibility and configuration. Thus, the model provides low barrier entry for non-expert users who may not have access to commercial digital audio workstation (DAW) applications or proprietary plug-ins. Previous architectures intended to enable use of external MIDI hardware, i.e., Stange et al. (2018), Celik (2016), Code (2002), Moussa (2006); however, these architectures were either DAW dependent or required specialized hardware. More recently, several browser-native microtonal tools have been developed to allow for in-browser microtonal practice with external MIDI export, specifically Allami, Diggins and Parviainen (2021) and Paulick (2022). However, while those systems have provided examples of viable in-browser microtonal practice and external-MIDI export; they provide fixed user selected scales, but do not adaptively retune based upon changes in key. To our knowledge, this current system is the first browser-native adaptive JI tuner which includes key-anchored retuning, a hierarchical MTS/MPE protocol fallback, direct external MIDI hardware output and a MIDI 2.0 UMP proof-of-concept.

Educational and Residency Relevance. Collaborative discovery workshops/residency models such as GIM emphasise the collaborative learning experience of ubimus concepts among varying degrees of expertise. Since no software needs to be installed prior to execution, this architecture can easily be run in classrooms and on students' personal devices with minimal administrative burden. Thus, micro-tonality becomes a feasible means of exploring expressiveness in a controlled manner versus an area of specialization; e.g., students can compare beating at the 12-TET major third with a pure 5:4 ratio on their own keyboards within minutes of opening a webpage. This approach to exploring micro-tunality is consistent with the residency's educational mission of integrating international graduate students into active ubimus practices.

Beyond Common-Practice Repertoire. While the empirical evaluation reported in §4 utilises two Western classical works to evaluate modes against a known repertoire baseline, the mechanisms that govern the system – namely, key-anchored, integer-ratio retuning with hierarchical protocol fallback – are completely independent of common-practice notation and accommodate a broader set of potential creative ubimus contexts: drone-based and ambient practice, where lack of modulation causes pure-interval tunability to dominate the aesthetic affordances; generative and algorithmic struck-string interaction in the sense of Keller, Jagwani and Lazzarini (2025), where tuning becomes a real-time compositional aspect; and explorations of non-Western harmonic systems. Most recently, computational research has focused on developing tools for alternative temperaments — Dalmazzo, Déguernel and Sturm's (2025) Max-for-Live tool for 53-TET modal interchange, and Nikkar, Sacchetto and Rottondi's (2025) self-organising map navigation of microtonal scales rooted in the Iranian Radif — operate in fixed alternative TETs rather than adaptive JI, and operates in DAW or analytical contexts. It is theoretically possible to extend the browser-native protocol substrate described above to such systems by simply replacing the JI scale-ratio table with an equivalent tuning vector (the 12-class MTS Scale/Octave message and ±2 semitone MPE pitch-bend window each have sufficient headroom to support 22- or 53-TET pitch sets, depending on MPE channel assignment.) We note this possibility here as one of future direction possibilities rather than as part of what was empirically demonstrated in §4, since the underlying empirical work in §4 continues to operate under key-based JI.

Limitations. Four significant technical limitations exist. Firstly, due to the unidirectional nature of MIDI 1.0, automatic determination of MTS capabilities is impossible (Luke & Ludovico, 2024); thus, the system relies on the user verifying that MTS was successful and selecting MPE manually otherwise. Secondly, MPE mode resolution decreases significantly from 0.024 cents to approximately 0.59 cents if synthesisers fail to interpret RPN 0 as specified. Lastly, timing jitter from USB-MIDI interfaces may exceed predictions made from serial byte-count analyses. Additionally, while there are currently some implementations available for the Web MIDI API, which are still in W3C working draft status, there are currently very

few implementations of the Web MIDI API outside of chromium-based browsers, which could potentially create difficulties for novice users attempting to configure MIDI (Baratè & Ludovico, 2022). Fourthly, the evaluation is technical rather than perceptual: we did not include a listening study nor direct acoustic measurements of action-to-sound latency across multiple devices and synthesisers.

We anticipate addressing these limitations over the next several iterations in three areas: (1) real-time MIDI 2.0 UMP transmission, removing MPE channel-allocation constraints and the 15-voice polyphony limit imposed by the current Web MIDI API, once it supports MIDI 2.0 UMP; (2) using symbolic representations for calculating just intonation, allowing us to remove the latency associated with detecting keys; and (3) using MIDI 2.0 Capability Inquiry (MIDI-CI), allowing us to automatically negotiate protocols, thereby transforming the hierarchical fallback into a completely automated device discovery process should they support it, as well as future browser MIDI APIs.

6. Conclusion

This paper presents an adaptive Just intonation framework based upon a multi-protocol, browser native architecture. This structure allows for the combination of MTS Scale/Octave and Single Note message types, optimised MPE Pitch Bend, and a Proof of Concept implementation of a MIDI 2.0 UMP to provide compatibility, optimisation and reduced deployment friction. In addition to a hierarchical fallback model where MTS Scale/Octave Tuning will be employed when there is availability of System Exclusive permissions and a Precision Optimised MPE mode which locks the sensitivity of the pitch-bending to +/-2 Semitones via RPN 0 negotiation when either MTS is unavailable or ineffective, a forward compatible proof-of-concept implementation of the MIDI 2.0 engine is provided. The between-mode latency differences were large (Cohen's d = 1.54-1.71), while both modes exceeded the engineering resolution reference: MTS achieved approximately 0.012 cents (~246 times the 3 cent reference value) and Precision Optimised MPE achieved approximately 0.024 cents (~123 times the reference value). Estimated processing-plus-transmission latencies for both tested modes were less than 5 ms. Unlike most other previous studies, we view this study as being primarily structural (as opposed to algorithmic): to date, no previously developed adaptive tuning systems have integrated hierarchical fallbacks between standardised MIDI tuning protocol standards, direct external hardware output from a browser environment, and zero-installation browser-native deployment. We position this contribution within the ubimus research agenda on interactive string timbres (Keller et al., 2025) and accessible creative participation (Keller et al., 2019); the aim of this contribution is to make adaptive Just Intonation a practical option for a larger number of musicians, students, and participants in ubimus residencies like those found at GIM.

Acknowledgements

This work is part of Instant Harmonies, a doctoral research project at Maynooth University that aims to make just-intonation tools accessible, adaptable, and usable with existing MIDI equipment without specialised hardware or software. The ultimate goal of the Instant Harmonies project is to enable greater access to microtonal music creation by developing a free, web-based platform that allows musicians who are both professionals and non-professionals to create Just Intonated music utilising their current MIDI equipment and do not require specialised hardware or software.

References

Allami, K., Diggins, S., & Parviainen, T. (2021). Leimma & Apotome [Web applications]. Counterpoint, with Khyam Allami. https://isartum.net/leimma | https://isartum.net/apotome (Recipient of the Isao Tomita Special Prize, Ars Electronica 2021.)

Baratè, A., & Ludovico, L. A. (2022). Web MIDI API: State of the art and future perspectives. Journal of the Audio Engineering Society, 70(11), 918–925. https://doi.org/10.17743/jaes.2022.0028

Celik, S. (2016). Micro-MIDI: A real time, dynamic microtonal MIDI application. International Journal of Engineering and Science Invention, 5(9), 1–7.

Code, D. L. (2002). Groven.Max: An adaptive tuning system for MIDI pianos. Computer Music Journal, 26(2), 50–61.

Dalmazzo, D., Déguernel, K., & Sturm, B. (2025). A computer application to explore 53-tone equal temperament harmonies through modal interchange. In Proceedings of the International Conference on New Interfaces for Musical Expression (NIME 2025), Canberra, Australia. https://doi.org/10.5281/zenodo.15698843

Hasegawa, R. (2013). Just intonation: An aesthetic and theoretical renewal. In N. Donin & L. Feneyrou (Eds.), Théorie et composition musicales au vingtième siècle (pp. 1499–1531). Symétrie.

Jack, R. H., Mehrabi, A., Stockman, T., & McPherson, A. (2018). Action–sound latency and the perceived quality of digital musical instruments: Comparing professional percussionists and amateur musicians. Music Perception, 36(1), 109–128. https://doi.org/10.1525/mp.2018.36.1.109

Keller, D., Schiavoni, F., & Lazzarini, V. (2019). Ubiquitous music: Perspectives and challenges. Journal of New Music Research, 48(4), 309–315. https://doi.org/10.1080/09298215.2019.1659828

Keller, D., Jagwani, A., & Lazzarini, V. (2025). The ubimus plugging framework: Deploying FPGA-based prototypes for ubiquitous music hardware design. Computers, 14(4), 155. https://doi.org/10.3390/computers14040155

Luke, S., & Ludovico, L. A. (2024). MIDI 1.5, so to speak. In Proceedings of the Sound and Music Computing Conference (SMC 2024) (pp. 373–382).

Mann, H. B., & Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. The Annals of Mathematical Statistics, 18(1), 50-60. https://doi.org/10.1214/aoms/1177730491

Mäki-Patola, T., & Hämäläinen, P. (2004). Latency tolerance for gesture-controlled continuous sound instruments without tactile feedback. In Proceedings of the International Computer Music Conference (ICMC) (pp. 409–416).

MIDI Manufacturers Association. (2014). MIDI 1.0 Detailed Specification (Document version 96.1, third edition, February 2014). Los Angeles, CA: MMA. (Original work published 1996.)

MIDI Manufacturers Association & Association of Musical Electronics Industry. (2022). MIDI Polyphonic Expression (MPE) specification (Version 1.1, Document RP-053). MMA/AMEI.

Muzzulini, D. (2021). Isaac Newton's microtonal approach to just intonation. Empirical Musicology Review, 15(3–4), 223–248. https://doi.org/10.18061/emr.v15i3-4.7647

Mohrlok, W. (2003). The Hermode Tuning system. https://sethares.engr.wisc.edu/paperspdf/hermode.pdf

Moussa, A. S. (2006). Perception-based microtuning over MIDI networks. IEEE Multimedia, 13(1), 56–64. https://doi.org/10.1109/MMUL.2006.18

Nikkar, K., Sacchetto, M., & Rottondi, C. (2025). Mapping the neighborhood of microtonal music scales using self-organizing maps. In Proceedings of the 20th International Audio Mostly Conference. ACM. https://doi.org/10.1145/3771594.3771618

Paulick, D. (2022). Exploring microtonal harmony with Web Audio and Web MIDI. In Proceedings of the Web Audio Conference (WAC 2022). Université Côte d'Azur, Cannes, France. ISSN 2663-5844. https://webaudioconf.com/proceedings/

Plomp, R., & Levelt, W. J. M. (1965). Tonal consonance and critical bandwidth. Journal of the Acoustical Society of America, 38(4), 548–560. https://doi.org/10.1121/1.1909741

Schwär, S., Rosenzweig, S., & Müller, M. (2021). A differentiable cost measure for intonation processing in polyphonic music. In Proceedings of the 22nd ISMIR Conference (pp. 626–633).

Sethares, W. A. (2002). Real-time adaptive tunings using Max. Journal of New Music Research, 31(4), 347–355. https://doi.org/10.1076/jnmr.31.4.347.14163

Shapiro, S. S., & Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). Biometrika, 52(3-4), 591-611. https://doi.org/10.1093/biomet/52.3-4.591

Stange, K., Wick, C., & Hinrichsen, H. (2018). Playing music in just intonation: A dynamically adaptive tuning scheme. Computer Music Journal, 42(3), 47–62. https://doi.org/10.1162/comj_a_00478

Su, R., Timoney, J., & Keller, D. (2024). Just intonation and inharmonicity in struck-string interaction. In Proceedings of the Ubiquitous Music Symposium (UbiMus 2024), Macau, China.

Su, R., Timoney, J., & Keller, D. (2025). MIDI adaptive tuning strategies by means of AI-based struck-string interaction in ubimus. In Proceedings of the Ubiquitous Music Symposium (UbiMus 2025).

Volkov, D. (2023). Pivotuner: automatic real-time pure intonation and microtonal modulation. arXiv preprint arXiv:2306.03873.

Vos, J. (1982). The perception of pure and mistuned musical fifths and major thirds: Thresholds for discrimination, beats, and identification. Perception & Psychophysics, 32(4), 297–313. https://doi.org/10.3758/BF03206236

Vos, J. (1984). Spectral effects in the perception of pure and tempered intervals: Discrimination and beats. Perception & Psychophysics, 35(2), 173–185. https://doi.org/10.3758/BF03203897

W3C. (2019). High Resolution Time Level 2. W3C Recommendation. https://www.w3.org/TR/hr-time-2/

Weiser, M. (1991). The computer for the 21st century. Scientific American, 265(3), 94–104.

Welch, B. L. (1947). The generalization of ‘student’s’ problem when several different population variances are involved. Biometrika, 34(1–2), 28–35. https://doi.org/10.1093/biomet/34.1-2.28

Wilson, C., & Wilson, M. (2025). Web MIDI API (W3C Working Draft). World Wide Web Consortium. https://www.w3.org/TR/webmidi/