Exploiting Latency In The Design Of A Networked Music Performance System For Percussive Collective Improvisation

Ari Liloia
Carnegie Mellon University
Pittsburgh, PA, US
aliloia@alumni.cmu.edu

Roger Dannenberg
Carnegie Mellon University
Pittsburgh, PA, US
rbd@cs.cmu.edu

Abstract

We present the design, prototype implementation, and informal testing of a distributed web-based networked music performance (NMP) system for collaborative improvisation and experimentation. Influenced by composition and interaction design techniques from a wide range of work on collaborative virtual music environments, rather than treating latency as inherently disruptive to the musical and social engagement that characterizes traditional performance, we incorporate and exploit network delay to facilitate and visualize them, providing a novel approach to creating "jam session"-like experiences without a separate audience. During sessions, users collaboratively perform semi-improvised music in quasi-real time. The production and interpretation of individual musical gestures ("drum hits") are visualized in a continuously devised feedback network. The music produced can be treated as a starting point for compositions developed asynchronously, or as complete pieces of music produced live.

CCS Concepts

• Networks → Network experimentation; **• Computer systems organization → Reliability**; **• Human-centered computing → Collaborative interaction**; **Web-based interaction**; *Visualization theory, concepts and paradigms*; *Graph drawings*; **Synchronous editors**.

Keywords

Networked Music Performance, Web-Based Music Systems, Network Latency, Collective Improvisation, Percussion

1 Introduction

In this paper, we present the design, prototype implementation, and informal testing of a distributed web-based networked music performance (NMP) system for collaborative semi-structured group improvisation. The system implements a novel musical and visual strategy that incorporates and disguises end-to-end communication latencies on the order of seconds between distributed users. Said strategy also accommodates co-located users; if multiple instances of the client application are open at the same time in the same space, their overlapping output maintains musical (rhythmic) coherence. When a performance is complete, the system distributes records of users' musical activity. These can be used as starting points for compositions developed asynchronously. The system then offers a specialized collaborative environment where users can create distinctive music together during quasi-real time performances; the results of said performances can be seamlessly integrated into their existing compositional workflows.

We first explore why established conventions for *traditional* (co-located, with familiar instruments) collaborative improvisational performance make it feel *live* to actively participating performers and passive audience members. We review contemporary literature on liveness in performance and suggest that this is due to performers' clear and continuous demonstration of agency over musical output, which is possible due to the ample, legible and continuous multi-sensory feedback found in traditional settings. We discuss how and why novel distributed networked music performance (NMP) systems subject to significant latency favor a kind of performance that demands less musical awareness, thus reflecting less performer agency in musical output.

We explore successful strategies from existing distributed NMP tools for facilitating quasi-real time interaction and "performer-centered" performance in the face of significant communication latency. We review techniques for promoting awareness in several novel performance contexts, adapt them to a distributed setting if necessary. We incorporate these techniques into a musical and visual paradigm for a prototype system, which includes the responsibilities of different participating users, the UI components that facilitate their interaction, and the musical delays that hide transmission latency. We then describe a prototype implementation of the system and its components. We describe the details of several informal tests of the system, in which testers provided their informed consent, and present a piece of music composed asynchronously post-performance from records of these test sessions.

We hope that this work inspires more musicians and technologists to explore latency, particularly at longer timescales, as a compositional tool, rather than treating it as a problem to eliminate. We also hope that the work inspires other NMP developers to explore real-time interaction as an ideation exercise for asynchronous composition.

The most recent iteration of the prototype, as well as some supplementary materials referenced throughout the paper, are available at `https://github.com/arililoia-cmu/elbs_nime`.

2 Background And Related Work

2.1 Conventions For Traditional Performance

Collective improvisational performance or *jamming* involves quasi-real time interaction and *in-time* [26] music creation. Performers simultaneously create and perceive a complete piece of music. Musical goals usually involve coherence in combined ensemble output [30] and complexity along musical dimensions of salience, such as rhythm or texture [39].

Traditionally, performers and audiences are co-located [47] to the extent that auditory and visual latency are not disruptive to musical coordination [2]. Performers utilize "traditional" instruments, whose musical patterns and degree of skill required for

NIME '25, June 24–27, 2025, Canberra, Australia

operation have, over time, become common knowledge [18, 32]. The traditional setting is characterized by *visibility*, in that both performers and passive audiences can perceive the activities and skills carried out to fulfill musical goals [31]. Performers' *awareness* of musical responsibilities, relationships, and musical activity [3] is afforded by the highly visible setting. As they perform, they demonstrably assert musical *agency* [46] and witness others doing the same. [32] and [30] provide a comprehensive overview of these ideas from an embodied cognition perspective.

Performance is characterized by *liveness*, which is associated with unique feelings that non-performed music cannot replicate. Contemporary literature considers different perspectives for passive audiences and active performers: Performers may associate liveness with witnessing a "dynamically performed assertion of human presence" [43] in ensemble musical output. This is reinforced by the *risk* present in traditional settings where meeting musical goals is not guaranteed [14] and can be attributed to user agency when present. Alternatively, from the perspective of a passive audience, a musical event can feel "live" (and in turn, be perceived as a performance) if it effectively simulates the spontaneity and interactivity usually associated with achieving musical goals, regardless of human agency reflected in musical output or risk involved. [1, 9, 43] provide comprehensive overviews of these ideas.

2.2 Networked Music Performance

The Hub [6] are credited with introducing *distributed* [7] or *non-realistic* [20] networked music performance, in which musicians connected over computer networks with characteristics that disrupt conventional methods for collaboration and coordination, such as significant latency and physical distance between users at network endpoints, interact in quasi-real time while producing music in-time. They are also credited as progenitors of the encompassing field now called *networked music performance* (NMP), which has endured as a flexible category of musical tools. [20, 25, 42, 49] provide comprehensive overviews of the entire NMP space and describe how the term has developed.

We identify two main considerations in distributed NMP: how musical timing is affected by latency, and how the lack of visibility in a networked setting affects awareness, especially that of performer agency over musical output. The following subsections review established strategies addressing these issues in novel performance contexts.

2.2.1 Effect Of Latency On Musical Timing. Both latency and jitter (variation in latency) increase significantly and reliably with distance between network endpoints [37]. While musicians connected over a two-way audio stream can play with up to 75 milliseconds of latency without needing to adjust their auditory feedback, latency in networked systems can be much higher. This has musical consequences; certain kinds of rhythmic synchronization become more difficult with jitter [25], and longer delays make interaction feel less immediate [17].

Some NMP systems address this through musical styles less dependent on precise timing, such as texture-based atmospheric music [5]. A strategy that works with *cyclical* music (based around musical cycles) is the "fake time" approach (FTA) [1], which extends latency to avoid jitter. The technique requires imposing a set tempo, synchronizing local clocks across clients, estimating the worst-case end-to-end latency (henceforth called *WCL* and measured in milliseconds), and implementing a scheduling system with which future execution times for transmitted data can be scheduled. Extra latency is added to this execution time, such that the "precedent time" [35] between transmission and message execution upon reception is greater than or equal to *WCL* and is a multiple of the duration of one musical cycle. Control data is then afforded the length of one musical cycle for end-to-end transmission. If players maintain their awareness of timing within and between individual parts, combined output reconstructed at network endpoints maintains rhythmic coherence. NINJAM [12] and GDC [15] implement this technique for transmitting and reconstructing audio and control data, respectively.

2.2.2 Effect Of Visibility On Awareness And Agency. Visibility is less present in distributed, non-realistic NMP systems that facilitate user interaction through digital interfaces (UIs) and novel digital instruments (DMIs). There is precedent in NMP for reducing the importance of performer awareness, and in turn, that of performer agency, in producing music: Users may play over backing tracks [40] or rearrange segments of pre-composed music [4]. This may be done to pursue social or exploratory goals; while early net music surveys [19, 49] separated these from performance tools, contemporary overviews [20, 25] characterize them as NMP. Music produced by these systems may contain coherence and complexity not attributable to user agency. While an audience-centered perspective of performance might welcome this approach, we take a performer-centered approach and explore strategies that manifest musical agency and facilitate awareness in a setting subject to latency and inherently lacking visibility.

Communication latency and novel DMIs lead to ambiguity in other users' responsibilities and capabilities, hindering the development of influence patterns. To combat this, some tools enforce "follower-leader" relationships between groups of users [16, 40].

While visual feedback in traditional performance usually comes from peripheral vision [45], direct visual feedback promotes awareness in novel settings. [31] suggests introducing task-specific visual artifacts in a shared space to communicate roles and activities, especially "metaphor-based mappings" referencing common associations between action and movement. Visualizations of musical gestures can be helpful: [48] introduces the "anticipatory score" pattern for displaying upcoming sounds in a collectively created scrolling score. [33] notes the value of visualizing sources of sounds already created, and how they are "modified" as they are perceived and reinterpreted by other musicians.

Excessive activity can crowd the virtual space [4] and obscure which collaborative activities have the highest priority [8]. Effective approaches include using proximity in a virtual space to communicate interdependence [44] to focusing attention on specific regions of said shared space, as well as simply limiting the number of participants.

However, in displaced NMP, latency prevents immediate ensemble-wide updates to shared virtual spaces. *Client-side prediction*, a pattern from video games, can be used to simulate immediate feedback in a shared space to gestures: If visualizations of users' own musical gestures are predictable for some time after they are produced, they can be displayed locally before their successful broadcast is confirmed, then reconciled if necessary afterwards [21].

[1] [20] survey categorized various NMP tools under this term, with [22] being a significant early example

2.3 Novel Directions

NMP tools are compelling for their ability to "sonify", or manifest musically, network medium characteristics like latency. Most systems exploring this incorporate latency on the order of milliseconds into similar musical timescales, such as the pitch of a sound [11], the parameters of spatial diffusion algorithms [10], and an "echo" effect of roughly 100 milliseconds [41]. Little existing work scales up latency to different musically relevant timescales.

[35] extend the FTA with the "mutual anticipated session" (MAS), which varies precedent times between pairs of musicians, such that a unique, time-shifted and rhythmically coherent "perspective" of the ensemble's complete musical output is constructed at each client, resulting in "bi-located" [41] rhythmic patterns. It is novel to use scaled inter-user latencies as precedent times in a MAS-like system, with care taken not to set them below WCL.

While recent NIMEs address interest in novel DAW-based solo performance [36] and collaborative simultaneous interaction [27], few NMP tools exist that allow detailed asynchronous DAW-based editing. Endlesss [2], which was discontinued in June 2024, permitted downloading individual loops created during performance as audio files. Distributing performance records as MIDI files [3] lends itself to more DAW-based musical practices, especially for users interested in timbres not presented by the system. While the phrase "jam session" is commonly used in NIME proceedings to describe any collective improvisation, the phrase's roots in jazz [38] and the common "jam band" format [23] highlight the potential of collective improvisations as ideation exercises. Few NMP tools explore this context.

3 Design

In this section, we present a musical and interaction paradigm for a prototype system influenced by techniques presented in the previous section. Physically displaced users interact over a network, subject to significant latency and with personal computers as network endpoints. An advance scheduling system is implemented. A tempo is imposed and used to map time to *beats*, which increase with time according to said tempo.

3.1 Client Classes And UI

The application implements three *classes* for clients, *Composer*, *Performer* and *Listener*. Users interact during *sessions* of collaborative musical activity, which are either *started* or *stopped*.

Collaborative musical activity is mediated by a two-dimensional locally rendered interactive representation of the entire ensemble, referred to henceforth as the *grid*. Clients are represented by simple colored *nodes* on grid vertices and identifiable by client class: Composers are red, Performers are blue, and Listeners are green. Clients' own nodes appear black.

3.2 Drum Hits

The widespread appeal of drums, especially in a collaborative "drum circle" setting, can be attributed to their accessibility for beginners and potential for advanced musicianship, making them an ideal starting point for a DMI. The action of playing a hand drum translates especially well to a computer keyboard-based digital instrument, both produce sound immediately and offer a natural rebound when "struck".

The music produced using the system is percussive and based around musical cycles. The system's primary musical gesture and format for instruction is the *drum hit*, or a Composer or Performer pressing a key on their computer keyboard to "play" a virtual drum. This act triggers the immediate local playback of a drum sample, and a hit marker is immediately displayed over the user's node. Control data related to the pressed key / drum hit is then transmitted to other clients; the effect and intended interpretation varies based on the relationship between the transmitting and receiving client.

An advance scheduling system for musical contributions accurately delays execution by an integer number of beats. Consider a Client A that triggers a drum hit at beat b_S, which is received by Client B at time b_R, before one musical cycle has passed. Client A's drum hit is played back on Client B at beat $b_S + C$, C beats after b_S. C is such that the duration of C beats is greater than or equal to WCL.

3.3 Client Roles

Composers can click and drag their nodes to draw paths along grid lines, or *connections*, between themselves and Performer nodes. When a Composer plays a drum (triggers a drum hit), one corresponding *hit marker* per connection appears over its node. These hit markers then travel along connections towards Performers over some integer number of beats. The time it takes for a hit marker to move from one node to another is the *procession time*; procession times are quantized to beat multiples. Unless otherwise noted, the procession time of a Composer's hit marker in beats is the length of the connection along which it travels in grid lines. Composers can click on connections they have drawn to delete them. Multiple Composers can be connected to the same Performer; if their pacing and timing are consistent, their combined output at the Performer is rhythmically coherent.

`extras/gifs/composer.gif` on the project GitHub page shows a Composer drawing a connection between itself and a Performer, playing several drum hits, then deleting the connection. Note that the hit markers start out moving slower, then speed up. This is explained in Section 4.6.

Performers follow the motion of composer hit markers like a scrolling score, playing drums as they arrive at their nodes. Given that nodes are quantized to grid vertices and procession times are quantized to beat multiples, if Composers maintain pacing and timing between parts, Performers "perceive" rhythmically coherent versions of combined output. When a Performer plays a drum, a 45-degree rotated square "wavefront" ripples outwards from its avatar. This shape is chosen because it allows a point on the wavefront to move either one grid line (the point on the vertex of the square) or across the diagonal of one grid square (the equivalent of two grid lines) in one beat. Unless otherwise noted, the procession time of a Performer's hit marker in beats is the distance between itself and any Listener in grid lines. Performer hit markers eventually "collide" with all listener nodes.

`extras/gifs/performer.gif` on the project GitHub page shows a Performer responding to drum hits from a Composer, attempting to time the arrival of the hit markers with its own drum hits. Note that the hit markers start out moving slower, then speed up. This is explained in Section 4.6.

Listeners perceive the combined output of all performers; each Performer drum hit arriving at their node plays back a drum sound in their client. Unlike Composer drum hits, Performer drum hits are not "directed": all Listeners hear drum hits from

[2] https://endlesss.fm/
[3] https://www.midi.org/specifications

all Performers. Listeners click and drag their own nodes to experience different timing combinations of combined Performer outputs, which, if Performers maintain pacing and timing between parts, are rhythmically coherent at the Listener. Listener perspectives are "recorded" and distributed after sessions.

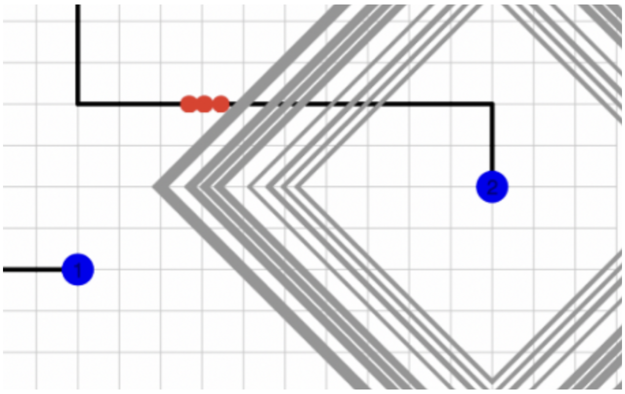

Figure 1: A section of the grid during play showing two Performers. Performer (2) is playing much more than the Performer (1), as evidenced by the difference in number of Performer hit markers surrounding it.

The urgency of a hit marker is proportional to its proximity to a node on the grid. Users are encouraged to focus on grid areas closer to their own node; other grid activity takes place in peripheral vision. Users contribute to coherence in ensemble output by monitoring other users' playing styles via the grid, including those to which they are not connected. For example, 1 shows a setup with two Performers; one labeled (1) on the left side of the figure and one labeled (2) on the right. As shown by the number and density of square hit markers propagating outward from Performer (2) relative to Performer (1), Performer (2) is playing much more, and may be behaving disruptively. No Composers are shown in the figure, but both Performers have incoming connections. To discourage this behavior, a Composer connected to Performer (2) could break their connection. Alternatively, Performer (1) could accommodate this behavior by adjusting their playing to match or counter Performer (2)'s style.

In this manner, users continuously observe musical agency propagating through a feedback network of other users, which is continuously devised as Listeners move around and as Composers edit connections and tempo. Musical agency is visualized to promote awareness and make the experience feel "live".

4 Implementation

In this section, we describe the prototype implementation of the system.

4.1 Network Topology and Communication

The Global Drum Circle (GDC), a distributed web-based system to enable drum circle performances across the Internet developed by the same authors, was used in software that formed the basis for this new system. Subsections 4.1 and 4.5 refer to implementation details slightly or completely unchanged from the 2022 version of this software.

The prototype back end is implemented using Java [4] and Maven [5], and builds off the Webbit [6] base code for base code for a WebSocket and HTTP server. The front end is implemented using p5js [7]. Client-side audio playback is implemented using the WebAudio API [8]. Communication takes place over WebSockets [9] using O2lite [10]. The prototype implements a "hub-and-spoke" network topology, in which information from end users is routed through and stored on a central server; it is then straightforward to log performance data as it is routed through the system, maintain a master clock, and reject or reconcile invalid actions from clients.

Before sessions, one user downloads the application, compiles it on a virtual machine with a public IP address, and independently distributes a link to the browser-based client application.

4.2 Upper Bound For End-To-End Communication

It is necessary to pick a value for the WCL, or the upper bound for end-to-end communication. In the interest of allowing for experimentation with different network conditions, we allow this value to be easily set during setup. However, we implement a default value of 3500 ms. At time of writing, Google Cloud's virtual machine performance dashboard gives the highest median round-trip transmission time between a cloud region and external endpoint (i.e. from external endpoint to cloud region to external endpoint) as 350 milliseconds [11]; our default value is then order of magnitude above this for safety. During informal tests with distributed users, no issues suggesting a higher WCL was necessary presented themselves.

4.3 Role Choice And Setup

Upon connecting to the server, the client prompts users to enter authentication credentials and choose a client class. A *chat window* visible during setup explains metrics for success related to the role chosen. During this process, the client's local clock and timemap synchronize with the server, and client-server round-trip time (RTT) is crudely estimated using N O2lite messages initially spaced by S milliseconds and increasing by a factor of F for each subsequent measurement. The implementation allows these values to be chosen during setup, but default values $N = 10$, $S = 20$ and $F = 2$ are set, as this takes roughly 20 seconds. Testing implied that the average setup took slightly more time than this.

The client then sends this value to the server, which associates it with the client instance. Client instances that complete this process are *validated*. As shown in Figure 2, a live count of validated clients by class is displayed; once one client of each class has been validated, they can vote to confirm themselves as session participants. Voting restarts if a new client is validated or a validated client drops out.

[4] https://www.java.com/en/
[5] https://maven.apache.org/
[6] https://github.com/webbit/webbit
[7] https://p5js.org/
[8] https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API
[9] https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API
[10] https://www.cs.cmu.edu/~rbd/blog/nime-blog22may2022.html
[11] https://cloud.google.com/network-intelligence-center/docs/performance-dashboard/how-to/view-google-cloud-latency

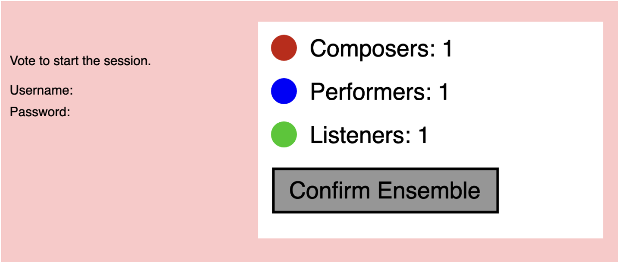

Figure 2: The composer client's UI after one client of each class has been validated.

4.4 Converting RTTs to Initial Distances On Grid

RTTs are roughly estimated to generate initial node placements on the grid. In a process carried out on the server, client instances are modeled as masses connected by springs with rest lengths proportional to the sum of their RTTs. Forces in x and y are calculated iteratively to displace nodes; each node's position is quantized to a G by G grid in random order [13]. After M iterations, nodes achieve relative distances approximately proportional to their scaled transmission times. While G and M can be customized by the user during setup, we set default values of $G = 16$, so that a hit marker traveling across the grid in a straight line will do so in four measures, and $M = 1000$, as all realistic node and RTT configurations tested during development converged within this many iterations. Latency is thus manifested on a novel musical timescale. The choice of G also sets a constraint of $G * G$ on the number of users that can join a session.

4.5 Role-Specific UI Components and Abilities

When the server finishes this calculation, it broadcasts grid positions to users, which then display said grid alongside private and/or role-specific UI elements. All clients implement a tempo indicator, a mixer for controlling different drum volumes (other drums, their own drums, metronome drums), a "started/stopped" indicator, and a series of lights that blink in sync with the metronome. The metronome helps Composers and Performers maintain their pacing and timing; its volume can also be changed. The ensemble begins in the "stopped" state. Any Composer can initiate play.

Client classes' private UI components differ in small ways. The tempo wheel and start/stop indicator in the composer interface are interactive, any composer can interact with them to initiate changes to the state of the ensemble. Figure 3 shows the Listener, Performer and Composer private UI components.

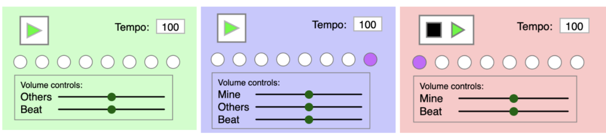

Figure 3: Private UI components from left to right: Listener, Performer, Composer. Each "Volume controls" component has a slider for each source of drum the client class "perceives".

As mentioned earlier, a Composer or Performer pressing a key triggers local playback of one of two conga drum samples, based on whether the key pressed is on the left or right side of the user's keyboard.

The prototype does not implement functionality for changing these drum sounds; all Composers and Performers have access to the same two sounds. Both drums produce identical hit markers on the grid.

4.6 Hidden Delay

Procession times do not accommodate transmission delay, thus, we introduce another "hidden" delay $H \geq WCL$ between one client's action and its display time on other clients. Figure 4 is a timing diagram of the general pattern of end-to-end communication from Composer to Performer / from Performer to Listener.

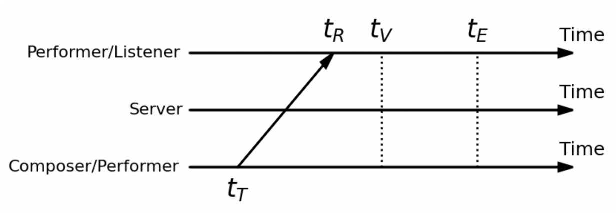

Figure 4: A Composer / Performer plays a drum at time t_T, which is scheduled for display on a Performer / Listener at t_V, such that $t_V - t_T \geq WCL$. It is actually received at t_R, ahead of the worst-case scenario. At t_V it is displayed in the client and begins to move away from the Composer / Performer node, reaching the Performer / Listener node at time t_E, such that $t_E - t_V$ is the procession time.

In addition to accommodating the worst-case transmission latency WCL, the hidden delay ensures that the overlapping output of multiple clients maintains rhythmic coherence and "shifts" users' timelines forward, allowing accurate representation of Performer response times to Composer drum hits to all clients.

4.6.1 Maintaining Rhythmic Coherence In Overlapping Client Output. While the system is geared towards distributed (visually and sonically isolated) users, requiring three remote users is inconvenient for demonstration, and potential users may hesitate to organize a group solely to test it, so we allow users to operate multiple clients on the same machine. It then becomes important to consider rhythmic coherence in overlapping output of multiple overlapping client instances. To accommodate this, we set $H = xB$ where x is an integer and B is the duration of one beat in milliseconds, such that H is "beat-aligned". The implementation sets $x = 4$; Figure 6 reflects this choice. This value can be changed during setup.

The maximum tempo permitted by the system is calculated using $60 * x/(H * 1000)$, given aforementioned default values, this is $60*4/3.5 = (60*4)/3.5 = 68.57 \approx 68$, such that procession times can be one beat long without being smaller than WCL.

4.6.2 Displaying Performer Response Times. To promote awareness, we maintain Performers' responsiveness to hit markers across clients; their skill can be assessed by the degree of simultaneity of the arrival of a Composer hit marker at their node and the appearance of a Performer hit marker. While Performers see Conductor actions delayed to hide latency to the Performer, to create the illusion of instantaneous response from the Performer, Composers must also see Performer actions delayed by another H; the same goes for Performers and Listeners. Given that the movement of hit markers is completely deterministic, we disguise delay by implementing a kind of client-side prediction: Actions requiring responses are delayed by $2H$ and others by H. Users' own drum hits are displayed immediately but move at different rates, such that all drum hits are generally one grid line away from their source at the same time. Thus, it is appropriate to display and play back users' own drum hits immediately in their client application, as confirmation of reception occurs once a hit marker has moved one grid unit from the node position. Immediate responsiveness to user actions is a favorable trait of DMIs [28] and contributes to the feeling of liveness.

Figure 5 shows an example section of the grid - a Composer (1), Performer (2) and Listener (3) are lined up such that the "life cycle" of a drum hit would be represented by a hit marker moving from left to right, from Composer to Performer and from Performer to Listener. Another Composer (4) and Performer (5) are present, representing all possible perspectives.

We show this process as applied to Figure 5, which covers all possible user perspectives, in Figure 6.

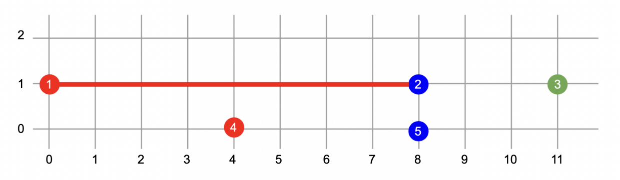

Figure 5: An example section of the grid, containing a Composer (1), Performer (2) and Listener (3) arranged in a horizontal line, as well as a Composer (4) and Performer (5) not in said line.

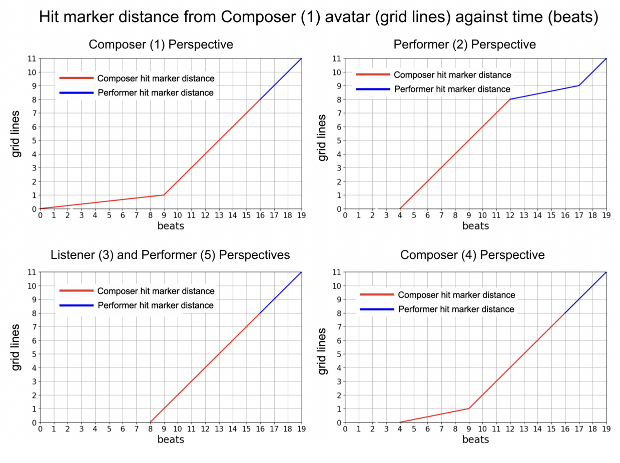

Figure 6: Hit marker distance from composer (1) avatar in grid lines over time. Red lines represent Composer hit markers, blue represent Performer hit markers. The same event is presented in four different ways: From the perspective of Composer (1) (top left), Performer (2) (top right), Composer (4) (bottom right), and Listener (3) and Performer (5), who perceive the same path (bottom left).

`extras/gifs/hidden_delay.gif` on the project GitHub page shows the movement of Composer and Performer hit markers over time as illustrated in Figure 6 for the configuration of nodes in Figure 5.

When play has been stopped or a client suddenly disconnects, an *interruption window* covers the grid to express that normal play is no longer possible.

4.7 Conflict Resolution And Verification

As the server maintains the state of the ensemble, it maintains a record of occupied grid positions and rejects invalid actions. Drum hit messages, changes in tempo, Listener positions, and newly drawn connections are followed by *cooldown periods* exceeding WCL, during which time they cannot perform other actions and the change is afforded the appropriate amount of time for transmission throughout the ensemble. Performer drums are muted to listeners during their cooldown periods.

`extras/gifs/listener_cooldown.gif` on the project GitHub page shows a Listener clicking on its node, moving it to a different location, and experiencing the cooldown time, which is visualized as a small countdown over the node.

4.8 Logging Performance Information / MIDI File Reconstruction

The server continuously logs tempo changes, drum hits, and cooldown periods to reconstruct each Listener's unique perspective. These logs are converted to CSV files and written to one MIDI file per listener; with each Performer's contribution on a separate track. Once MIDI files are generated, the client displays download buttons that send HTTP POST requests for specific files.

5 Testing

5.1 Test Session

An informal testing session was carried out with three distributed participants, all of whom provided their informed consent. Participants first met over a video call and were shown a slideshow introducing the project. A link to the client application was then distributed. Three rounds of play were carried out with the system: The first, a "free for all", gauged the clarity of class-specific instructions provided during setup. No further instructions were given during performance; confusions were addressed and resolved afterwards. In the second round of play, all users switched roles. In the third round, two users managed two clients' responsibilities with two open tabs; one controlled a Performer and a Listener, one controlled two Listeners, and another controlled a Composer. Data was logged in all three rounds and distributed as MIDI files; no issues were encountered when opening them in a range of DAWs. Suggestions from the test session since implemented in the prototype include making clients' nodes more identifiable on the grid and clarifying the instructions given during setup. The MIDI and CSV files are available on the project GitHub page under `extras/logs`.

5.2 Complete Piece of Music

A piece of music composed by the first author based on the third test session MIDI files is available on the project GitHub page under `extras/composition.wav`.

The test session MIDI is voiced using a marimba soundfont from the 1928 Steinway Piano Kontakt library by 8dio [12]; the mallet sounds are percussive and maintain the quality of the prototype's conga sounds, but are pitched differently to make the piece more compelling. To make sure the parts of the piece

[12]https://8dio.com/products/1928-scoring-piano

that originated from the test session were identifiable, no more overtly rhythmic elements were added.

The atmospheric sounds accompanying the marimba were generated using Synplant by Sonic Charge [13], the SoundMagic Spectral plugin suite by Michael Norris [14], and the standalone Argeïphontes Lyre application by Akira Rabelais [15].

From here, the three Listeners involved in the session are referred to as Listeners 0, 4 and 5, as given by their IDs in `session_3_log.csv`. During this session, the Performer repeatedly cycled through three motifs: a series of regular eighth notes, followed by a period of near-silence, followed by chaotic playing. The complete work is divided into three sections that mirror each of these motifs. The first section of the piece (0:00 - 0:44) uses MIDI from Listeners 4 and 5; they are positioned at different distances from the Performer, creating a "call and response" effect in their combined parts. The second section of the piece (0:44 - 1:03) begins as the mallets fade out and the atmospheric sounds become louder. The third section (1:03 - 2:08) begins as the mallets fade back in, becomes more chaotic up to 1:40, then gets quieter as the piece fades out.

6 Conclusion

In this paper, we presented the design motivations, prototype implementation, and informal testing of a distributed web-based NMP system for collaborative improvisation and experimentation.a

The process by which latency is estimated in the prototype, while crude and not thorough, generated a reasonable set of values for use as avatar positions. However, future work will explore more accurate measurements of inter-user latency, potentially using an adaptive algorithm or more measurements fanned out over a longer period of time. User feedback also suggests a more legible scrolling score for Performers than the grid would be useful.

Future work could also include implementing functionality for more complex sound design. While this could be an interesting direction, we acknowledge that music producers are unlikely to abandon established sound design or sampling workflows for browser-based alternatives, and that this system is appealing for its use as a specialized collaborative environment where users can create distinctive music together during quasi-real time performances, rather than a full DAW. A more compelling approach is to focus on a wider range of post-session editing capabilities, or on performance capabilities that do not overwhelm the core collaborative experience, such as allowing users to upload their own custom drum samples. In the current prototype, which gives all clients the same drum samples, the grid is mostly responsible for helping users differentiate each others' contributions. Adding unique drum sounds could make this easier.

This work was motivated by a desire to address an underexplored perspective in the NMP field, namely, that of the "jamming" musician interested in generating material for use in independently developed compositions. Over time, interest has grown in embracing network latencies and disruptions as "crucial" characteristics of the network medium worth embracing, for both musical and social benefit [24, 29]. It is our hope that this work inspires more developers and technologists to exploit latency as a creative restriction, and to consider quasi-real time interaction as an ideation exercise for asynchronous composition practices.

[13]https://soniccharge.com/synplant
[14]https://michaelnorris.info/software/soundmagic-spectral
[15]https://akirarabelais.com/lyre/

7 Ethical Standards

This paper complies with NIME ethical standards [34]. Work on the software project whose implementation formed the basis of this system was funded by a grant from NetEase in Summer 2022. The system described here is not supported by any endorsement, and was carried out in pursuit of the first author's M.S. degree from Carnegie Mellon University, without additional project-specific funding from any source. All involved participants during test sessions provided informed consent.

References

[1] Philip Auslander. 2022. *Liveness: Performance in a Mediatized Culture* (3 ed.). Routledge, New York, NY, USA. https://doi.org/10.4324/9781003031314

[2] Christopher Bartlette, Dave Headlam, Mark Bocko, and Gordana Velikic. 2006. Effect of Network Latency on Interactive Musical Performance. *Music Perception: An Interdisciplinary Journal* 24, 1 (2006), 49–62. http://www.jstor.org/stable/10.1525/mp.2006.24.1.49

[3] Steve Benford, John M. Bowers, Lennart E. Fahlén, and Chris Greenhalgh. 1994. Managing mutual awareness in collaborative virtual environments. In *Proceedings of the Virtual Reality Software and Technology '94*. WORLD SCIENTIFIC, Singapore, 223–236. https://doi.org/10.1142/9789814350938_0018

[4] Tina Blaine and Sidney S. Fels. 2003. Contexts of Collaborative Musical Experiences. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (22-24 May, 2003). Montreal, Canada, 129–134. https://doi.org/10.5281/zenodo.1176490

[5] Chris Brown. 1998. Eternal Music Client. https://www.transjam.com/eternal/eternal_client.html.

[6] Chris Brown and John Bischoff. 2005. Computer Network Music Bands: A History Of The League Of Automatic Music Composers And The Hub. In *At a Distance: Precursors to Art and Activism on the Internet*, Annmarie Chandler and Norie Neumark (Eds.). MIT Press, Cambridge, Massachusetts, 372–392. in *At a Distance: Precursors to Art and Activism on the Internet*, Annmarie Chandler and Norie Neumark (ed.).

[7] Nick Bryan-Kinns. 2011. Annotating Distributed Scores for Mutual Engagement in Daisyphone and Beyond. *Leonardo Music Journal* 21 (2011), 51–55. http://www.jstor.org/stable/41416823

[8] Nick Bryan-Kinns. 2013. Mutual engagement and collocation with shared representations. *International Journal of Human Computer Studies* 71 (01 2013), 76–90. https://doi.org/10.1016/j.ijhcs.2012.02.004

[9] Marvin Carlson. 2004. *Performance: A Critical Introduction (2nd ed.)*. Routledge. https://doi.org/10.4324/9781315016153

[10] Chris Chafe. 2003. Distributed Internet Reverberation for Audio Collaboration. In *Proc. AES 24th Int. Conf.* Banff. https://ccrma.stanford.edu/~cc/shtml/dirac.shtml

[11] Chris Chafe, Scott Wilson, and Daniel Walling. 2002. Physical model synthesis with application to Internet acoustics. In *2002 International Conference on Acoustics, Speech, and Signal Processing*, Vol. 4. Orlando, FL, IV–4056–IV–4059. https://doi.org/10.1109/ICASSP.2002.5745548

[12] Cockos Incorporated. 2005. NINJAM. https://www.cockos.com/ninjam/.

[13] Tom Cortina. 2020. Introduction to Computing for Creative Practice, Fall 2020, Lecture 22: Mutual Interaction. https://www.cs.cmu.edu/ tcortina/15104-f20/lectures/.

[14] Luke Dahl. 2012. Wicked Problems and Design Considerations in Composing for Laptop Orchestra. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. University of Michigan, Ann Arbor, Michigan. https://doi.org/10.5281/zenodo.1178239

[15] Roger Dannenberg and Ari Liloia. 2022. Global Drum Circle (Private Github Repository).

[16] Roger Dannenberg and Tom Neuendorffer. 2000. Scaling Up Live Internet Performance with The Global Net Orchestra. In *International Conference on Mathematics and Computing*.

[17] Stefano Delle Monache, Luca Comanducci, Michele Buccoli, Massimiliano Zanoni, Augusto Sarti, Enrico Pietrocola, Filippo Berbenni, and Giovanni Cospito. 2019. A Presence and Performance-Driven Framework to Investigate Interactive Networked Music Learning Scenarios. *Wireless Communications and Mobile Computing* 2019, 1 (2019), 4593853. https://doi.org/10.1155/2019/4593853 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1155/2019/4593853

[18] Robert Ek. 2024. Playing with resistance. , Article 24 (September 2024), 6 pages. https://doi.org/10.5281/zenodo.13904814

[19] Golo Föllmer. 2005. Electronic, aesthetic and social factors in Net music. *Organised Sound* 10, 3 (2005), 185–192. https://doi.org/10.1017/S1355771805000920

[20] Leonardo Gabrielli and Stefano Squartini. 2016. *Wireless Networked Music Performance* (1 ed.). Springer Singapore.

[21] Gabriel Gambetta. 2024. *Fast-Paced Multiplayer (Part II): Client-Side Prediction and Server Reconciliation.* Retrieved December 13, 2024 from https://www.

gabrielgambetta.com/client-side-prediction-server-reconciliation.html

[22] Masataka Goto, Ryo Neyama, and Yoichi Muraoka. 1997. RMCP: Remote Music Control Protocol - Design and Applications. In *Proceedings of the 1997 International Computer Music Conference, ICMC 1997, Thessaloniki, Greece, September 25-30, 1997*. Michigan Publishing. https://hdl.handle.net/2027/spo.bbp2372.1997.118

[23] Lee B. Brown; David Goldblatt; Theodore Gracyk. 2018. *Jazz and the Philosophy of Art* (paperback ed.). Routledge.

[24] Jim Hollan and Scott Stornetta. 1992. Beyond being there. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Monterey, California, USA) *(CHI '92)*. Association for Computing Machinery, New York, NY, USA, 119–125. https://doi.org/10.1145/142750.142769

[25] Miriam Iorwerth. 2023. *Networked Music Performance: Theory and Applications* (1 ed.). Focal Press, New York, NY. https://doi.org/10.4324/9781003268857

[26] Alexander Refsum Jensenius. 2022. *Sound Actions: Conceptualizing Musical Instruments* (1st ed.). MIT Press, Cambridge, MA. https://doi.org/10.7551/mitpress/14220.001.0001

[27] Daniel Jones. 2023. AbletonOSC: A unified control API for Ableton Live. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Miguel Ortiz and Adnan Marquez-Borbon (Eds.). Mexico City, Mexico, Article 60, 5 pages. https://doi.org/10.5281/zenodo.11189234

[28] Sergi Jordà. 2004. Digital Instruments and Players: Part II – Diversity, Freedom and Control. In *Proceedings of the 2004 International Computer Music Conference*. Miami, USA, 706–709.

[29] Sergi Jordà, Daniel Gómez-Marín, Ángel Faraldo, and Perfecto Herrera. 2016. Drumming with style: From user needs to a working prototype. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Queensland Conservatorium Griffith University, Brisbane, Australia, 365–370. https://doi.org/10.5281/zenodo.1176048

[30] Peter Keller. 2008. Joint action in music performance. *Enacting intersubjectivity: A cognitive and social perspective to the study of interactions, 205-221 (2008)* 10 (01 2008).

[31] Scott R. Klemmer, Björn Hartmann, and Leila Takayama. 2006. How bodies matter: five themes for interaction design. In *Proceedings of the 6th Conference on Designing Interactive Systems* (University Park, PA, USA) *(DIS '06)*. Association for Computing Machinery, New York, NY, USA, 140–149. https://doi.org/10.1145/1142405.1142429

[32] Marc Leman, Pieter-Jan Maes, Luc Nijs, and Edith Van Dyck. 2018. What is Embodied Music Cognition? In *Springer Handbook of Systematic Musicology* (1st ed. ed.), Rolf Bader (Ed.). Springer International Publishing, Cham, Switzerland, 747–760. https://doi.org/10.1007/978-3-662-55004-5

[33] Tim Merritt, Weiman Kow, Christopher Ng, Kevin McGee, and Lonce Wyse. 2010. Who makes what sound? supporting real-time musical improvisations of electroacoustic ensembles. In *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction* (Brisbane, Australia) *(OZCHI '10)*. Association for Computing Machinery, New York, NY, USA, 112–119. https://doi.org/10.1145/1952222.1952245

[34] Fabio Morreale, Nicolas Gold, Cécile Chevalier, and Raul Masu. 2023. NIME Principles & Code of Practice on Ethical Research. https://doi.org/10.5281/zenodo.7545682

[35] Yuka Obu, Tomoyuki Kato, and Tatsuhiro Yonekura. 2003. M.A.S.: A Protocol for a Musical Session in a Sound Field where Synchronization between Musical Notes is not guaranteed. In *International Conference on Mathematics and Computing*. https://quod.lib.umich.edu/cgi/p/pod/dod-idx/m-as-a-protocol-for-a-musical-session-in-a-sound-field-where.pdf?c=icmc;idno=bbp2372.2003.016;format=pdf

[36] Yunyu Ong, Robert Sazdov, and Andrew Johnston. 2024. Opening DAWs to Interactive Music - Making an Orchestra out of Soloists. , Article 10 (September 2024), 8 pages. https://doi.org/10.5281/zenodo.13904782

[37] L.L. Peterson and B.S. Davie. 2021. *Computer Networks: A Systems Approach* (sixth ed.). Elsevier Science. https://book.systemsapproach.org/

[38] Ricardo F. Pinheiro. 2014. The Jam Session and Jazz Studies. *International Review of the Aesthetics and Sociology of Music* 45, 2 (2014), 335–344. http://www.jstor.org/stable/43198651

[39] Jon Prince, William Thompson, and Mark Schmuckler. 2009. Pitch and time, tonality and meter: how do musical dimensions combine? *Journal of experimental psychology. Human perception and performance* 35 (10 2009), 1598–617. https://doi.org/10.1037/a0016456

[40] Jun qi Deng, Francis Chi Moon Lau, Ho-Cheung Ng, Yu-Kwong Kwok, Hung-Kwan Chen, and Yu heng Liu. 2014. WIJAM: A Mobile Collaborative Improvisation Platform under Master-players Paradigm. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Goldsmiths, University of London, London, United Kingdom, 407–410. https://doi.org/10.5281/zenodo.1178746

[41] Pedro Rebelo and Robert King. 2010. Anticipation in networked musical performance. In *Proceedings of the 2010 International Conference on Electronic Visualisation and the Arts* (London, UK) *(EVA'10)*. BCS Learning & Development Ltd., Swindon, GBR, 31–36.

[42] Cristina Rottondi, Chris Chafe, Claudio Allocchio, and Augusto Sarti. 2016. An Overview on Networked Music Performance Technologies. *IEEE Access* 4 (2016), 8823–8843. https://doi.org/10.1109/ACCESS.2016.2628440

[43] Paul Sanden. 2013. *Liveness in Modern Music: Musicians, Technology, and the Perception of Performance*. Routledge. https://doi.org/10.4324/9780203078518

[44] Norbert Schnell and Sebastien Robaszkiewicz. 2015. Soundworks – A playground for artists and developers to create collaborative mobile web performances. In *Proceedings of the Web Audio Conference (WAC'15)*. https://hal.science/hal-01580797.

[45] Franziska Schroeder and Pedro Rebelo. 2009. Sounding the network: The body as disturbant. *Leonardo Electronic Almanac* Vol 16 Issue 4 – 5 (20 May 2009), 1. https://doi.org/LEA/DispersiveAnatomies/DA_schroeder-rebelo.pdf

[46] Agostino Di Scipio. 2021. *Thinking Liveness in Performance with Live Electronics: The Need for an Eco-systemic Notion of Agency*. Leuven University Press, 171–194. http://www.jstor.org/stable/j.ctv1ccbg96.12

[47] Steve Wurtzler. 1992. "She Sang Live, But The Microphone Was Turned Off:" The Live, the Recorded and the Subject of Representation. In *Sound Theory, Sound Practice*, Rick Altman (Ed.). Routledge, New York, NY, USA, 87–103.

[48] Lonce Wyse and Jude Yew. 2014. A Real-Time Score for Collaborative Just-in-Time Composition. *Organised Sound* 19, 3 (2014), 260–267. https://doi.org/10.1017/S1355771814000247

[49] Álvaro Barbosa. 2003. Displaced Soundscapes: A Survey of Network Systems for Music and Sonic Art Creation. *Leonardo Music Journal* 13 (2003), 53–59.