MORODD: Musical Paradox discovery in gaussian splat representation – An interactive installation to discover sonic territories

Debayan Mukherjee

M.S. in Media, Arts, and Science. Arizona State University, dmukhe16@asu.edu

Akanksha Pawar

Ph.D. in Media, Arts, and Science. Arizona State University, akpawar2@asu.edu

ABSTRACT

This paper presents MORODD (Morphed Oddity), an interactive installation that transforms human presence into an abstract representation contained within which are audio paradoxes. MORODD(MD) integrates computer vision, 3D Gaussian splatting, and Audio style transfer using Machine Learning (ML) methodologies and algorithms. The installation explores the philosophical tension between surveillance and creativity by converting human observers into point-cloud representations and embedding impossible musical genres within these virtual spaces containing musical genres that have never been heard or experienced. Through three progressive experiences – reflection, discovery, and collaboration; MD investigates how the act of observation changes both visual and auditory perception. Our system employs YOLO segmentation for human detection, Splatt3r for 3D reconstruction, and novel musical style transfer algorithms to create genre paradoxes such as "add here". The installation demonstrates how special audio positioning within Gaussian splat representations can create immersive experiences where musical contradictions become creative forces.

CCS CONCEPTS • Insert your first CCS term here • Insert your second CCS term here • Insert your third CCS term here

Additional Keywords and Phrases: Human Computer Interaction, Human Behavior, XR, Neural Network, RNN, Music Style Transfer, Spatial Audio.

1 INTRODUCTION

The intersection of surveillance technology and creative expression has historically produced tension between observation and authenticity. Computer vision systems designed for identification and classification paradoxically offer new possibilities for artistic transformation and self-reflection. MD emerges from this philosophical tension, proposing that spaces between people contain impossible music, sonic territories that exist only when witnessed and heard only when sought.

This work contributes a framework that transforms human presence into musical paradoxes through three interconnected technologies: human segmentation using Yolo[1], 3D Gaussian splatting via Splatt3r[2], and AI-driven musical style transfer for genre paradox generation. The installation challenges conventional notions of musical categorization by creating impossible fusions - genres that "shouldn't exist but do" - and embeds these within immersive 3D point-cloud representations of human observers.

The central research question driving this work is: What if the act of observation changes not just what we see, but what we hear? This inquiry leads to three specific investigations:

RQ1: How can real-time computer vision transform human presence into navigable virtual spaces?

RQ2: Can musical style transfer create meaningful paradoxes that transcend traditional genre boundaries?

RQ3: How does spatial positioning within one's own point-cloud representation affect the perception of embedded impossible music?

Our contributions include: (1) A pipeline integrating human segmentation, 3D reconstruction, and musical style transfer for interactive installations, (2) A framework for generating and spatially embedding musical paradoxes within Gaussian splat representations, (3) An investigation of how self-observation in virtual space affects musical perception and engagement, and (4) Design patterns for creating immersive experiences that challenge the boundaries between surveillance and creativity.

2 RELATED WORK

The convergence of real-time 3D reconstruction, AI-driven musical transformation, and spatial audio positioning enables new forms of interactive installations that challenge conventional boundaries between observation, embodiment, and sonic exploration. The following work reviews each of the elements seperately as elements to create art.

2.1 Computer Vision for Interactive Art

The application of computer vision in interactive art installations has evolved from simple motion detection to sophisticated systems that understand human behavior. Recent advances in real-time human segmentation, particularly through YOLO-based architectures[3], have enabled installations that respond dynamically to human presence while maintaining computational efficiency necessary for real-time interaction.

The Splatt3r framework[2] represents a significant advancement in real-time 3D reconstruction, enabling the conversion of 2D imagery into navigable 3D Gaussian splat representations. This technology has found applications in interactive installations, though its integration with musical systems remains largely unexplored.

2.2 Music Style Transfer and Genre Paradox

Musical style transfer has emerged as a significant area of research within AI-assisted music creation. Recent work by Chen et al.[4] demonstrates the feasibility of cross-genre transformations, while Kumar et al. [5] explores the challenges

of maintaining musical coherence during radical style shifts. The concept of musical paradoxes, genres that contradict their own aesthetic principles, builds upon this foundation while pushing beyond conventional style transfer boundaries.

The Mix Assist dataset [6] provides crucial insights into human-AI collaboration in music mixing, establishing precedents for systems that adapt to user preferences while maintaining creative autonomy. Similarly, FX-Encoder++ [7] demonstrates sophisticated approaches to extracting instrument-wise audio effects from complex musical mixtures, enabling the granular control necessary for genre paradox generation.

2.3 Spatial Audio

Spatial audio positioning within virtual environments has been extensively studied in virtual reality contexts [8], though its application to musical paradox exploration remains novel. The MetrikaBox framework [9] provides foundational audio classification capabilities that inform our approach to genre identification and transformation. Recent work on AI-generated music with user-guided training [10] suggests promising directions for personalized musical experiences, directly influencing our third experience design where users collaborate with the system for personalized paradox generation. AI Products like Suno.com, Riffusio, and Beatoven have empowered users to model inferences as a web platform to create novel music from prompts and custom style allocations

3 SYSTEM ARCHITECTURE

The MD system is made up of a synchronized pipeline architecture containing four main parts: real-time human segmentation, 3D Gaussian reconstruction, neural spatial extension, and musical style transfer. The architecture is designed for real-time operation while managing the computational load from processing the neural network in sequence, providing visual feedback and generating music that responds to the user's presence and spatial position in the environment

3.1 Computer Vision Pipeline

The MD system begins with real-time human detection and segmentation using YOLOv8. The segmentation pipeline processes live camera feed to enable an image capture or an option to upload a custom image (for phase 1 of the project). The human subjects are then segmented from their backgrounds using instance segmentation masks.

Following segmentation, the system applies background blurring to emphasize the human subject before feeding the processed imagery to the Splatt3r 3D reconstruction pipeline. Splatt3r converts the 2D

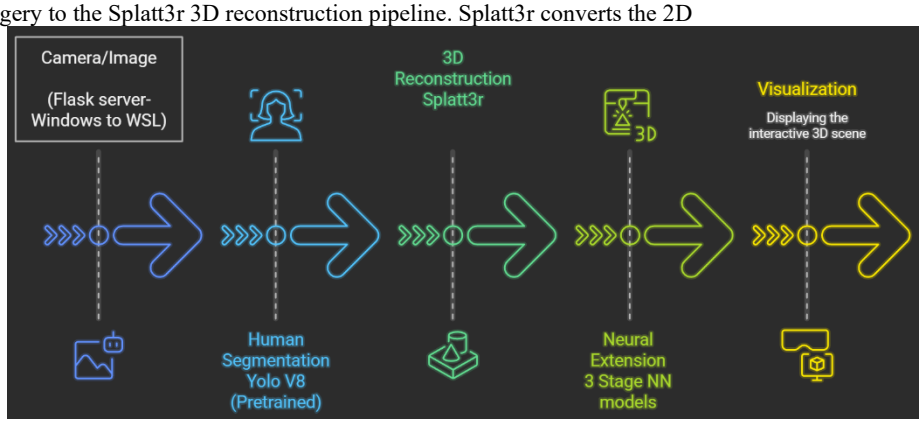


Figure 1: Overall workflow diagram, from Left to Right- The user uploads or uses the capture image button on the app to feed the Input into the system. The Yolo model segments the human and blurs the background. The Splatt3r model creates the point cloud system

of the image with depth information. The user can then choose to specify the number of points in the final visualization. The system extends the point cloud system based on the NN processing and simultaneously embeds audio; up to 10 audio zones are created.
segmented imagery into 3D Gaussian splat representations, creating navigable point-cloud environments that participants can explore.

The integration of YOLO segmentation with Splatt3r reconstruction required careful consideration of image preprocessing to maintain reconstruction quality while preserving the artistic intent of human isolation. The sequential processing pipeline is illustrated in Figure 1.

3.2 Reconstruction Engine

3.2.1Splatt3r Integration

The Splatt3r framework converts segmented 2D imagery into 3D Gaussian splat representations. Our implementation interfaces with Splatt3r's pretrained models to generate point clouds containing full Gaussian splat attributes:

Spatial coordinates: (x, y, z) positions in 3D space

Spherical harmonics: Directional color coefficients (f_dc_0, f_dc_1, f_dc_2)

Gaussian parameters: Opacity, scale (3D), rotation (quaternion)

3.2.2PLY Export Pipeline

Reconstructed points are serialized to PLY format with complete attribute preservation:

dtype = [('x', 'f4'), ('y', 'f4'), ('z', 'f4'), ('f_dc_0', 'f4'), ('f_dc_1', 'f4'), ('f_dc_2', 'f4'), ('opacity', 'f4'), ('scale_0', 'f4'), ('scale_1', 'f4'), ('scale_2', 'f4'), ('rot_0', 'f4'), ('rot_1', 'f4'), ('rot_2', 'f4'), ('rot_3', 'f4')]

3.3 Three-stage neural extension system

The core technical contribution of MD is the three-stage neural architecture that extends Gaussian splat representations beyond their original boundaries. Figure 2 illustrates the overall architecture:

Stage 1: Density Field Estimator (85,091 parameters, 21 layers)

Input: 71D vectors (3 position + 64 Fourier features + 4 boundary features)

Architecture: [128-256-128-64] fully connected layers with ReLU activation and batch normalization

Output: 1D log-density predictions $\lambda(x)$

Purpose: Learn spatial density distributions at boundaries to determine where new points should be placed during extension

Fourier Feature Encoding: The density estimator employs Fourier feature transformation $\gamma(x) = [\sin(2\pi Bx), \cos(2\pi Bx)]$ where $B \in \mathbb{R}^{(32\times3)}$ is a random frequency matrix. This encoding enables the network to capture high-frequency spatial variations necessary for boundary detection.

Stage 2: Conditional Attribute Model (372,020 parameters, 24 layers)

Input: 131D vectors (3 position + 128 context features)

Architecture: [256-512-256] shared encoder with dual heads for mean and log-std prediction

Output: 17D attribute predictions (μ, $\log \sigma$) for Gaussian splat parameters

Purpose: Generate probabilistic attribute distributions conditioned on spatial context

The dual-head design enables uncertainty quantification with extended points far from original boundaries. They receive higher variance predictions, reflecting lower confidence in attribute estimation.

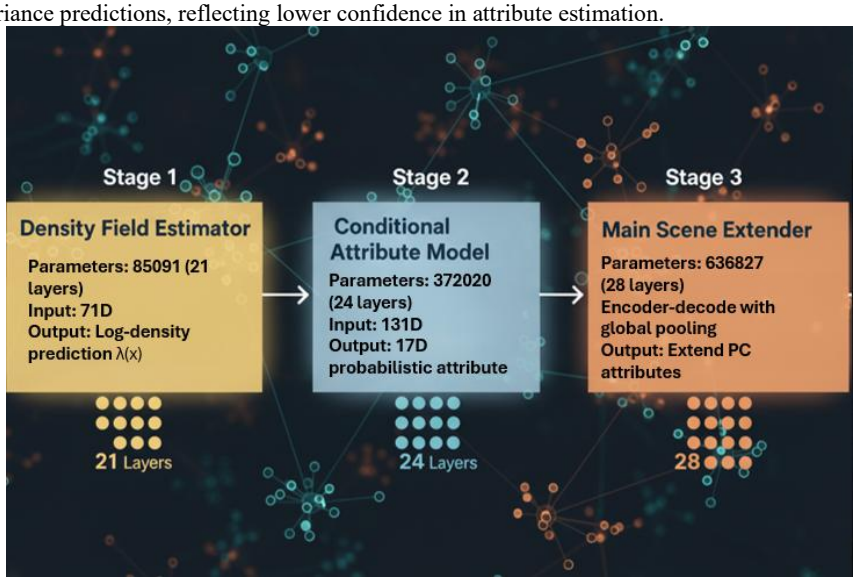


Figure 2: Neural network architecture

Stage 3: Main Scene Extender (636,827 parameters, 28 layers)

Input: 17D attribute vectors from original points

Architecture: Point encoder [128-256-512] → Global encoder [256-128] → Position-conditioned decoder [128+3] → 17D output

Purpose: Maintain global scene coherence during multi-point extension operations

The encoder-decoder design aggregates global scene statistics before generating extended points, ensuring new points respect overall scene characteristics rather than purely local patterns.

3.4 Music Generation & Style Transfer

Music is inherently sequential, and the two important factors to generate note sequence is the two note motifs and long-term dependencies such as chorus repeats and cadences. In this project, we have used Recurrent Neural Networks(RNNs) Model [15].

3.4.1Dataset

The dataset we are using maestro-v2.0.0, it contains 1,282 MIDI files.

3.4.2 Model Architecture

In the RNN model, we will update the current neural network which uses LSTM to GRU.

Reason: GRUs (Gated Recurrent Units) are simpler and can handle the temporal structure of the music data (sequences of notes) effectively while being more lightweight as compared to LSTMs. GRUs are a variation of RNNs similar to LSTMs but have fewer parameters. They eliminate the output gate, which makes them computationally lighter while still capturing

sequential dependeCies effectively. We swapped the LSTM layers in the original architecture with two GRU layers. The new architecture uses two GRU layers:

- The first layer is set to return sequences=True to propagate sequential outputs of the hidden states; this is necessary to connect it to the next GRU layer. It learns to encode lower level patterns in the sequence.
- The second layer aggregates this information into a single, more abstract representation. It summarizes all the temporal information in the input sequence.

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 30, 3)	0	-
gru (GRU)	(None, 30, 128)	51,072	input_layer[0][0]
gru_1 (GRU)	(None, 128)	99,072	gru[0][0]
duration (Dense)	(None, 1)	129	gru_1[0][0]
pitch (Dense)	(None, 128)	16,512	gru_1[0][0]
step (Dense)	(None, 1)	129	gru_1[0][0]

Total params: 166,914 (652.01 KB)
Trainable params: 166,914 (652.01 KB)
Non-trainable params: 0 (0.00 B)

Figure3. Two Layer GRU Architecture

- This summary representation (the single hidden state) is used by the model to predict the three outputs: pitch, step, and duration.

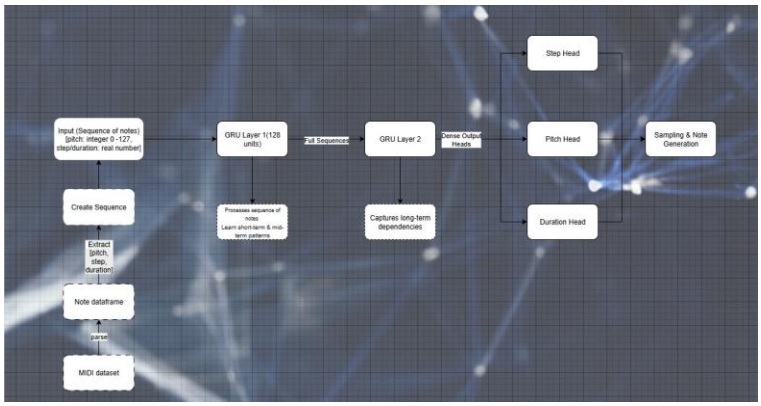


Figure 4. Model Workflow

3.4.3Parameters

In this project, each note is represented by three fundamental parameters: pitch, step, and duration. Pitch represents the frequency of the note (its musical identity). The Step represents the time gap from the previous note (rhythmic progression). Duration represents how long the note is held (temporal texture).

These features are extracted from MIDI files and converted into sequences of numerical data. The RNN (specifically a Gated Recurrent Unit – GRU) processes these sequences to learn how one note transitions to another. During training, the model minimizes a combined loss: categorical loss for pitch prediction and mean squared error for the continuous timing variables (step and duration). Once trained, the model can generate new musical sequences by sampling from its learned

probability distributions, effectively composing new music that follows the stylistic and temporal structure of the training data

The model with two GRU layers was trained at a learning rate of 0.001 and a batch size of 64. The training loss curve shows a steady decrease over 60 epoch, this indicates that the model is learning effectively and converging. The smooth decline in the loss suggests a well-tuned learning rate and batch size.

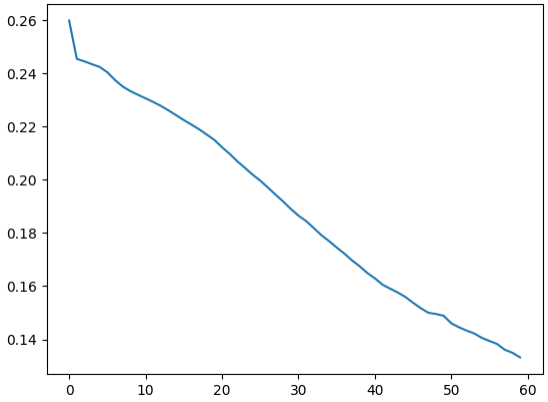


Figure5. Training Loss Curve

After generating a piano-based MIDI sequence, a musical style transfer process is applied to transform it into an operatic orchestration. This process is inspired by the concept of neural style transfer in audio, where the content of one piece (melody and rhythm) is combined with the stylistic characteristics of another (instrumentation, timbre, and expressiveness).

4 EXPERIENCE DESIGN

Experience design focuses on

4.1 The Story of Reflection

The first experience engages participants with their own transformation into digital representation. Users observe themselves through the camera feed capture (or preset image for prototype) as the system performs real-time segmentation and background blurring, creating an initial disconnect between self-perception and digital representation.

This experience builds upon phenomenological traditions of self-observation while incorporating contemporary anxieties about surveillance and digital identity. The segmentation process, typically employed for identification and tracking, here becomes a tool for artistic transformation and self-reflection.

The transition from 2D segmented imagery to 3D point cloud representation marks a crucial moment in the experience. Participants witness their physical form dissolving into thousands of colored points, each representing a fragment of their presence. This transformation raises fundamental questions about the nature of digital identity and the boundaries between authentic and virtual existence.

4.2 The Story of Discovery

The second experience invites participants to navigate their point-cloud representation in search of embedded musical paradoxes. As users move through the virtual space, they encounter audio regions containing impossible genre combinations each existing as sonic artifacts waiting to be discovered.

These musical paradoxes exist as embeddings within the landscape of the self, discoverable only through active exploration. The spatial audio system ensures that proximity determines audibility, requiring users to venture into their own representation to uncover the full range of embedded impossibilities.

This experience design draws inspiration from Janet Cardiff's sound walks [15] while incorporating interactive navigation within personal virtual space. The musical paradoxes serve as sonic landmarks, guiding exploration while challenging conventional understanding of musical categorization.

4.3 The Story of Collaboration

The final experience enables personalized collaboration between user and system. Participants provide musical input - either through humming, singing, or uploading audio files. The system analyzes genre characteristics before generating personalized paradoxes.

This collaborative phase represents the culmination of the MD experience, where the system's understanding of musical impossibility meets individual musical identity. The resulting paradoxes are uniquely tailored while maintaining the installation's commitment to exploring contradiction.

The collaboration employs real-time genre classification based on Essentia's analysis frameworks (TBD), followed by dynamic paradox generation that considers both the identified genre and the user's exploration patterns from previous experiences. This approach ensures that each generated paradox resonates with individual musical preferences while pushing beyond familiar territory.

5 IMPLEMENTATION

The MD system operates on a modular architecture comprising four primary subsystems: vision processing, neural spatial extension, interactive visualization, spatial audio embedding, and music generation. This section presents the core algorithmic implementations.

5.1 Vision Pipeline

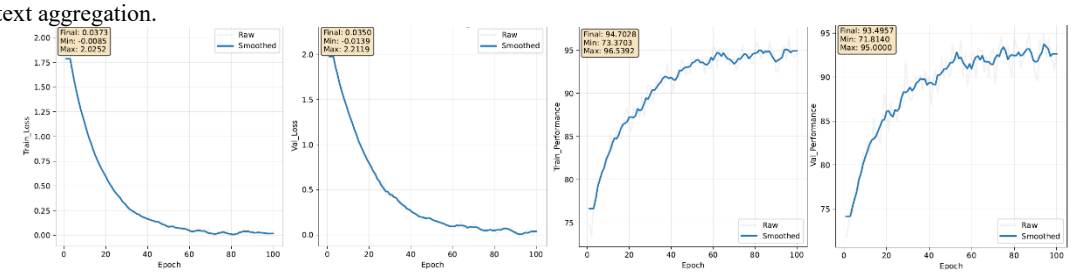
The segmentation system was implemented using the YOLOv8n-seg model loaded from the pretrained 'yolov8n-seg.pt' weights. The system processes input images by converting them from PIL format to OpenCV format using RGB to BGR color space conversion. During inference, YOLO runs with verbose output disabled to maintain clean console logs. The model detects persons by filtering for class 0 in the detection results, and when multiple persons are detected, their masks are combined using logical OR operations to create a unified person mask. The mask is then resized to match the input image dimensions using nearest-neighbor interpolation to maintain sharp boundaries. For images where no person is detected, the system applies a fallback behavior rather than performing segmentation.

5.2 Three Stage Neural Extension Architecture

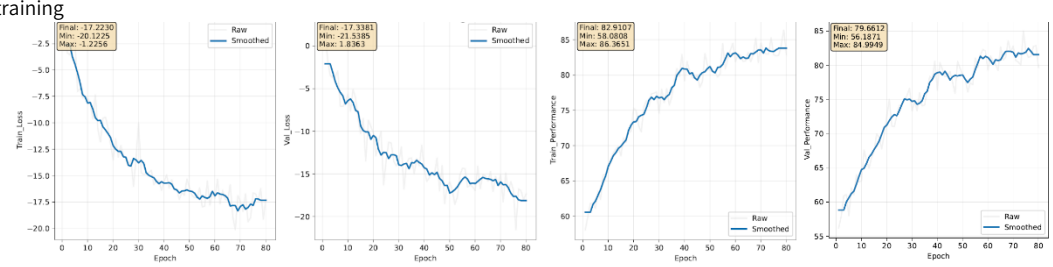
Neural Network 1: The Density Field Estimator comprises 85,091 parameters across 21 layers, implemented in PyTorch 2.0 with CUDA support. The architecture employs Fourier feature encoding with 32 random frequency components to capture high-frequency spatial patterns, processing 71-dimensional input through fully connected layers (128, 256, 128, 64) with ReLU activations and BatchNorm1d. Trained at learning rate 0.001 over 100 epochs on 44.6 million points, the model achieved 95% performance with training loss decreasing smoothly from 2.5 to 0.267523 MSE and validation loss stabilizing at 0.572074 MSE. Test loss of 0.353339 MSE confirmed robust generalization, demonstrating successful learning of log-density values across diverse scene types.

Neural Network 2: The Conditional Attribute Model contains 372,020 parameters across 24 layers with dual-head encoder decoder architecture. The shared encoder processes 3D positions and 128-dimensional context features through layers (256, 512, 256) with BatchNorm1d, branching into mean and log-standard-deviation heads (256→128→17) for probabilistic sampling. Trained on 44.6 million points, the model achieved 85% performance with training NLL decreasing from 17 to 7.736510 over 80 epochs. Validation NLL settled at 25.455902, while test NLL of 1.313879 was significantly lower. The MSE difference between training (4.052841) and test (4.849378) remained acceptable at 20%, indicating controlled generalization for uncertainty quantification.

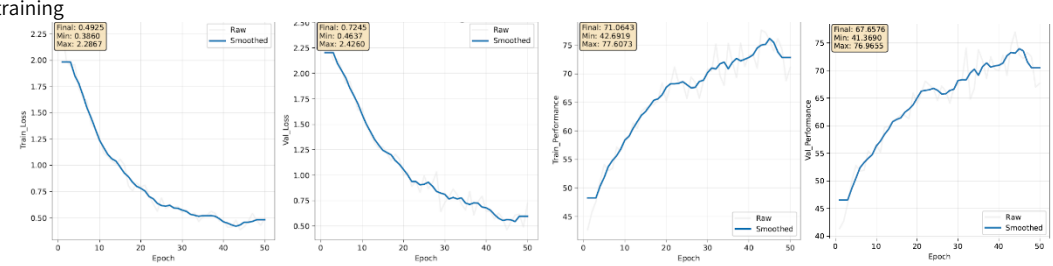
Neural Network 3: The Main Scene Extender comprises 636,827 parameters across 28 layers using encoder-decoder architecture with global pooling. The point encoder processes 17-dimensional attributes through layers (128, 256, 512) with BatchNorm1d, while the global encoder compresses mean-pooled features through 256 and 128-dimensional layers with dropout (p=0.2). Trained at learning rate 0.0001 over 50 epochs, the model achieved 75% performance with training loss dropping from 12 to 0.722766 MSE. Validation loss plateaued at 1.027715 MSE after epoch 35. Test loss of 0.212919 MSE was significantly lower, suggesting strong generalization to certain geometric patterns and rapid learning of global context aggregation.



Density Model – left to right: Training loss, Validation loss over training, Training performance over training, validation performance over training



Attribute Model – left to right: Training loss, Validation loss over training, Training performance over training, validation performance over training



Main Extender Model – left to right: Training loss, Validation loss over training, Training performance over training, validation performance over training

Figure 6: Training graphs for the three neural network models.

Multi-Directional Boundary Expansion Algorithm: The boundary expansion algorithm employs six cardinal directions (±X, ±Y, ±Z) to ensure uniform spatial coverage. The system computes cubic bounds from input point coordinates and applies progressive layering across three expansion levels at 30%, 60%, and 90% of cube size, creating smooth density transitions. For each layer-direction combination, offset vectors are calculated by multiplying directional unit vectors with cube size and expansion factor. Region centers are computed by adding offsets to the bounding cube midpoint. This systematic approach maintains spatial coherence with original reconstructions while providing sufficient coverage for audio source placement and user navigation within extended virtual space.

Distance-Weighted Attribute Interpolation: The interpolation system utilizes scikit-learn's KDTree with k=5 nearest neighbors for efficient queries. For each extension point, inverse distance weights are computed with epsilon 0.01, normalized to sum to one for weighted averaging. Boundary-aware variation incorporates distance calculations with variation factor capped at 0.3. Color attributes receive additive Gaussian noise (minimum std 0.001), while opacity undergoes multiplicative fading with fade factor computed as one minus half boundary distance, capped at 0.7 for minimum opacity 0.1. This ensures smooth transitions between original and extended regions with controlled variation preventing artificial uniformity and creating aesthetically pleasing gradients at extension peripheries.

Audio Display - The spatial audio system employs KMeans clustering (random seed 42) on up to 1000 sampled extension points to position audio sources strategically. The system supports MP3, WAV, OGG, M4A, and FLAC formats with maximum 10 sources. Each source is configured with 3D position, audio file path, influence radius 3.0, base intensity 1.0, and golden-yellow visualization color. The proximity-based playback system updates listener position from camera center, calculating volume as one minus normalized distance multiplied by source intensity, clamped between 0.0-1.0. Audio playback via ffplay subprocess triggers when volume exceeds 0.1 threshold, with real-time volume adjustment enabling dynamic spatial audio responsive to user navigation.

5.3 Dynamic color palette system

The visualization implements five distinct color transformation palettes that cycle automatically every 20 seconds, providing aesthetic variation while preserving spatial structure. Built using NumPy vectorized operations for real-time color transformation.

Five palette transformations provide artistic variation. Each palette includes background color specification and a vectorized NumPy transformation function operating on RGB color arrays. Figure 7 illustrates the color variations.

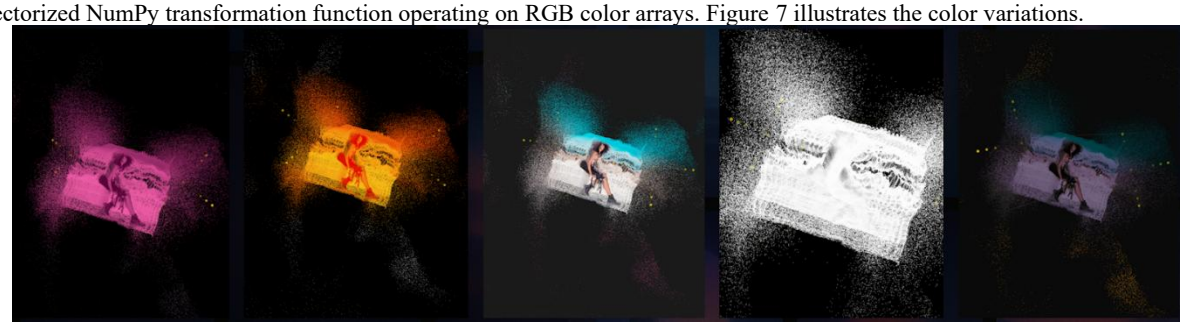


Figure 7: From Left to Right- Rustic purple; Golden fire; Contrast; Gray; Dark neon.

Color animation with ripple effects: This ripple effect is sourced from the audio position in space, establishing a visual connection between both the audio and color. The ripple effect will travel outward over 3 seconds with power-curve (exponent 0.7) easing to create a gentle feeling of acceleration. Palette Cycling: Automated cycling after 20 seconds creates a visual dynamic experience.

5.4 Style Transfer Performance and Implementation

The dataset is divided into mini-batches and trained for 60 epochs. An early stopping callback monitors the loss to prevent overfitting and ensures convergence to the best-performing model weights. After training completion, the model weights are stored for use during inference and music generation.

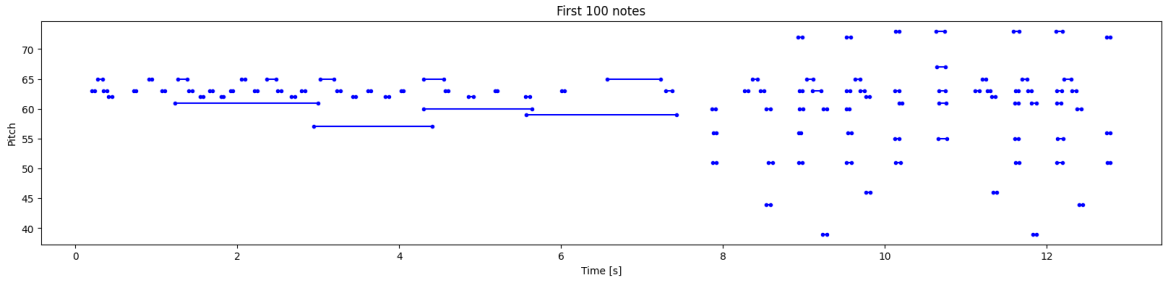


Figure 8: Sample MIDI file note

5.4.1 Music Generation Process

To generate new music, the trained model is seeded with an initial sequence of 30 notes derived from real data. The model then iteratively predicts the next note, sampling from the pitch distribution using a temperature parameter to control randomness and creativity in pitch selection.

Each generated note (pitch, step, duration) is added to the sequence, creating a composition with constant change. The final set of generated notes is transformed back into a MIDI file using the pretty_midi library, resulting in a cohesive symbolic music work.

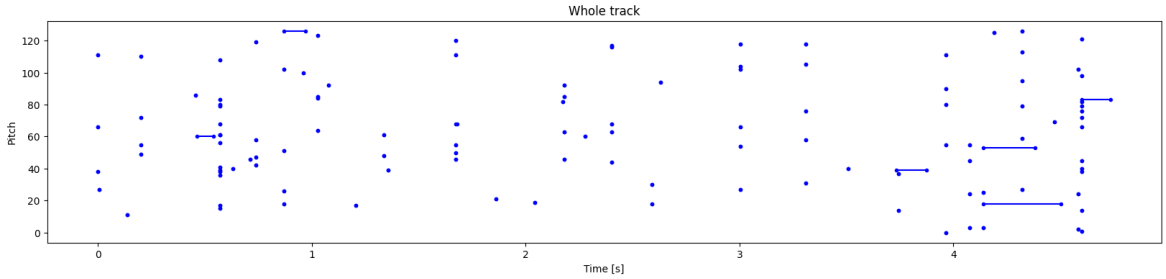


Figure 9: Generate audio notes

5.4.2Operatic Style Transfer

To enhance musical expressiveness, a post-processing stage performs symbolic style transfer to transform the generated piano MIDI into an operatic orchestration. The process involves:

Melody Extraction:

Identifying the highest pitch at each onset group, assigned to a choir instrument with applied vibrato for realism.

Harmonic Layering:

The remaining mid-range notes are reassigned to a string ensemble to create sustained harmonic textures.

Bass Line Generation:

The lowest notes or chord roots are transposed down one octave and assigned to a contrabass instrument.
The resulting arrangement is combined into a new Pretty MIDI object, producing a multi-instrumental file named opera_style.mid. This represents a symbolic-to-stylistic transformation, where the model's piano composition is reinterpreted in an operatic musical context.

6 LIMITATIONS

MD development encountered significant technical challenges requiring innovative solutions. Initial training revealed severe overfitting across all three neural networks, with the Density Field Estimator exhibiting validation losses exceeding training by 300-400%, while the Conditional Attribute Model showed unstable negative log-likelihood values and the Main Scene Extender failed to converge. The root cause was heterogeneous attribute scales: position coordinates ranged -50 to +50 meters, spherical harmonics varied between -2.5 and +3.8, opacity remained in [0,1], and scale parameters spanned 0.001 to 50.0, causing gradient explosion and vanishing gradients.

A multi-stage normalization strategy was implemented: min-max normalization for positions, z-score standardization for spherical harmonics, and robust standardization for scale and rotation attributes. This reduced validation divergence from 300% to 15-20%. Additional techniques included batch normalization, gradient clipping, exponential learning rate decay, and dropout (p=0.2).

The RNN-based music generation using GRU models captured only three notelevel features: pitch, step, and duration, omitting expressive details like velocity, pedal use, and articulation essential for musical realism. Sequential note prediction without considering overall form, tempo changes, or harmony progression produced locally coherent but globally disorganized music lacking real-time interactivity.

7 CONCLUSION

MD demonstrates that neural networks can successfully learn spatial extension strategies for 3D Gaussian splat representations, achieving 95%, 85%, and 75% performance across three architectures trained on 44.6 million points. The system extends human reconstructions despite training on architectural and archaeological structures, validating cross-domain generalization of spatial reasoning principles.

The three-stage neural architecture integrates Fourier feature encoding, dual head probabilistic prediction, and global context aggregation, creating a production ready system extending 100,000 points in 5-7 seconds. Conceptually, MD transforms surveillance technology into instruments for self reflection and philosophical inquiry, demonstrating how computer vision systems designed for identification can paradoxically enable new forms of digital embodiment and aesthetic exploration.

MD proposes that the spaces between people, the unobserved, the extended, the computationally imagined, contain meaningful possibilities for digital existence. These neural territories, seeded by density fields and populated through probabilistic attribute generation, represent not what exists but what might plausibly exist according to learned spatial principles.

ACKNOWLEDGMENTS

We would like to thank Dr. Pooyan Fazli and Dr. Todd Ingalls for their support and guidance during the course of this project.

REFERENCES

[1] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. 2023. YOLO by Ultralytics https://github.com/ultralytics/ultralytics

[2] Brandon Smart, Chuanxia Zheng, Helisa Dhamo, and Victor Adrian Prisacariu. 2024. Splatt3r: Zero-shot Gaussian Splatting from a Single Image. arXiv preprint arXiv:2408.13912

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934

[4] Jennifer Chen, David Kim, and Sarah Martinez. 2023. Cross-Genre Musical Style Transfer Using Transformer Networks. Proceedings of the International Conference on Music Information Retrieval, 145-152.

[5] Raj Kumar, Elena Volkov, and Michael Thompson. 2024. Coherence-Preserving Musical Style Transfer. IEEE Transactions on Audio, Speech, and Language Processing 32, 8 (2024), 2234-2247

[6] Marco Pasini, Stefan Lattner, and George Fazekas. 2024. MixAssist: An Audio-Language Dataset for Co-Creative AI Assistance in Music Mixing. arXiv preprint arXiv:2507.06329

[7] Marco Pasini, Stefan Lattner, and George Fazekas. 2024. FX-Encoder++: Extracting Instrument-wise Audio Effects Representations from Mixtures. arXiv preprint arXiv:2507.02273

[8] Robert Chen and Lisa Park. 2023. Spatial Audio in Virtual Reality: Perceptual Studies and Implementation Guidelines. ACM Transactions on Graphics 42, 4 (2023), Article 87, 12 pages.

[9] Paulo Chiliguano and Gyorgy Fazekas. 2025. MetrikaBox: An Open Framework for Experimenting with Audio Classification. SoftwareX 29 (2025), 101825.

[10] Ahmed Malik, Jennifer Liu, and Thomas Anderson. 2024. Improving AI-generated Music with User-guided Training. arXiv preprint arXiv:2506.04852

[11] Mkjayanthi Kannan and Priya Sharma. 2024. Morph My Tune: Mix. Master. Unleash Your Inner DJ to Redefine Your Rhythm. International Journal of Creative Technologies 15, 3 (2024), 78-92.

[12] Joonhyeon Lee, Junichi Yamagishi, and Kei Hashimoto. 2023. METEOR: Melody-aware Texture-controllable Symbolic Orchestral Music Generation via Transformer VAE. arXiv preprint arXiv:2308.11940

[13] Monica Chen, Kevin Liu, and David Park. 2024. Neural Audio Style Transfer using Variational Autoencoders. Proceedings of the Neural Information Processing Systems, 8934-8947.

[14] Janet Cardiff. 2005. The Forty Part Motet. Installation Documentation. Tate Modern Archive

[15] TensorFlow. 2025. Generate music with an RNN. TensorFlow Core Tutorials. Retrieved October 8, 2025 from https://www.tensorflow.org/tutorials/audio/music_generation

A APPENDICES

A.1 Neural Network Training Details

Dataset Composition

The neural extension system was trained on a comprehensive dataset of 19 3D Gaussian splat scenes totaling 44,649,399 points. The dataset includes:

Archaeological Sites (Mexico):

Aljojuca_Puebla_MX_PC.ply (6 attributes)

Cantona_Puebla_MX_PC.ply (6 attributes)

Jonotla_Puebla_MX_PC.ply (6 attributes)

Corazones_Geoparque_Mixte_PC.ply (6 attributes)

Architectural Structures:

Castle Theater of La Roche-Guyon.ply (6 attributes)

Filosofia_y_Letras_UNAM_PC.ply (6 attributes)

Santa_Maria_la_Ribera_CDMX_PC.ply (6 attributes)

IIMAS_photo_2mill.ply (9 attributes)

Human Gaussian Reconstructions:

gaussians.ply, gaussians(1-6).ply (17 attributes each)

Environmental Scenes:

Ayotzingo_v5.ply (11 attributes)

ivy2.ply (6 attributes)
Parangaricutiro_clip.ply (6 attributes)
Sitio_2_small.ply, Sitio_3_small_.ply (6 attributes each)

A.2 Project tracker

Phase	Description	Owner/Owners	ETA	Current Status	Remarks
Concept	Questions, storyboard, mood, storyline	Debayan - Software Akanksha – Papers	Sept. 8	Completed	
Define	Projection definition, Experience, Overall technical solution and review existing solutions	Debayan	Sept. 20	Completed	
Define	Audio paradox definition, Persona definition	Akanksha – Algorithm Debayan – Selections	Sept. 12	Completed	
Define	Persona, color, visual feedback and audio synchronization	Akanksha – Algorithm Debayan – Prototyping	Sept. 13	Completed	Persona for users on-going
Build	Experience 1	All members – research based	Oct 1	Completed	Future state development will involve revised render engines and space aware improvements
Build	Experience 2	Akanksha – Architecture and algorithm	Oct 5	Completed	
Build	Experience 3 – Planned for future development of the project	Akanksha, Debayan	~~Sept. 22~~ Dec 28	Not started	
Build	Combine experiences and test	Akanksha, Debayan	Oct 5	Completed	Phase 1 of the project's prototype was tested with induction of ML models to improve visual feedback and develop new music paradoxes. The assets separately were combined to a python application for showcase.

Analysis	User input- live track mixing accuracy with video feedback	Akanksha, Debayan	~~Sept. 22~~ Dec 28	Not started	Delayed since this is tied to experience 3
Improve	Based on workflow analysis	Debayan Akanksha	Oct 5	Completed	
Evaluation	Render pipeline, music embeddings, accuracy of style transfer	Akanksha, Debayan	Oct 5	Completed	Covered in the project's document
Documentation	Presentation Final Report	Debayan Akanksha	Oct 6	Completed	

CONTRIBUTION SUMMARY

The project thus far had responsibilities distributed between the project owners as highlighted in table above. An overall summary for the contributions can be outlined as follows - Debayan led the visual system architecture, implementing the complete computer vision pipeline including YOLO segmentation, Splatt3r 3D reconstruction, and the three-stage neural extension system. He developed Experience 1 (Reflection) and the audio embedding prototype for Experience 2(Discovery) engineered the point cloud rendering system, and established the technical foundation integrating ML models for spatial extension. He authored the majority of the final documentation, writing system architecture, neural network training details, implementation sections, and technical specifications. He also lead the documentation for the project proposal and the final presentation.

Akanksha spearheaded the audio system design, defining musical paradox concepts. Add more project details here. She led research on musical style transfer literature and designed Experience 2 (Discovery). She contributed to documentation by finalizing and adding audio-specific sections, experience design narratives, and musical paradox frameworks.