OPERA*craft*: Blurring the Lines between Real and Virtual

Ivica Ico Bukvic
Virginia Tech
School of Performing Arts
Institute for Creativity, Arts, and Technology
ico@vt.edu

Cody Cahoon
Virginia Tech
Computer Science
codyc@vt.edu

Ariana Wyatt
Virginia Tech
School of Performing Arts
ariana1@vt.edu

Tracy Cowden
Virginia Tech
School of Performing Arts
tcowden@vt.edu

Katie Dredger
James Madison University
College of Education
dredgemk@jmu.edu

ABSTRACT

In the following paper we present an innovative approach to coupling gaming, telematics, machinima, and opera to produce a hybrid performance art form and an arts+technology education platform. To achieve this, we leverage a custom Minecraft video game and sandbox mod and pd-l2ork real-time digital signal processing environment. The result is a malleable telematic-ready platform capable of supporting a broad array of artistic forms beyond its original intent, including theatre, cinema, as well as machinima and other experimental genres.

1. BACKGROUND

Making art with found technologies is as old as art making itself. Therefore it comes as no surprise that video games, gaming engines, and virtual 3D environments are being used to produce movies beyond their original intent. We refer to this form of art as machinima [1][2]. More recently, with the emergence of the sandbox video game genre, most notably the ubiquitous Minecraft [3], lines between entertainment, creativity, and learning are all but gone. Today, online video channels like YouTube [4] are increasingly populated with in-game footage exploring various virtual 3D environments in a sandbox-like fashion, coupled with recordings of conversations among players who are there simply sharing their personal reactions to the ensuing adventure. Arguably these can be seen as a subset of machinima with first-person point of view and minimal post-production.

Minecraft, as a signature example of a sandbox-game hybrid has seen a widespread adoption in various learning contexts [3][5][6][7][8][9] including the most unsuspecting uses, such as 3D printing [10] and rendering 3D video feed from Kinect [11]. The inherent malleability and a stylized low-threshold visual design invites users to tinker with blocks, shapes, textures, sounds, behavior, etc. [12]. Of particular interest are music videos that use custom renditions of popular pop songs with Minecraft-centric lyrics where due to limitations of in-game characters' expressions (mouth movement, body gestures, emotions, etc.) the videos are often rendered using professional 3D modeling tools that work hard at recreating the 8-bit-style graphics of the surrounding environment, while making characters considerably more elaborate [13][14]. Legal ramifications aside, the popularity of these tunes has reached such proportions that they can now be purchased from online music stores, such as iTunes. Another notable aspect of Minecraft is its robust online network code--it is not uncommon to participate in online environments with thousands of players present, something that even today very few online games can scale to.

It is worth noting a significant divide between machinima renditions such as the aforesaid music videos versus the first-person in-game footage presented earlier. This is particularly potent given a rich modding community that (save for a few isolated efforts [15]) has steered away from modding the character features to allow them to be more expressive. It appears that having similar set of features within the gaming engine itself would open doors for a seemingly unique set of opportunities where the gaming environment could become synonymous with a more complex production environment, akin to that of a post-produced machinima, while concurrently leveraging the multiplayer and consequently massive online real-time participation and/or observation of such a production.

2. MOTIVATION: INSTANT OPERA

There is a significant body of evidence showing that in-depth exposure to the arts has remarkable, far-reaching effects. Students in quality art programs benefit from a wide range of positive effects including development of creativity and thinking skills, better self-expression, appreciation of art and music, learning about other cultures, and enriched personal satisfaction with their achievements [16]. The particular genre of opera outreach–involving non-musicians in the creative process–is being done around the world. Wolf Trap Opera (Vienna, VA)

has a program where the K-12 audience chooses a story and the professional opera singers improvise this story on the spot. Another program in Cardiff, Wales works with homeless adults. Their motto is "giving homeless people a voice," and they work to help the homeless gain confidence and self-esteem. In developing our own initiative designed to engage high school students in the creative process, we added a significant technological component. We envisioned an environment where students could write a story and libretto, build a virtual set, costume virtual characters, and ultimately control the characters within the virtual setting in a live performance. Therefore we assembled a team of professors, graduate students, and undergraduates to guide a group of high school students in creating an opera. Our team consisted of two professors from Teaching and Learning (Katie Dredger, English Education and Kelly Parkes, Music Education), three professors from the Department of Music (Ariana Wyatt, Voice; Tracy Cowden, piano; and Ivica Ico Bukvic, Music Technology), a graduate student in stage management (Amy Luce), and an undergraduate computer science major (Cody Cahoon). Below we primarily focus on project's technical component developed by Bukvic and Cahoon.

3. INTRODUCING OPERACRAFT

In an attempt to identify optimal 3D virtual environment that would support the Instant Opera paradigm, Bukvic suggested the use of Minecraft. The obvious advantages included a game-sandbox hybrid offering a vibrant and diverse creative community, from modders to artists. More so, with its popularity among the target population, the environment has a proven educational track, serving as a potent retention catalyst. Minecraft was not without limitations, however, many of which are described in the Background section. In addition, Minecraft lacked access to the original API, with the only option at the time being community-driven effort to decompile JAVA runtime into a human-readable API. Consequently, modding limitations were not entirely clear, requiring further tests and assessments, and ultimately leading to implementations that may be seen more as a workaround rather than a maintainable feature. Nonetheless, following initial assessment, the research team found Minecraft a favorable foundation and consequently decided to name the project OPERA*craft*.

4. IMPLEMENTATION

For OPERA*craft*, we wanted to leverage Minecraft's core facilities that are easily accessible to our target audiences, such as an ability to design "skins" (textures), collaborative scene design, and out-of-box multiplayer support with chat and other core functionalities. Some features were left intact, while others required changes that ranged from fine-tuning to complete redesign. Such improvements are further discussed below along with new additions. All modifications are based off of the Minecraft 1.5.2 codebase.

One of the focal challenges was to make Minecraft as comprehensive of an environment for real-time video production. This meant implementing new features that would allow existing facilities to serve as a camera feed without extraneous and/or distracting GUI widgets, as well as allowing for multiple simultaneous camera views one could easily switch between and broadcast on the main camera view or projector. The ensuing camera views were essentially additional players that were visible to actors in a form of solid color characters and provided visual cues (switching their head colors from red to green), so that actors can at all times know where are all cameras are located as well as which camera is currently active. The same camera characters were rendered invisible to the projector whose view automatically latched onto the desired camera view and had no GUI markers that would detract from an impression of a genuine camera feed. The projector was invisible at all times and was not interacted with directly beyond being issued commands to latch onto camera views. The only exception were chat messages that were repurposed for subtitles allowing one line at a time (so as to limit the on-screen clutter). Concurrently, all acting players within the game were allowed to receive additional chat messages from the stage manager that were visible only on their screens, while projector ignored them.

A collection of new features necessary for seamless scene changes included fade ins and outs (implemented as a workaround by making the projector wear an increasing number of semi-transparent "helmets"), as well as an ability to instantly teleport performers and cameras to a predetermined set of coordinates to minimize transition times.

Key	Action
j	Both arms are in default (down) position (Fig.1.a)
k	The left arm is placed directly out (Fig.1.b)
l	The right arm is placed directly out (Fig.1.c)
u	Both arms are up (Fig.1.d)
i	The left arm is placed at an left-upward angle (Fig.1.e)
o	The right arm is placed at a right-upward angle (Fig.1.f)
h	Hides/shows the on-screen GUI
y	Hides/shows the on-screen hints for arm movement/positioning

Table 1. Client-side key mapping.

As a subset of the aforesaid challenge, the technical team sought ways to make acting more expressive and closer to the levels of post-produced machinima. At the very core this consisted of mouth movement and arm gestures. Other, finer adjustments included removal of player decals and other visual notifications that may detract from the immersion (from audience's vantage point). In order to identify camera players from performers and the main camera view (projector) and thus restrict

behaviors of each client in respect to aforesaid features, the mod compared client usernames to a hardwired database. In the current mod version, this and other settings have been abstracted as separate configuration text files that allow for mod's easy repurposing for other productions.

Arm movement was handled client-side. Namely, each user had access to an array of arm movements that were mapped symmetrically across the keyboard targeting each arm separately. Given that arm motion was interpolated, more complex arm motions were possible by triggering various arm positions in the middle of another animation. As a result the ensuing mod provided additional key mappings described in Table 1 and further elaborated upon in the Figure 1 below. The most recent mod version further allows for user to determine the speed of arm movement that is determined by the length of the key press—the longer the key press, the slower the arm movement. This also means that the arm movement commences once the key has been released and thus requires preemptively timed key presses on slower arm motions.

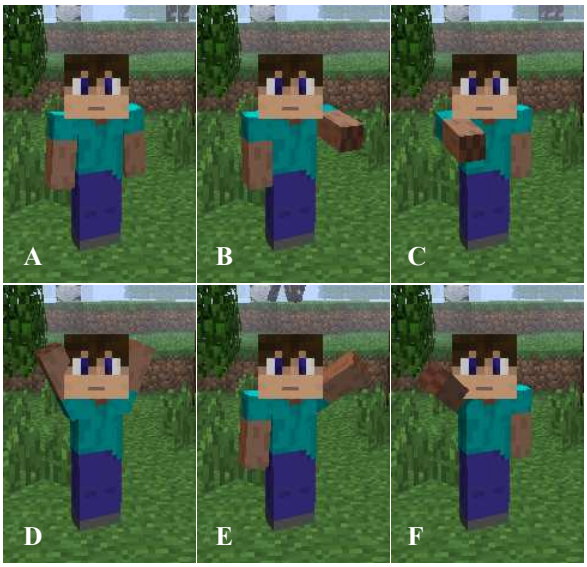

Figure 1. Avatar arm positions (see table 1 above for additional explanation).

The visual aspect of mouth movement/animation was implemented client-side, while the DSP component was devised in Pd-L2Ork [17][18][19][20]. We will discuss the DSP component in the next section below. In order to avoid making significant changes to the Minecraft codebase, we opted for designing animated mouths as a collection of "helmets" that are rapidly altered. As a result, Minecraft characters had larger than usual heads that were now painted on top of the virtual helmet placed on top of their real heads. Although a workaround, the choice was in good part driven by the fact that face texture is coupled with the rest of the skin texture, making it difficult to update dynamically. Thus the implementation has paved way towards more complex animations further down the road, such as eye animations (blinking, wincing, sad or happy, etc.) and other facial expressions.

4.1 Connecting Pd-L2Ork and Minecraft

Given the Minecraft engine did not provide core audio DSP facilities necessary for mouth movement to match that of singers, with focus on rapid prototyping, we set out to provide a networked interface between Pd-L2Ork and Minecraft mod OPERA*craft* that would allow us to feed DSP data into the mod and alter the environment. To achieve this, we relied on Pure Data [21] and by extension Pd-L2Ork's FUDI protocol [22] which resembles a simplified version of the Open Sound Control (OSC) protocol [23]. This allowed us to do audio processing inside Pd-L2Ork and feed the ensuing mouth movement data into Minecraft. All data was broadcast across the subnet using UDP packets (x.x.x.255 address) to minimize configuration issues and sidestep the necessity of specifying receiving client's IP address. The mouth movement was not the only use for the networked protocol and therefore the aforesaid broadcast of networked data also paved way towards splitting various production tasks among multiple broadcasting clients, which proved instrumental in the final production.

Command/Syntax	Result
<client> @mouth <mouth position 0-4>	The specific client's mouth is changed to position 0-4 (Fig.2)
@warn <message>	Stage cues are displayed only on actors/players' screens
@tpall <teleport position 0-6>	Teleports actors/players to a position before a new scene (currently hardwired inside the mod)
<client> @text <message>	Displays a new subtitle for 10 seconds from specified player (synonymous to a client posting a chat on their own)
<client> @fade <fade level 0-15>	Fades the client's screen ranging from 0 (clear) to 15 (completely black)
@time <0-18000>	Sets the in-game time to the number specified (0=dawn, 18000=midnight)
@view <client>	Makes the projector player take over the specified client's view

Table 2. Networked messages syntax and results.

Considering the production team within the physical performance space (where virtual opera was displayed on a large screen and out of which it was broadcast live out into the world) was scattered across the performance space, with some participants located in control booths and catwalks, and others on stage, having multiple networked machines broadcasting newfound protocol data allowed us to minimize requirement for co-location. As a

result, we utilized three broadcasting clients, one for mouth movement and scene control, second for stage management messages, and third for subtitles, thus allowing for workload to be split evenly among multiple production staff members.

Since the Minecraft user names were already internally hardwired in order for the Minecraft mod to be able to autoconfigure itself to a specfic role, we chose to use main camera view client (projector) as the entry point for all networked data and from there distribute its behavior via chat client. While this solution may be seen largely as yet another workaround due to incomplete access to the underlying Minecraft API, doing so has also allowed us to exchange relevant client-side data packets via the same protocol without requiring us to design another multiplayer data packet for each of the desired behaviors. This was achieved using reserved commands that were prepended with an "@" hook as described in the Table 2 above. Since Minecraft's chat engine invisibly embeds client username into all chat commands (1st argument in the syntax in Table 2), all users would have ability to also issue commands locally via chat as a means of easy debugging and/or local control (e.g. situations where each client processes its own voice in situ) and make them active across the entire networked ecosystem, the only shortcoming being clients who may be joining later who would not be able to retrieve preexisting states of players already in-game. As the production expected all players to be present from start, this was not seen as an issue worth addressing in this iteration.

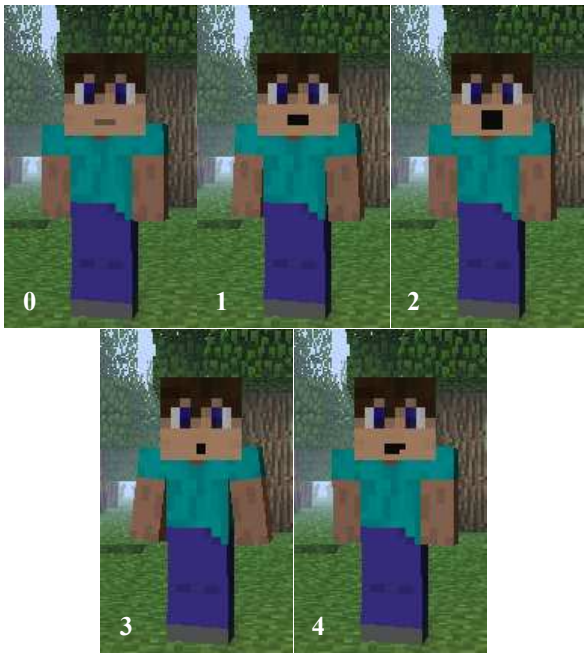

Figure 2. Avatar mouth animation states from top-left to bottom-right: 0 (mouth closed), 1 and 2 (alternating; mouth open singing a vowel-based melisma), and 3-4 (random; transients and/or significant pitch delta).

4.2 Pd-L2Ork Component

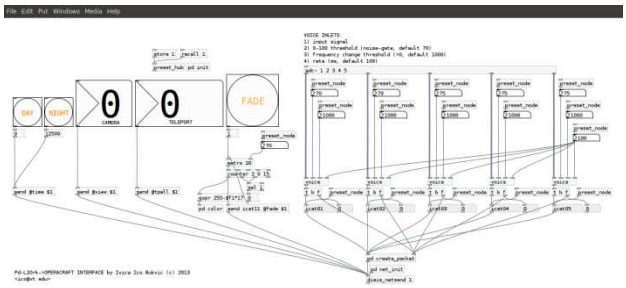

Figure 3. Pd-L2Ork patch with mouth animation and scene control engine.

Pd-L2Ork's primary role was to provide voice analysis and mouth animation (Figure 3). Secondary functions included distribution of production team's roles as well as scene and camera control (Figure 3). Voice analysis relied on audio streams captured using one microphone per vocalist. It utilized frequency of zero crossings per buffer of captured audio data and by doing so extrapolating presence of transients. Given the simplicity of animation (mouth relied on 4x4 pixel area, limiting our ability to visibly project nuances of pronunciation), we did not invest time in further isolating formants or any of the more advanced speech analysis features.

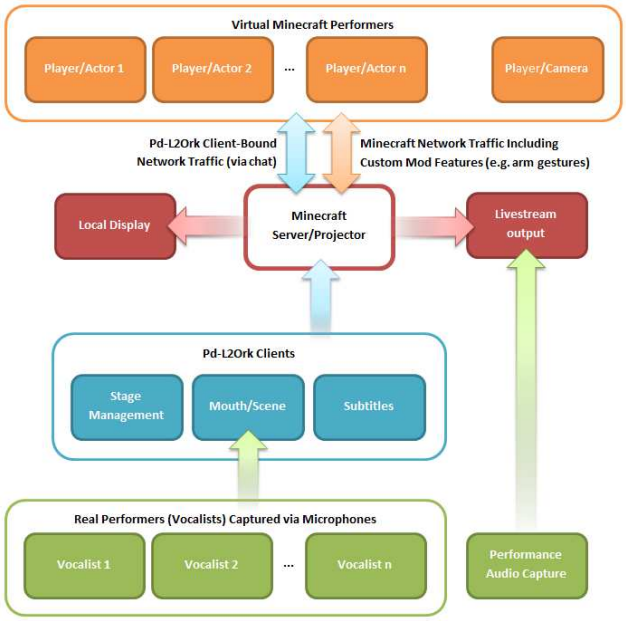

Figure 4. Final Technical Setup Diagram.

For the initial production there were a total of 5 mouth states that generated a cartoon-like animation: 0 being the closed mouth; 1 and 2 (sequentially alternating) being open mouth vibrating in operatic style, reflecting a long melisma on an open vowel; while 3 and 4 being randomly picked when detecting transients. Scene control interface (coupled with the mouth detection, shown on Figure 2 above) relied on networked messages to invoke fade-ins and outs and teleport players and camera to new positions. Two additional clients were provided for subtitles and stage management cues, both of which were embed-

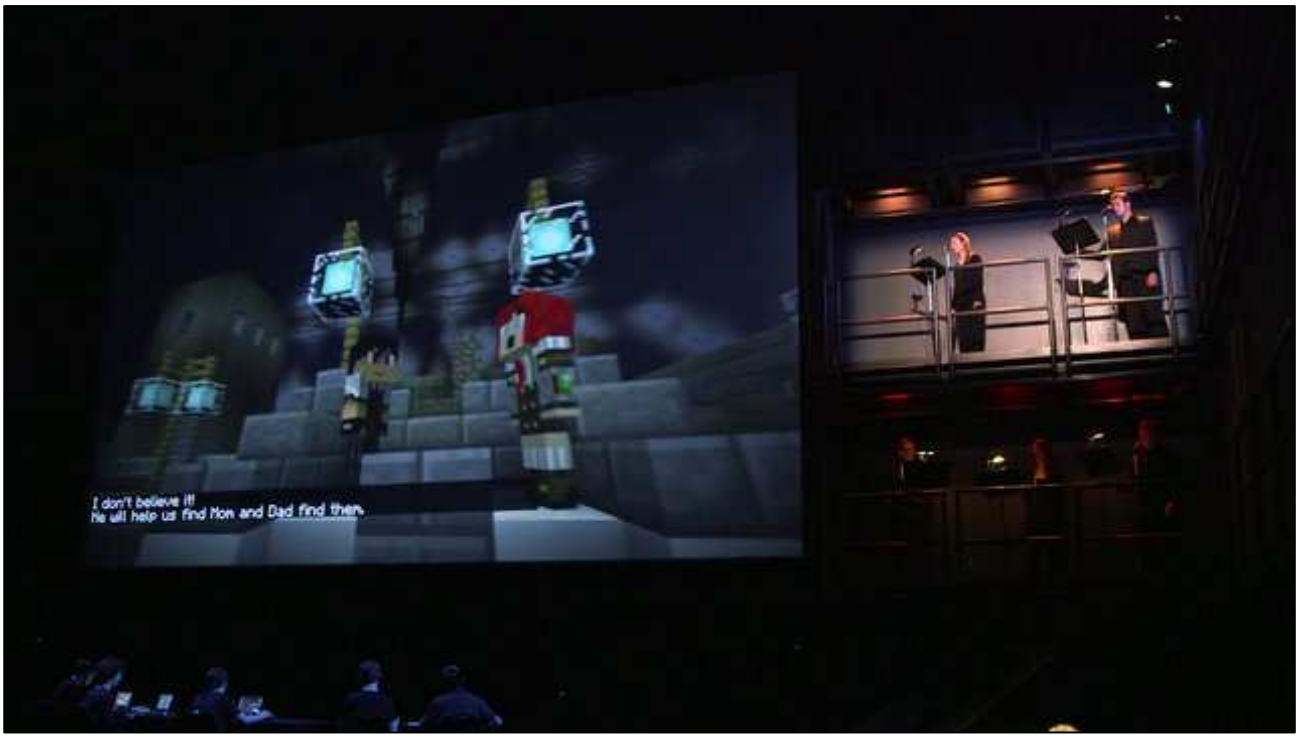

Figure 5. Photo from the premiere in the Moss Arts Center Cube with five vocalists on two catwalk levels (right), pianist (bottom right), five virtual performers and one cameraman (bottom left), and a large projection screen (left). The rest of the production team (not pictured here) was distributed across various catwalk levels and their respective control booths.

ded in pd-l2ork's "coll" object and broadcast sequentially with a push of a button. Consequently, the ensuing technical setup layout is depicted in Figure 4.

5. OPERA PRODUCTION

From a learning perspective, ten high school students participating in OPERA*craft* production engaged in a series of scaffolded creative exercises designed to build a full-fledged opera. They started with a libretto.

In order to compose the libretto, the students first collaborated in a workshop setting to determine the theme of the piece and the message that they were interested in exploring or challenging in regards to human existence [24][25]. Together, the students first identified the theme, characters, and setting before moving on to plot elements. Inspired by dystopian and coming-of-age fiction known for its popularity among adolescents [26], the student participants chose a person vs. person conflict in order to show resistance toward evil authority. This evil authority was personified in Emperor X in a post-apocalyptic underground world. The student libretto authors, familiar with Minecraft, conceived of the plot with a particular setting in mind. The libretto coach, a former high school English teacher, assigned pairs of students to specific scenes to mentor them through this process.

Building the set within the virtual environment proved a powerful engagement catalyst and has largely influenced the plot development. Using in-game editing tools and a third party external editor MCEdit [27], students transformed the world from open plains to an underground cave, filled with stalactites and dawned with stone and rundown houses which represented the poor and rich classes of this underground world, and finally centered with a monumental, villainous castle in which numerous scenes take place.

When dealing with the creation of custom characters' costumes, one costume for each character was a must. Students wanted the costumes to appear as if the in-game characters were from older times, but also wanted their outfits to seem relevant. The final costumes were chosen from an online Minecraft "steampunk" skin catalog.

Because exposition and character development were difficult within the short time constraints, the dialog of the libretto had to focus on plot movement instead of the building of pathos. Like a short story, the falling action and resolution of the libretto were tightly linked and left the audience with a feeling of a lack of closure. The student authors of the libretto, after discussion, agreed that this ending was necessary based on time constraints. Furthermore, the lack of development in the conclusion of the plot was purposeful in that it inspired interest in a sequel featuring younger brother Marcus as the protagonist, as well as fan fiction.

While the students were working on development of the characters and the story, Cowden and Wyatt researched excerpts from Mozart opera that might be appropriate for each section of the libretto, searching for selections that had the appropriate dramatic feel for each section. The focus was on solos, duets, and trios that would match as closely as possible what the students were creating in the libretto. This is, of course, backwards to the normal process of writing an opera, but worked rather well within

the time constraints and in a setting with students who are not musically proficient. Selections from several different Mozart operas were brought to students' attention, offering them some choices, and demonstrated how these musical selections might work for each scene. The students used their ideas about character and story development to match music selection with a particular scene. In some cases they resorted to adapting the libretto to better match the musical constraints.

Once the libretto draft was completed, the cast met with Cowden to coach the music. During this process editing continued, and words and rhythms were changed to be more clearly understood in a sung context and therefore reinforce the plot clarity. As per opera's tradition, musical rehearsals were followed by staging. In the staging rehearsals the high school students practiced controlling their avatars and the supporting infrastructure of the newfound OPERA*craft* mod, while the vocalists sang their parts. Wyatt helped guide the high school students in controlling their characters effectively in order to provide clear storytelling. Cowden created transitional music material to make scenes flow seamlessly together. She also introduced instrumental excerpts to serve as an overture at the beginning of the opera as well as instrumental-only music for the "fight scene" and the conclusion of the story. Again, the students were presented with choices and selected music based on what they felt best fit their libretto. Finally, collegiate singers suggested adding wordless humming to the final scene, which incorporated music from Mozart's Requiem. The staging rehearsals included a significant amount of time rehearsing the "fight scene," at which point we included a rehearsal with a stage combat director, Cara Rawlings, who coached the high school students in how to adapt stage combat techniques for their Minecraft characters.

Throughout the staging part of the rehearsal, the show's main camera view was rehearsed and controlled by a Virginia Tech student who had previously worked on the project before graduating. The cameraman used a combination of flying, strafing, moving, running, and crouching to create different angles throughout the performance. The cameraman also took advantage of the Minecraft bow, which zooms in when in use. This was used to zoom in on characters during longer scenes, such as during the character Lilith's aria.

5.1 Premiere

The premiere involved five unique characters, one camera player, five student vocalists accompanied by a piano reduction of aria arrangements, and five production staff manning the technology. In addition, the production also relied on a number of Virginia Tech Moss Arts Center staff members for lighting, rigging, and ushering needs. The opera was performed twice in front of a standing room only audience. Performances were also streamed live via livestream bringing additional 7,810 unique viewers. A recording of one of the opera livestreams can be viewed at <http://youtu.be/BCFKgffSdwM>. The event garnered a considerable amount of attention through mainstream and online media (e.g. [28][29][30][31][32]).

5.2 System Performance

Given that the networked system was on its own dedicated Ethernet, network packet latency among different clients was less than 5ms and as such did not play a major role in overall system's latency. Pd-L2Ork's transient detection did not require complex calculations but did rely on a generous buffer size. Hence its latency was limited by the audio buffer size which was approx. 93ms (4096 bytes at 44,100Hz sampling rate), producing a ~10fps mouth animation. Other Pd-L2Ork features, like camera control and subtitles produced minimal (network-related) latency between Pd-L2Ork and the Minecraft engine, resulting in responsive low-latency video production system. Despite a generously-sized audio buffer for transient and amplitude analysis which generated largest theoretical latency, the overall experience did not appear delayed, in part because its output manifested in a visual domain.

6. CONCLUSIONS

The project has resulted in an engaging outreach arts experience for eight high school boys, and two live sold-out performances of an opera that concurrently unfolded both in real and virtual worlds. The performances were streamed live and viewed around the world and had over thirty thousand hits in the month following. The ensuing opera production has received positive acclaim and we have had numerous requests for a sequel.

Apart from its artistic and outreach impact, the project bore another deliverable, the Minecraft-Pd-L2Ork hybrid mod OPERA*craft*—a malleable technology whose features enable Minecraft's in-game facilities to approach that of a post-produced machinima. Consequently, the research team envisions the ensuing implementation being appropriate in a broad range of live and post-production scenarios, beyond its original intent, from machinima movie-making to theatre. The same also offers interesting opportunities at extending virtual presence and consequently outreach by allowing audience to engage with the production directly in-game. The existing mod offers unique opportunities for observers to study action from their own personalized vantage point in addition to predetermined camera views, paving way towards more immersive ways of experiencing telematic performances [33]. The same technology also has the potential to serve as a means of archiving and revisiting past performances in an immersive and easily accessible format.

Although we utilized most of the facilities offered by the newfound OPERA*craft* mod, some features remained underexploited mainly due to time constraints, leaving room for further enhancement of future productions. One of those was multiple camera views that were scrapped due to lack of adequate rehearsal time. Another produc-

tion-level consideration includes improved microphone and vocalist placement to limit cross-contamination between different audio streams and thus preventing false positives in mouth animations.

Based on the experiences obtained through the OPERA*craft* premiere, we also observe the following limitations that will need to be addressed in the near future to ensure that the mod can continue to scale with newer versions of Minecraft. Namely, the mod in its current state relies exclusively on the community-driven API based on the version 1.5.2 that may in the long run limit code's upstream compatibility. Some features had to rely on last-minute workarounds, such as inconsistent teleporting that required two consecutive commands to ensure that both cameras and the projector have switched positions. In a production user-specific behaviors were hardwired, limiting ability for the program to be easily applied in different scenarios. In the latest iteration, however, these components have been extracted into separate text-based configuration files and are now user-configurable. The chat system may require additional filtering to prevent internal Minecraft notifications on the main camera view (e.g. "character xyz died"), while also allowing players to chat among each other without making such messages visible on the main camera (projector) view. Another shortcoming of the chat-based implementation for custom commands is that clients who may be joining later will not be able to retrieve preexisting states of players already in-game (as would be the case with telematic visitors/observers who may join in midway through the play/performance). This is something that will have to be looked at in the next iteration.

6.1 Obtaining OPERA*craft*

OPERA*craft* is envisioned as a freely available open source project designed to promote outreach and education in its broadest sense. The latest iteration of OPERA*craft* mod and supporting Pd-L2Ork patches are available at OPERA*craft*'s website [34]. For additional information on the mod contact technical director Bukvic.

7. FUTURE WORK

Apart from the shortcomings identified in the previous section, in the coming months the team will look into further expanding online resources with supporting documentation, with the focal intent of promoting crowdsourcing further development, including aforesaid enhancements. Most notably, we are looking to further broaden the expressive potential of in-game avatars.

We also envision the ability for telematic spectators to observe production in-game, either by latching onto one of the camera views or by allowing them to freely explore action from their own desired angle. While this is technically already possible, we would like to address the aforesaid challenge of observers who are joining late and whose client may not have all the up-to-date states for individual players and/or cameras.

Another desired feature would be the ability to replay action, which would require logging movement and action data from all in-game players (excluding aforesaid virtual audience members). This, however, may prove tricky in respect to non-player characters (NPCs) whose unpredictable spawning and movement will be difficult to reproduce without significant alterations to the Minecraft codebase.

8. ACKNOWLEDGMENTS

The research team would like to hereby thank Virginia Tech's Institute for Creativity, Arts, and Technology (ICAT) for funding this initiative as well as both ICAT and Virginia Tech Moss Arts Center for staffing and support of this event. Likewise, we would like to thank the students involved in the project, as well as Minecraft, Pd-L2Ork, and Pure-Data communities without whom this project would not have been possible.

9. REFERENCES

[1] H. Hancock and J. Ingram, *Machinima for dummies*. John Wiley & Sons, 2007.

[2] D. Morris, M. Kelland, and D. Lloyd, *Machinima: Making animated movies in 3D virtual environments*. Muska & Lipman/Premier-Trade, 2005.

[3] S. C. Duncan, "Minecraft, beyond construction and survival," *Well Play. J. Video Games Value Mean.*, vol. 1, no. 1, pp. 1–22, 2011.

[4] J. Burgess and J. Green, *YouTube: Online video and participatory culture*. John Wiley & Sons, 2013.

[5] J. Brand and S. Kinash, "Crafting minds in Minecraft," *Educ. Technol. Solut.*, vol. 55, pp. 56–58, Aug. 2013.

[6] C. Schifter and M. Cipollone, "Minecraft as a teaching tool: One case study," *Soc. Inf. Technol. Teach. Educ. Int. Conf. 2013*, vol. 2013, no. 1, pp. 2951–2955, 20130325.

[7] B. Cotton, *Minecraft: guided emergent game design*. 2011.

[8] D. Short, "Teaching scientific concepts using a virtual world-Minecraft.," *Teach. Sci. J. Aust. Sci. Teach. Assoc.*, vol. 58, no. 3, 2012.

[9] J. H. L. Lee-Leugner, "Youth, gaming, and the network society: exploring the agentic potential of gameplay in Minecraft," Thesis, Communication, Art & Technology: School of Communication, 2013.

[10] J. Aron, "Minecraft videogame blurs borders of 3D-printing," *New Sci.*, vol. 211, no. 2823, p. 20, 2011.

[11] "Minecraft + Kinect : Building Worlds!," 21-Jan-2011. [Online]. Available: http://www.youtube.com/watch?v=x2mCDkqXki0&feature=youtube_gdata_player. [Accessed: 30-Mar-2014].

[12] A. Leavitt, "The Source of Open-Source Culture: Participation in the Production of an Open Media

Artifact, Minecraft," *Sel. Pap. Internet Res.*, vol. 3, no. 0, Oct. 2013.

[13] "♫ 'Wrecking Mob' - A Minecraft Parody of Miley Cyrus' Wrecking Ball," 05-Nov-2013. [Online]. Available: http://www.youtube.com/watch?v=UiKOf4jNbm8&feature=youtube_gdata_player. [Accessed: 30-Mar-2014].

[14] "'Dragons' - A Minecraft Parody of 'Radioactive' By Imagine Dragons (Music Video)," 08-Jan-2014. [Online]. Available: http://www.youtube.com/watch?v=MfERdhRCUps&feature=youtube_gdata_player. [Accessed: 30-Mar-2014].

[15] "Minecraft - Smart Moving Mod Demo," 13-Dec-2011. [Online]. Available: http://www.youtube.com/watch?v=hjmmQ0aTAZQ&feature=youtube_gdata_player. [Accessed: 30-Mar-2014].

[16] E. W. Eisner, *The arts and the creation of mind*. Yale University Press, 2002.

[17] I. Bukvic, "A Behind-the-Scenes Peek at World's First Linux-Based Laptop Orchestra – The Design of L2Ork Infrastructure and Lessons Learned," presented at the Linux Audio Conference, Stanford, California, 2012, pp. 55–60.

[18] I. Bukvic, T. Martin, E. Standley, and M. Matthews, "Introducing L2Ork: Linux Laptop Orchestra," presented at the New Interfaces for Musical Expression, 2010, pp. 170–173.

[19] I. Bukvic, T. Martin, and M. Matthews, "Moving Beyond Academia Through Open Source. Solutions–Introducing L2Ork, Virginia Tech's Linux Laptop Orchestra," *J. SEAMUS*, 2011.

[20] I. Bukvic, "Virginia Tech Department of Music L2Ork - Software - Linux Laptop Orchestra." [Online]. Available: http://l2ork.music.vt.edu/main/?page_id=56. [Accessed: 08-Feb-2012].

[21] M. Puckette, "Pure Data: another integrated computer music environment," *Proc. Int. Comput. MUSIC Conf.*, pp. 37–41, 1996.

[22] "FUDI," *Wikipedia, the free encyclopedia*. 30-Mar-2014.

[23] M. Wright, "Open Sound Control-A New Protocol for Communicationg with Sound Synthesizers," in *Proceedings of the 1997 International Computer Music Conference*, 1997, pp. 101–104.

[24] H. W. Balk, "The Craft of Creating Opera Restoring a Lost Legacy through the Workshop Process," *Opera Q.*, vol. 1, no. 2, pp. 91–108, 1983.

[25] M. Ballam, "Creating the LIBRETTO," presented at the Utah Festival Opera & Musical Theatre, 2014.

[26] C. Hintz and E. Ostry, *Utopian and Dystopian Writing for Children and Young Adults*. Routledge, 2013.

[27] "MCEdit, a Minecraft World Editor." [Online]. Available: http://www.mcedit.net/. [Accessed: 01-Apr-2014].

[28] J. Scalzo, "Operacraft, not surprisingly, brings opera to Minecraft," *Warp Zoned*. .

[29] D. Haglund, "Minecraft, the Opera," *Slate*, 09-Dec-2013.

[30] J. Inverne, "Virginia Tech's OPERAcraft Brings Mozart to Minecraft, the Computer Game," *Classicalite*. [Online]. Available: http://www.classicalite.com/articles/4326/20131209/virginia-techs-operacraft-brings-mozart-to-minecraft-the-computer-game.htm. [Accessed: 01-Apr-2014].

[31] S. Tang, "Virginia Tech's New Minecraft Machinima: OPERAcraft (It's Exactly What You Think It Is)." [Online]. Available: http://www.gameskinny.com/wtbmr/virginia-techs-new-minecraft-machinima-operacraft-its-exactly-what-you-think-it-is. [Accessed: 01-Apr-2014].

[32] A. Mac, "OPERAcraft," *AMT Lab @ CMU*. [Online]. Available: http://amt-lab.org/blog/2013/9/case-study-operacraft. [Accessed: 01-Apr-2014].

[33] S. Dixon, *Digital performance: a history of new media in theater, dance, performance art, and installation*. MIT Press (MA), 2007.

[34] C. Cahoon, "OPERAcraft," *OPERAcraft*. [Online]. Available: http://disis.music.vt.edu/OPERAcraft/. [Accessed: 13-Jul-2014].