Audio Engineering Society

Convention Paper

Presented at the AES 161st Convention
2026 October 30-November 1, Nashville, TN, USA

This paper was peer-reviewed as a complete manuscript for presentation at this convention. This paper is available in the AES E-Library (http://www.aes.org/e-lib), all rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

Voice-Controlled Audio Routing in Professional Dante Networks Using Large Language Models and the Model Context Protocol

Johannes Kreitel-Lüschow[1]

[1]*Technische Hochschule Ostwestfalen-Lippe (TH OWL), Detmold, Germany*

Correspondence should be addressed to Johannes Kreitel-Lüschow (johannes.lueschow@th-owl.de)

ABSTRACT

Professional Audio-over-IP (AoIP) networks based on Dante are widely deployed in broadcast, live production, and educational facilities, yet their management interfaces demand expert knowledge of network topology and device-naming conventions. This paper presents a peer-reviewed implementation of a voice-controlled Dante routing system built on Large Language Models (LLMs) and the Model Context Protocol (MCP). We contribute a purpose-built Dante MCP Server exposing six structured tools for querying and modifying audio routes through the Dante Domain Manager (DDM) GraphQL API, integrated with local automatic speech recognition and commercial LLM tool use, and protected by a domain-level safety layer that prevents changes to production infrastructure. The pipeline was validated on a real, multi-domain Dante infrastructure of approximately 39 devices; end-to-end latency is approximately 1770 ms, dominated by LLM inference and acceptable for non-real-time supervisory routing. A controlled within-subject study (N=10; five experts, five non-experts) compared voice against conventional graphical user interface (GUI)-based routing using the NASA Task Load Index (NASA-TLX) and the System Usability Scale (SUS). Contrary to our hypothesis, voice imposed significantly higher cognitive load (RTLX 24.1 vs. 11.6; Wilcoxon p=.002, r=.89) and lower usability (SUS 66 vs. 80; p=.008); non-experts nonetheless completed 90% of voice tasks, and a modality-specific failure signature emerged in real-time correction (four voice failures vs. zero GUI). We trace these failures to delete-command disambiguation, a concrete and addressable design target rather than a limit of voice control. The Dante MCP Server is LLM-agnostic and will be released as open source.

1 Introduction

1.1 Motivation

Professional Audio-over-IP (AoIP) networks based on the Dante protocol have become the de-facto standard for digital audio distribution in broadcast studios, live production venues, theaters, and educational facilities. With over 4,600 Dante-enabled products from more than 490 manufacturers and an installed base exceeding seven million audio and video devices [1], Dante manages audio signal flow across standard IP infrastructure. In professionally managed installations — broadcast facilities, university campuses, corporate AV environments, and houses of worship — network governance is handled through the Dante Domain Manager (DDM), providing centralized multi-domain administration, user authentication, and programmatic API ac-

cess. These are precisely the environments where operational complexity is highest: many devices across multiple subnets, multiple operators with different permission levels, and routing decisions that must be made reliably under time pressure.

Despite this professional management layer, the DDM graphical interface and its companion tool Dante Controller require users to understand network topology, device naming conventions, and bidirectional channel assignment schemes before any routing action can be taken—procedural overhead that introduces measurable cognitive load and error risk in time-critical live production scenarios [2]. Graphical interfaces have repeatedly been shown to impose unnecessary demands on operators during such tasks [3], while voice-based interfaces can reduce error rates and completion times in analogous professional contexts [4].

The emergence of LLMs capable of structured tool use presents a fundamentally new option: rather than training users to operate a graphical user interface (GUI), the system could be trained to understand users. Natural language commands such as "Route the stage left microphone to the front-of-house recorder" express routing intent directly, without navigating menus or recalling device identifiers. This paper investigates whether such an interface can be built for professional Dante networks and whether it reduces cognitive load in practice.

1.2 Research Gap

Recent work has explored LLM-based control across several domains adjacent to this problem—audio content processing [5, 6], generic IP network configuration [7, 8], device parameter control via the Model Context Protocol [9, 10], and voice control under high cognitive load [11]—as detailed in Section 2. None of these systems, however, address the control of professional AoIP routing infrastructure. MCP2OSC, the closest prior work, operates at the device parameter layer (volume, equalization, reverb) rather than modifying signal routing between devices across the network. Concurrent with this work, DVIGear announced a commercial MCP integration for their DisplayNet video distribution platform [12], confirming the broader relevance of the approach, but targeting video infrastructure without peer-reviewed validation.

To the best of the authors' knowledge, no prior peer-reviewed work has described an LLM- or MCP-based system for controlling professional Dante audio routing infrastructure. This gap motivates the present work.

1.3 Contributions

This paper makes the following contributions:

1. **Dante MCP Server:** a purpose-built FastMCP server exposing six structured tools and one network resource for querying/modifying Dante audio routing via the DDM GraphQL API—a vendor-neutral bridge between any MCP-compatible LLM and a Dante network, with built-in safeguards against protected-domain modification.
2. **End-to-end voice-to-route pipeline:** integrating Whisper speech recognition [13], Claude LLM tool use, the Dante MCP Server, and the DDM GraphQL API, validated on a real 39-device Dante infrastructure.
3. **Empirical evaluation:** a controlled user study (N=10) comparing the system against conventional GUI-based Dante management via the NASA Task Load Index (NASA-TLX) [14], the System Usability Scale (SUS) [15], and task completion across expert/non-expert groups.
4. **Open-source release** of the Dante MCP Server upon paper acceptance, enabling replication and adoption across DDM-based Dante installations.

2 Related Work

LLM-based orchestration has advanced rapidly in audio content and effects processing [5, 6], real-time device parameter control via OSC [10], and multi-agent editing workflows [16], but these systems operate at the content or device-parameter level rather than network routing. In generic IP infrastructure, LLMs demonstrate zero-touch configuration [7, 8, 17] and agentic, non-expert-accessible control [18]—paralleling our motivation; natural language has also been translated into OPC UA calls for manufacturing machinery (96–98% accuracy) [19], with MCP shown applicable to industrial control interfaces [20]. The Model Context Protocol [9] is an open, JSON-RPC-based standard for LLM-to-tool integration with rapidly growing adoption [21]; the finding that precisely bounded tool sets significantly outperform generic API passthroughs in accuracy [22] motivated our six domain-specific tools,

following the systematic API-wrapping methodology of [23]. Separately, voice interfaces remain responsive under extreme cognitive load in AR surgical simulations [11], and domain-specific speech errors can be mitigated via full-text LLM post-processing [24]—directly applicable to Dante device names.

In professional audio HCI, interface design measurably affects cognitive load [2, 25], though AI mixing assistance remains confined to content-level tasks [26]; AudioMiXR [3] found no significant expert/non-expert workload difference, directly informing our H3. Dante's Domain Manager (DDM) remains the only officially supported programmatic routing interface; AES67 and AES70/OCA [27] provide complementary open standards but neither addresses routing. MCP2OSC, the architecturally closest prior work, changes device parameter values (e.g., /audio/player/volume 0.5) but does not modify routing paths—no prior peer-reviewed work addresses LLM-mediated control of AoIP routing infrastructure (Section 1.2).

3 System Architecture

The proposed system translates natural language utterances into persistent audio routing changes in a professional Dante network. Figure 1(a) illustrates the end-to-end pipeline, which consists of five processing stages: speech capture, transcription, LLM-based intent resolution, MCP-mediated API dispatch, and DDM execution.

3.1 Speech Recognition and Intent Resolution

Audio input is captured via push-to-talk (spacebar hold, PyAudio, 16 kHz mono) and transcribed locally by OpenAI Whisper [13] (base model, German), preserving audio confidentiality and avoiding a streaming audio API. The base model was chosen over larger variants to minimize latency while retaining sufficient accuracy for Dante's structured device-name vocabulary.

The transcribed command is passed to the Claude API (claude-haiku-4-5) with a system prompt encoding the current Dante topology (domain names, device identifiers, channel assignments); the model responds with a structured tool call. Haiku was chosen over larger Claude models to minimize round-trip latency, since the bounded routing-command vocabulary does not require extended reasoning capacity. The prompt follows a role-play strategy—the LLM acts as a Dante network controller that resolves ambiguous device references against the known device list before issuing any tool call—mirroring the approach found effective for OSC address space generation [10] and consistent with findings that external context integration improves domain-specific recognition accuracy [28]. The complete system prompt template will be made publicly available upon acceptance for independent verification.

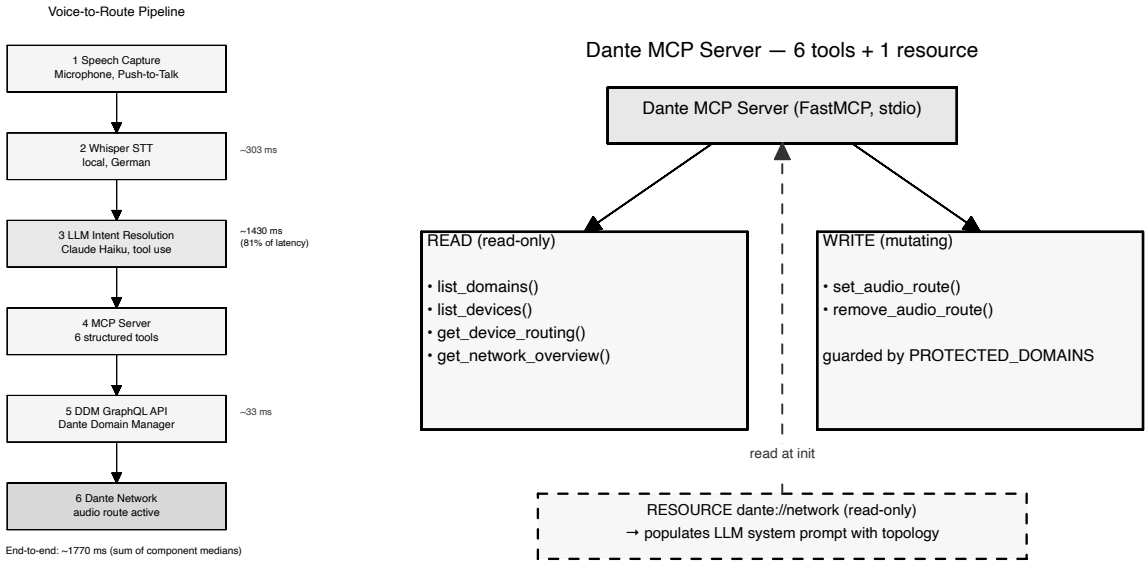
3.2 Dante MCP Server and DDM Integration

The Dante MCP Server, this work's central architectural contribution, is implemented in Python (FastMCP) and exposes six tools plus one resource (`dante://network`, providing read access to the complete topology) over the MCP stdio transport, making the Dante network accessible to any MCP-compatible LLM client (Figure 1b). The tool vocabulary was deliberately constrained to six operations rather than a generic API passthrough, following the finding that precisely bounded tools are a primary determinant of LLM accuracy in MCP integrations [22]; the transformation from the proprietary DDM GraphQL API [29] to this structured tool set follows the methodology of [23].

Implementation required handling specific DDM authentication constraints, discovered empirically—including a non-standard `User-Agent` string and raw API key header formatting (no OAuth Bearer prefix)—as well as traversing the domain hierarchy for device queries rather than a top-level query.

3.3 Safety Architecture

The server implements three safeguards for deployment alongside productive broadcast infrastructure: (1) **domain-level write protection**—the constant `PROTECTED_DOMAINS` (default {"Radio", "Audio", "Video"}) is checked in all write tools, returning an explicit refusal for any protected-domain target without issuing a mutation, while the isolated test domain remains available for experiments; (2) **credential isolation**—the DDM API key is loaded only from an environment variable and never appears in logs, LLM context, or tool return values; and (3) **network-level isolation**—the DDM endpoint is reachable only within the institutional network.



(a) End-to-end voice-to-route pipeline.

(b) Dante MCP Server: six read/write tools and one topology resource; write operations guarded by `PROTECTED_DOMAINS`.

Fig. 1: System pipeline (a) and MCP tool surface (b). LLM inference dominates the ∼1770 ms end-to-end latency (∼81% of the total).

3.4 Prototype Variants and Test Environment

The system evolved through four iterative prototypes sharing the same MCP server core, culminating in the macOS menu-bar application with push-to-talk and spoken confirmation used for the study. Evaluation was performed on an isolated test domain (six devices) within the host infrastructure, strictly separated from three protected production domains (Radio, Audio, Video) to prevent accidental modification of live infrastructure.

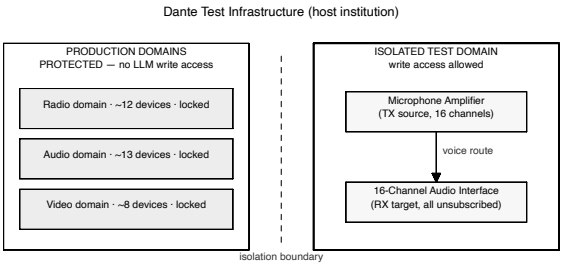


Fig. 2: Test infrastructure: three protected production domains and one isolated test domain used for all routing experiments.

4 Evaluation

The evaluation addresses two distinct questions: (1) what are the latency characteristics of the individual system components and the end-to-end pipeline, and (2) does the voice interface reduce cognitive load and improve task outcomes compared to conventional GUI-based Dante management?

4.1 System Performance: Latency Measurement

Latency was characterized for the complete pipeline under operational conditions on the host institution's Dante network. Four components were measured independently (n=10) using Python's `time.perf_counter()`: Whisper STT (audio capture to transcription), Claude API tool use (request dispatch to tool-call response), DDM GraphQL mutation (POST to acknowledgment), and end-to-end text-pipeline time (text input, bypassing Whisper, through mutation confirmation). Table 1's voice-pipeline total is the sum of these medians, not a single continuous trial; it is corroborated by the measured text-pipeline median (1467 ms) plus Whisper (303 ms), converging on the same ∼1770 ms. The native Dante audio transport latency (0.25–10 ms) is unaffected by routing

changes and excluded. Table 1 reports medians and standard deviations (02.06.2026).

LLM inference dominates the pipeline (∼81% of the ∼1770 ms voice total; DDM overhead <2%), appropriate for supervisory administrative tasks (e.g., scene changes, setup) rather than real-time performance mixing—aligning with Dewey's distinction between linear live and non-linear studio mixing [25]. Operators issue a command and await visual confirmation in the Dante Controller GUI, a deliberate process accommodating the ∼1.8s budget without impairing task completion, consistent with the task-completion-time efficiency framework of Wanderley and Orio [30]; this is unlike the <250 ms benchmark from surgical simulations [11], which applies to immediate haptic/visual feedback rather than the visual, asynchronous confirmation used here. Note that benchmarking Whisper against silent audio yields artificially high values (∼4800 ms, from exhausted decoding iterations without an early-exit trigger); the reported 305 ms median reflects valid measurement with representative 6-second spoken input.

4.2 User Study

4.2.1 Hypotheses

Three hypotheses guide the user study:

H1: Non-expert operators can successfully complete Dante audio routing tasks using the voice interface, achieving a Task Completion Rate of at least 80%.

H2: The voice interface results in significantly lower cognitive load than GUI-based routing, as measured by the NASA-TLX (RTLX; $p < 0.05$), with the effect expected to be most pronounced for Tasks 2 and 3, where routing complexity exceeds the procedural advantage of a familiar GUI.

H3: There is no significant difference in cognitive load between expert and non-expert participants when using the voice interface, indicating that the interface equalizes performance across expertise levels.

4.2.2 Participants

Participants were recruited from staff and students at the host institution. Two groups were formed based on prior Dante experience: experts (at least one year of hands-on Dante network operation) and non-experts (no prior Dante experience). A target of N=10 participants (5 per group) was defined as appropriate for an exploratory pilot study in a professional audio context, consistent with the sample sizes used in comparable HCI evaluations of audio interfaces [3] (e.g., AudioMiXR). While N=10 limits statistical power for general population claims, this sample size suffices to identify significant usability trends and latency bottlenecks. Effect sizes (rank-biserial correlation, r) are reported alongside p-values to provide meaningful magnitude estimates even with small samples.

4.2.3 Apparatus and Setup

All sessions were conducted in a single acoustically treated surround studio under identical conditions for every participant, in German. Voice input was captured with a t.bone GM 5212 gooseneck condenser microphone (supercardioid, table-mounted, ∼30 cm distance) via a MOTU M4 USB interface, toggled by push-to-talk, with the session simultaneously recorded to a Zoom F1-LP field recorder (48 kHz/16-bit WAV) for transcript analysis. The voice-control server ran on a MacBook Pro (Apple M2 Pro, 16 GB RAM) with on-device Whisper "base" STT and Claude claude-haiku-4-5 for intent parsing (requiring network access to the DDM and LLM API); system responses were announced via macOS text-to-speech. The GUI condition used the same MacBook with a 68.6 cm (27-inch) 4K display and mouse. The server build was deliberately frozen for the entire study period, and the same researcher moderated every session, eliminating apparatus and operator variation as confounds.

4.2.4 Study Design and Task Set

The study was conducted on the isolated test domain described in Section 3.4 (Figure 2).

A mixed design was used: expertise (expert vs. non-expert) as a between-subjects factor and interface condition (voice vs. GUI) as a within-subjects factor. Each participant completed all four routing tasks under both conditions, with condition order counterbalanced across participants to control for learning effects.

Component	Mean (ms)	Median (ms)	SD (ms)
DDM GraphQL query	6	6	1
DDM routing mutation	32	33	1
Claude Haiku (intent → tool call)	1587	1430	451
End-to-end text pipeline	**1572**	**1467**	**368**
Whisper base (real speech, 6 s)	305	303	14
End-to-end voice pipeline (est.)	**~1770**	**~1770**	—

Table 1: Measured latency per pipeline component (n=10, host institution network, 02.06.2026). End-to-end voice pipeline is estimated as Whisper median + Claude median + DDM mutation median (303 + 1430 + 32 ms).

Four tasks of increasing complexity were defined:

Task 1 (Simple): establish a single route between a specified transmitter and receiver channel (both device names provided). **Task 2 (Medium):** establish two further concurrent routes for a predefined recording scenario (device names provided; channel mapping determined by the participant). **Task 3 (Complex):** inspect the current routing state, remove all existing subscriptions, and apply a new multi-route configuration (recording → live-broadcast scenario) from a written description. **Task 4 (Live error correction):** in a simulated live scenario, correct a routing sent to the wrong destination in real time—removing and re-establishing routes to another destination—while all other active routes must remain uninterrupted, testing real-time recovery, the most safety-critical live-production operation.

The GUI baseline condition used Dante Controller, the standard Dante routing application, on a MacBook with a 68.6 cm (27-inch) 4K display and a mouse. The voice condition used the voice-control interface with push-to-talk activation.

4.2.5 Procedure

Each session lasted approximately 40–80 minutes and was approved by the host institution's ethics committee. After informed consent, participants received a brief interface introduction (non-experts received no prior Dante training) and completed a practice trial before measured tasks began. After each task, participants completed a brief RTLX assessment (∼3 min); after all four tasks in one condition, they completed the SUS and brief questions on impression, trust, and preference, then switched conditions following a five-minute break, followed by a short semi-structured debrief.

Task completion (binary), time-on-task, and error count were recorded by a silent observer. Session audio was retained only in pseudonymized form for offline transcript analysis, stored securely per data-protection requirements; no personally identifying audio was published or shared. In the voice condition, Whisper STT transcription errors were logged separately from user-initiated errors (any deviation between recognized and spoken device/channel names), since domain-specific STT errors constitute a distinct, separately addressable failure mode [24]—conflating them with interface-design failures would confound our metrics with speech-recognition accuracy rather than isolating voice interface usability itself.

4.2.6 Measurement Instruments

To limit fatigue from the repeated-measures design (8 task executions per participant), cognitive load was assessed per-task via the Raw Task Load Index (RTLX, unweighted average of the six NASA-TLX dimensions), which correlates highly ($r > 0.90$) with the weighted NASA-TLX while reducing administration time [31], [32]. Performance (rated 0 = very poor to 100 = very good) was inverted per NASA-TLX convention. A weighted NASA-TLX [14] was planned for validation but not administered; H2 was tested on the RTLX. Usability was assessed via the ten-item System Usability Scale (SUS) [15] (0–100).

4.2.7 Statistical Analysis

Because interface condition (voice vs. GUI) was a within-subjects factor, the two modalities were compared with the Wilcoxon signed-rank test (paired), whereas the between-subjects comparison (expert vs. non-expert) used the Mann-Whitney U test; both

are appropriate for ordinal data and small samples. Exact tests were used throughout. Statistical significance was defined at $p < 0.05$. Effect sizes are reported as r ($|Z|/\sqrt{n}$ for Wilcoxon; rank-biserial correlation for Mann-Whitney U).

5 Results

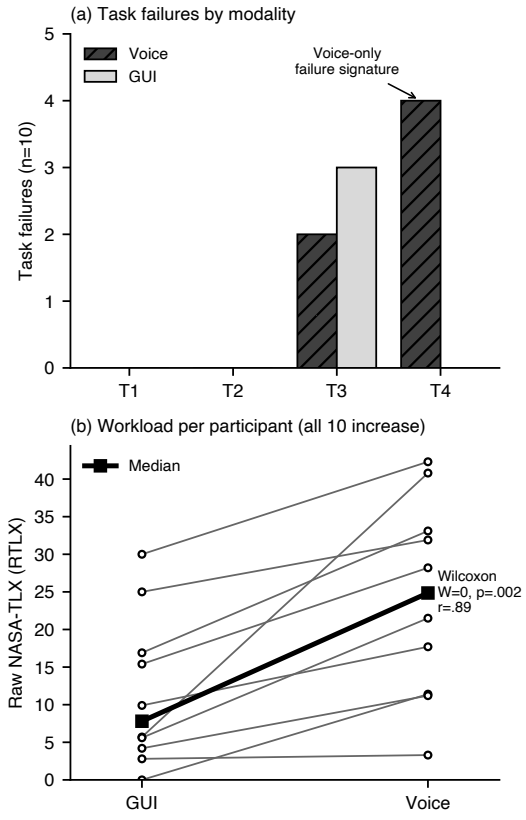


Fig. 3: Study results (n=10): (a) task failures by modality (Task 4 is voice-specific); (b) per-participant RTLX — all ten increase under voice (Wilcoxon W=0).

Ten participants completed the study (five Dante experts, five non-experts), with mode order fully counterbalanced (five GUI-first, five Voice-first). Each participant performed four routing tasks of increasing complexity in both the voice and the GUI condition, yielding 80 task instances. Workload is reported as RTLX and usability as SUS; tests and effect sizes follow Section 4.2.7.

Task	Voice failures	GUI failures
Task 1 (single route)	0	0
Task 2 (two routes)	0	0
Task 3 (delete + reconfigure)	2	3
Task 4 (live correction)	**4**	**0**

Table 2: Task failures by task and modality (n=10).

5.1 Task success

Across all 80 task instances, 71 (88.8%) were completed successfully; voice (85%; non-experts 90%, experts 80%) trailed the GUI (92.5%), with a clear diagnostic pattern by task (Table 2). Tasks 1–2 produced no failures in either modality; Task 3 (complex reconfiguration) was the overall error magnet at the level of individual error instances (37 of 51 recorded errors, 73%) but nearly balanced between modalities and mostly recoverable. The decisive, modality-specific *failure* signature is Task 4 (real-time error correction), where voice failed for four of ten participants versus zero under the GUI—correcting a routing error without disrupting concurrent routes proved uniquely difficult by voice. These patterns are not directly comparable: Task 3's error count aggregates recoverable sub-action errors across a multi-step sequence, whereas each Task 4 attempt was a single binary trial per participant and modality—the four voice failures are distinct unrecovered outcomes, not instances within a larger count.

5.2 Cognitive load (RTLX)

Contrary to H2, voice imposed significantly *higher*, unanimous workload than the GUI (mean RTLX 24.1 vs. 11.6, +12.5 points; Wilcoxon W=0, p=.002, r=.89); per-task RTLX contradicts H2's specific prediction of a Task 2–3 concentration—the increase was in fact *smallest* at Task 2 (+6.9 points) and largest at Task 4 (+23.8 points). Sub-scale increases were largest for Effort and Mental demand, consistent with participants' reports of waiting for system feedback (Table 3); self-rated Performance was essentially unchanged, indicating participants did not perceive their own success as modality-dependent even where objective outcomes differed.

5.3 Usability (SUS)

The GUI achieved a mean SUS of 80.2 (above the ~68 industry average) versus 66.0 for voice—a significant within-subject difference (Wilcoxon W=1, p=.008,

Dimension	GUI	Voice	Δ %
Mental	16.4	35.1	+114 %
Physical	4.7	12.0	+155 %
Temporal	18.2	32.7	+80 %
Effort	13.3	30.4	+129 %
Frustration	11.5	24.3	+111 %
Performance	94.7	89.6	−5 %

Table 3: NASA-TLX sub-scale means, GUI vs. voice (n=10); Performance as rated (100 = very good).

r=.85), with eight of nine non-tied pairs favoring the GUI. Thus, for both principal outcome measures—workload and usability—the GUI was preferred with a large effect size despite the small sample.

5.4 Effect of expertise

Descriptively, the voice interface did not equalize expertise as H3 predicted: non-experts reported both lower absolute voice workload (mean RTLX 19.2 vs. 29.1 for experts) and markedly higher voice usability (mean SUS 75.5 vs. 56.5), with a smaller voice-to-GUI usability gap (−8.0 vs. −20.5 SUS points)—effect sizes r = .52–.60 (usability) and r = .44 (workload). However, with five participants per group none of the between-group comparisons reached significance (all $p > .15$, the exact Mann–Whitney U's attainable minimum at this size being $p \approx .008$), so these should be read as **effect-size-supported trends, not confirmed differences**. A control analysis found no significant order effect (GUI-first vs. voice-first, all $p > .3$), confirming counterbalancing was effective and the voice-over-GUI result is not an artefact of presentation order.

5.5 Error analysis

Log-verified analysis of every voice command (transcription plus the resulting tool call) distinguished three error classes. **(i) Speech-recognition errors** (Whisper) — channel or device-name confusions and, notably, decay of the delete verb ("Entferne" mis-transcribed as "In serne" / "Wenn fernet") — occurred mainly in Task 3 but were largely recoverable, because connect commands tolerated substantial phonetic variance and unclear input triggered a fail-safe *show-routing* response rather than a destructive action. **(ii) Delete-command disambiguation errors** (system design) were the primary cause of the Task-4 failures: a delete instruction naming a device with a channel range but no explicit source/destination direction was resolved to the wrong side of the routing matrix, severing routes that were required to remain active; in one case an unspecific confirmation prompt deleted six routes instead of two. **(iii) User errors** (misreading the task, confusing studio A/B) accounted for the remainder. Critically, the connect operation produced no task failures in any participant, and the task-critical voice failures were driven predominantly by delete-command semantics rather than by raw transcription accuracy. A notable dissociation emerged between objective outcome and trust: two non-experts failed the identical Task-4 voice scenario, yet one, attributing the failure to the system, reported collapsed trust (SUS-voice 60, trust 2/5), while the other, attributing it to her own naming error, retained high trust (SUS-voice 87.5, trust 4/5).

6 Discussion and Conclusion

6.1 Principal Findings

The user study (n=10) yielded a clear and, in one respect, counter-intuitive picture. **H1 was confirmed** (non-experts: 90% voice success, exceeding the 80% threshold). **H2 was disconfirmed—in the opposite direction of the prediction:** rather than reducing cognitive load as the interface-overhead literature suggested, voice imposed significantly *higher*, unanimous workload and lower usability (Section 5.2–5.3). We attribute this reversal to an overhead intrinsic to the voice modality that outweighs the navigation cost it removes: the effort of formulating an exact spoken command, waiting for asynchronous feedback, and recovering from errors—reflected in the Mental and Effort sub-scale increases (Table 3). For a supervisory administrative task with an already efficient spatial GUI, speech adds rather than removes load.

H3 was not statistically supported but is descriptively informative: the voice interface did not equalize expertise (Section 5.4), with five participants per group insufficient for significance—these should be read as effect-size-supported trends, not confirmed group effects. The most consequential finding is the **modality-specific failure signature in Task 4** (real-time error correction, Table 2): log analysis attributes these failures primarily to delete-command disambiguation (a system-design issue) rather than speech-recognition accuracy—a concrete, addressable design target rather than a fundamental limit of voice control.

The complete implementation demonstrates that voice-controlled Dante audio routing is technically viable end-to-end within acceptable latency (<2 s), validating the design recommendation [22] that a bounded MCP tool set significantly improves LLM accuracy.

6.2 Limitations

Several constraints on this work merit explicit discussion.

Scope and proprietary API: The system is specific to Dante/DDM, the only official programmatic Dante routing interface; extending it to other AoIP standards (AES67, AES70/OCA, RAVENNA) via the tool design methodology of [23] remains future work.

Network topology complexity: The LLM resolves ambiguous device references against a known device list in the system prompt; this may degrade in networks with hundreds of devices or inconsistent naming conventions. Confidence scoring or multi-turn clarification dialogue (not yet implemented) could mitigate this.

Latency budget: End-to-end latency ($\sim$1770 ms) is acceptable for non-real-time studio/broadcast preparation but not for extremely time-critical live scenarios (e.g., on-air mixing), for which lower-latency local-LLM variants are planned as future work. We expect halving would ease the workload gap more than the correctness gap: the Effort/Mental/Temporal increases are tied to waiting for feedback (Section 6.1), consistent with Dewey's non-real-time framing [25]. Task 4 failures, however, stem from delete-command disambiguation (Section 5.5), not response time, and would likely persist even near AR/haptic latencies (<250 ms [11]), since the failure is semantic, not perceptual.

User study sample size: The user study (N=10) is modest but comparable to methodologically related work [3]. Statistical power for detecting differences between expert and non-expert groups is limited; results should be interpreted as pilot evidence rather than definitive claims.

Data privacy and cloud dependency: While Whisper speech recognition runs locally, intent resolution relies on the Claude API, so transcribed text leaves the local network—a significant constraint for high-security environments (e.g., classified audio content, strict data residency requirements). A local LLM variant (Section 6.3) would remove this dependency entirely.

Regulatory compliance: Under the EU AI Act amendments approved in June 2026 [33], AI systems used in critical infrastructure or processing biometric data such as voice audio are classified high-risk, with full enforcement from December 2027. This study's design already anticipates the resulting obligations: pseudonymized, metadata-separated storage; timestamped audit logging of every command and routing decision; and a decision-support (not autonomous) architecture in which humans retain final control. Full formal certification was outside scope, but compliance appears to require primarily procedural documentation rather than architectural change.

6.3 Future Work

Three directions warrant investigation beyond the AoIP-standards extension noted above. A **verification agent** in an editor/critic architecture [16] could confirm routing decisions before execution, reducing accidental misconfiguration. **Multimodal context integration**, following approaches that use external context to disambiguate speech [28], could integrate the DDM Domain Inventory as a dynamically updated entity reference to auto-correct device names extracted from ASR errors. Finally, a **local LLM variant** (quantized Ollama/Llama 3) would eliminate external API dependency, reduce latency to <500 ms, and preserve audio confidentiality; preliminary testing suggests this is feasible but requires more local compute than currently available.

6.4 Conclusion

This work presents a peer-reviewed system for LLM-based voice control of professional Dante audio network routing. Whisper speech recognition, Claude language understanding, and MCP-mediated DDM API access demonstrate that natural language interfaces can make complex infrastructure management accessible to non-expert operators without sacrificing safety or reliability. The Dante MCP Server, including complete implementation and system prompts, will be released as open source upon paper acceptance.

Acknowledgment of AI Assistance

Claude (Anthropic) assisted with editorial drafting of portions of this manuscript's text (June–July 2026); all content was reviewed and verified by the authors. No confidential or personal study data was input into any AI system.

References

[1] Audinate Group Limited, "Audinate Annual Report 2025," Annual Report FY2025, 2025.

[2] Bergner, S., Crider, M., Kirkpatrick, A. E., and Möller, T., "Mixing Board Versus Mouse Interaction in Value Adjustment Tasks," 2011.

[3] Woodard, B., Geleta, M., Jr., J. J. L., Fanelli, A., and Wilson, R., "AudioMiXR: Spatial Audio Object Manipulation with 6DoF for Sound Design in Augmented Reality," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2025, doi:10.1145/3749478.

[4] Harihar, D., Shrivastava, V., Talele, P., and Jahagirdar, A., "Voice-Based User Interface for Hands-Free Data Entry and Automation at Workplaces," *MethodsX*, 15, 2025, pMC12423416.

[5] Liang, J., Zhang, H., Liu, H., Cao, Y., Kong, Q., Liu, X., Wang, W., Plumbley, M. D., Phan, H., and Benetos, E., "WavCraft: Audio Editing and Generation with Large Language Models," in *LLM Agents Workshop @ ICLR 2024*, 2024.

[6] Doh, S., Koo, J., Martínez-Ramírez, M. A., Liao, W.-H., Nam, J., and Mitsufuji, Y., "Can Large Language Models Predict Audio Effects Parameters from Natural Language?" in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA 2025)*, 2025.

[7] Wang, C., Scazzariello, M., Farshin, A., Ferlin, S., Kostić, D., and Chiesa, M., "NetConfEval: Can LLMs Facilitate Network Configuration?" *ACM on Networking*, 2, 2024, doi:10.1145/3656296.

[8] Lira, O. G., Caicedo, O. M., and da Fonseca, N. L. S., "Large Language Models for Zero Touch Network Configuration Management," 2024.

[9] Anthropic and Model Context Protocol Contributors, "Model Context Protocol Specification," https://modelcontextprotocol.io/specification, 2024.

[10] Fan, Y.-Y., "MCP2OSC: Parametric Control by Natural Language," 2025, preprint, not peer-reviewed.

[11] Balamurugan, R., Hwsein, R. R., Singh, P., Murthy, S., Devaraj, J., Mamajonov, D. A., and Pirnazarova, M., "Real-Time Voice-Controlled Interface for AR/VR Surgical Simulations," in *2025 International Conference on Innovations and Emerging Technologies in AI & Communication Systems (IETACS)*, pp. 1535–1541, IEEE, 2025, doi:10.1109/IETACS68750.2025.11385587.

[12] DVIGear, "DVIGear Introduces DisplayNet Connect for AI Agents," Press Release, not peer-reviewed, 2026.

[13] Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I., "Robust Speech Recognition via Large-Scale Weak Supervision," in *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*, volume 202, 2023.

[14] Hart, S. G. and Staveland, L. E., "Development of NASA-TLX: Results of empirical and theoretical research," in P. A. Hancock and N. Meshkati, editors, *Human Mental Workload*, North-Holland, 1988, doi:10.1016/S0166-4115(08)62386-9.

[15] Brooke, J., "SUS: A quick and dirty usability scale," in P. W. Jordan, B. Thomas, B. A. Weerdmeester, and I. L. McClelland, editors, *Usability Evaluation in Industry*, Taylor & Francis, 1996.

[16] Sandoval-Castañeda, M., Russell, B., Sivic, J., Shakhnarovich, G., and Heilbron, F. C., "EditDuet: A Multi-Agent System for Video Non-Linear Editing," in *ACM SIGGRAPH 2025 Conference Papers*, 2025, doi:10.1145/3721238.3730761.

[17] Qiao, Y., Tran, P.-N., Yoon, J. S., Nguyen, L. X., Huh, E.-N., Niyato, D., and Hong, C. S., "A Comprehensive Survey on LLM-Based Network Management and Operations," *International Journal of Network Management*, 2025, doi:10.1002/nem.70029.

[18] Abane, A. et al., "An LLM-based Agentic Framework for Accessible Network Control," 2025.

[19] Hofmann, B., Piechulek, N., Kreitlein, S., Franke, J., and Bruendl, P., "Beyond Touch-Based HMI: Control Machines via Natural Language Using LLMs and OPC UA," 2025.

[20] da Silva, L. M. V., Köcher, A., and Gehlhoff, F., "Beyond Formal Semantics for Capabilities and Skills: Model Context Protocol in Manufacturing," 2025.

[21] Singh, A., Ehtesham, A., Kumar, S., and Khoei, T. T., "A Survey of the Model Context Protocol (MCP): Standardizing Context to Enhance Large Language Models (LLMs)," Preprints.org, 2025, doi:10.20944/preprints202504.0245.v1.

[22] Song, W., Zhong, H., Ding, Z., Xue, J., and Li, Y., "Help or Hurdle? Rethinking Model Context Protocol-Augmented Large Language Models," 2025.

[23] Mastouri, M., Ksontini, E., Barrak, A., and Kessentini, W., "From REST to MCP: An Empirical Study of API Wrapping and Automated Server Generation for LLM Agents," 2025.

[24] Tang, Z., Wang, D., Huang, S., and Shang, S., "Full-text Error Correction for Chinese Speech Recognition with Large Language Model," in *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2025, doi:10.1109/ICASSP49660.2025.10890161.

[25] Dewey, C. and Wakefield, J., "Elicitation and Quantitative Analysis of User Requirements for Audio Mixing Interface," in *144th Convention of the Audio Engineering Society*, 2018, aES E-Library ID 19452.

[26] Vanka, S. S., Safi, M., Rolland, J.-B., and Fazekas, G., "Adoption of AI Technology in Music Mixing Workflow: An Investigation," in *154th Convention of the Audio Engineering Society*, 2023, paper #10653.

[27] OCA Alliance, "AES70 / Open Control Architecture," AES Standards AES70-1-2024 and AES70-3-2024, 2024, https://ocaalliance.com/what-is-aes70/.

[28] Piao, M., Li, Y., Qiao, X., Zhao, X., Zhao, H., Zhang, M., and Yang, H., "Slide-EC: A Slide-Based Error Correction Framework for Speech Recognition," in *2025 10th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, pp. 254–258, IEEE, 2025, doi:10.1109/ICCCBDA64898.2025.11030430.

[29] Audinate Pty Ltd, *Dante Domain Manager — Developer Guide and API Reference*, 2025, accessed 2026-06-02.

[30] Wanderley, M. M. and Orio, N., "Evaluation of Input Devices for Musical Expression: Borrowing Tools from HCI," *Computer Music Journal*, 26(3), pp. 62–76, 2002, doi:10.1162/014892602320582981.

[31] Hart, S. G., "NASA-Task Load Index (NASA-TLX); 20 Years Later," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 50, pp. 904–908, SAGE Publications, 2006, doi:10.1177/154193120605000909.

[32] Byers, J. C., Bittner, A. C., and Hill, S. G., "Traditional and Raw Task Load Index (TLX) Correlations: Are Paired Comparisons Necessary?" in A. Mital, editor, *Advances in Industrial Ergonomics and Safety I*, pp. 481–485, Taylor & Francis, 1989.

[33] Morgan Lewis, "EU Approves Delays and Other Amendments to Certain EU AI Act Obligations: What Businesses Should Know," Legal Practice Note, 2026.