The Scratch Engine: A Dihedral Permutation Sequencer for Structural Control of Data Streams in Autonomous Computer Music Systems

Simon Kilshaw, 2025

Table of Contents

Portfolio Contents:

Title Album: Dihedral Killer Cutz https://kilshaw.duckdns.org/DIHEDRAL_KILLER_CUTZ/

Project development homepage. https://kilshaw.duckdns.org/THE_DATA_SELFIE

Scratch Engine Albums: https://kilshaw.duckdns.org/prosodic_development/indexc.html

Code Repository: https://github.com/kilshaw/The_Data_Selfie

1. Abstract

This auto ethnographical practice-based research interrogates the role of the digital luthier and music composer, in a post-human landscape. The accompanying audio compositions and code explore the intersection between anthropo-centric and cyborg-centric music making paradigms. The literature review sets out the aesthetic stance of the researcher, highlights key post-human and actor network theories theory and their application in computer music systems design, performance and composition tools. and establishes a morphological distributed agency framework that is tested through the creative outcomes presented here. This practice-based research presents *The*

Scratch Engine, a real-time computer musical instrument in which structural transformation is governed by the action of the dihedral group D_6. Rather than sequencing musical events or parameter values, the system sequences permutations of functional parameter roles, enabling continuous reconfiguration of instrumental structure while preserving material identity. Six numerical values form a fixed state vector that is treated as immutable material; all musical variation arises exclusively through permutation.

The instrument embeds the twelve symmetries of a regular hexagon—six rotations and six reflections—as its complete transformational vocabulary. These permutations act upon six persistent parameter containers corresponding to stable musical or signal-processing roles. Structural change is thus realised as reassignment of values to roles rather than alteration of the values themselves, foregrounding relational transformation over parametric variation.

The system operates autonomously once configured, generating new state vectors at fixed formal intervals and applying dihedral permutations at user-defined temporal resolutions. Permutation is enacted continuously in real time, producing fluid, uninterrupted melodic and rhythmic output characterised by coherence through conservation rather than repetition. The underlying group action is rendered through both a geometric hexagonal interface and an isomorphic matrix representation, enabling perceptual legibility, algebraic rigor, and direct computational translation within the Pure Data environment.

Conceptually, the work reconfigures principles of total serialism as a live, performative process. Serial organisation is preserved at the level of structural relations rather than pitch-class rows or fixed matrices, aligning the system with transformational

music theory while extending it into an autonomous, time-based musical instrument. By making group action audible and operational in real time, the Scratch Engine demonstrates how formal mathematical symmetry can function as a primary compositional agent within contemporary music systems.

2. Introduction

Contemporary computer music practice increasingly operates at the intersection of algorithmic autonomy, real-time systems, and posthuman performance paradigms. (Collins et al., 2003; Hayles, 1999; Roads, 2015). As musical instruments evolve from passive sound-producing tools into adaptive computational agents, the locus of compositional agency shifts away from the human performer alone and towards distributed systems in which structure emerges through algorithmic process, constraint, and machine temporality (Rowe, 1993; Roads, 1996; Agostini & Ghisi, 2013). This research situates itself within this evolving landscape by presenting *The Scratch Engine*, a dihedral permutation sequencer that reconceptualises musical control as the live transformation of structural roles rather than the sequencing of musical events.

The Scratch Engine is not designed as a responsive or improvisatory system in the traditional interactive sense. Instead, it belongs to a class of autonomous technological performance systems (ATPS) in which formal behaviour unfolds independently once initial conditions and constraints are defined (Rowe, 1993; Zicarelli, 2002). Performer interaction is deliberately minimal and strategic, limited to configuring temporal resolution, formal duration, and transformational pathways. After activation, the system generates, permutes,

and reconfigures its own internal structures in real time, producing continuous musical output without requiring moment-to-moment human intervention.

This design choice aligns the instrument with posthuman and cyborg-centric perspectives on musical agency, in which creativity is understood as emerging from the coupling of human intention, formal systems, and machine execution rather than from human gesture alone (Haraway, 1991; Hayles, 1999; Braidotti, 2013). The performer functions less as an expressive controller and more as a system designer, curator of constraints, or instigator of processes. Musical form is not enacted through direct manipulation, but through the activation of a formal ecology governed by symmetry, permutation, and temporal iteration.

By embedding the full action of the dihedral group D_6 directly into the control architecture of the instrument, this work positions mathematical structure not as an abstract compositional aid, but as an active, audible force. In doing so, it contributes to ongoing discourse in algorithmic composition, transformational music theory, and posthuman performance studies by demonstrating how formal systems can operate as autonomous musical agents while remaining perceptually legible and musically expressive (Lewin, 1987; Xenakis, 1971).

3 Literature review:

3.1 Creative and Musical Aesthetic Stance

Music is and has always been defined and bound by the compositional and performance tools (instruments) from which the music is born (Montague, 2017). This inseparable relationship between tools and musical output that all luthiers and instrument designers observe has transcended genre(Pinch et al, 2002), social standing(Blades 1992), geography(Théberge, 1997) and era, but evolves and is shaped by the emergent materials, mechanical affordances and technologies of the latter. Magnusson (2021) frames this through the concept of ethno-organology, arguing that instruments shape musical ideas, performance practices, and even genres, and that technological innovation has consistently driven musical evolution—from bone flutes to the pianoforte, from amplification systems to contemporary digital and tangible musical interfaces, where programming paradigms not only provide new affordances for musicians, composers and audiences, but particularly for digital luthier (the designers and programmers). From the earliest documented pipe instruments referenced by Homer (c 800 BC), through the radical post-industrial revolution sonic experiments of the Italian Futurists in the early 20th century (Marinetti et al, 1909), to the digital age of contemporary computer music where the aesthetics of music production are increasingly shaped by software environments, plugins, and algorithmic composition, the evolution of musical aesthetics has consistently mirrored the development of new technologies.

The Italian Futurists, particularly Luigi Russolo and Filippo Tommaso Marinetti, offer a compelling historical precedent for this research. Their invention of the Intonarumori mechanical noise-generating devices, marked a deliberate rupture with traditional musical

and aesthetic values and audience acceptances (Bucknell, 2020) The Futurists provocatively called the Intonarumori instrument, “il scoppiatore” (Serafin, 2012) This is a term usually used in Italian when referring to a bomb “going off” or war “breaking out”. It is to be seen as deliberately provocative and aesthetically challenging. Italian Futurists, and specifically their experiments in musical instrument design and performance instruction, benchmarks a useful reference for the start of this research project, not least because it demarcates a time in history where, empowered by the industrial revolution, electricity began to expand the timbral possibilities with these earliest of electronic musical instruments. From the 1913 publication of Luigi Russolo’s “Art of Noise” manifesto, the Futurists gave birth to an exuberant acceleration of audio-focused aesthetic design, production, system-based compositional protocols. This moment, which Donin and Stiegler describe as “the mechanical turn of musical sensitivity” (Steigler, 1998), serves as a conceptual and historical anchor for this research. It represents a pivotal shift in the ontology of music-making from a historical anthropocentric model to a new materialism(7a), thus entangling the agency of humans, machines, animals and environments. In this way it can be recognised as a precursor to a posthuman approach to music and sound making, proto-posthuman, where the machine becomes not merely a tool but a co-creator agent of musical production. The Futurists’ embracing of mechanical and later electronic instruments foreshadowed the emergence of contemporary posthuman digital lutherie, where composers and instrument designers engage with software, sensors, and algorithms to craft new sonic vocabularies. The creative practice in this research is situated within this lineage, yet it is consciously distanced from the political ideologies historically associated with early Futurism. Instead, it draws inspiration from the movement’s radical aesthetic ambitions and its commitment to technological experimentation. Like Russolo’s scoppiatore, the work presented in this

practice is driven by a similar aesthetic desire to rupture conventional sound worlds, by adopting a neo futurist approach to my own compositions and music making, through the exploration the expressive potential of emergent technologies.

The development of electronic instruments in the 20th century from Leon Theremin's early theremin (1909) and the introduction of Schaeffer's theorisation of the sound object (1948), to Chowning's complex spectra synthesis techniques (1973), further expanded the timbral and structural possibilities of music, performance, performance technique and the ability to record and capture sound. For example, these innovations enabled composers Cage (1939), Tenney (1961), and Varèse (1929) to move beyond the constraints of hither-to traditional instrumentation. Varèse's Ionisation (1929–31), for instance, anticipates an electronic sound world through its use of percussion and non-pitched metallic sound sources, signalling a shift toward a more abstract and technologically mediated sonic language, through the design and process of new sound making phenomena. In this context, the role of the composer increasingly resembles that of a digital luthier—a hybrid practitioner who designs, programs, and performs with bespoke computer music systems. The creative process becomes both technically systematic and aesthetically exploratory, driven by a desire to generate novel sonic materials and to engage with the expressive affordances of contemporary media. The compositions that accompany this research are thus framed by a neo-Futurist aesthetic, one that embraces speed, intensity, and technological fluency, while remaining critically aware of its historical and cultural implications.

3.2 Posthuman Agency: Anthropocentric and Cyborg-centric models in music creation.

The rise of interactive and algorithmic music systems has catalysed a shift away from anthropocentric models of musical creation, prompting a rethinking of authorship, agency, and the role of process and systems in the composition and production of computer music. Music Machine Learning Generative Adversarial Networks (Simon, Huang 2019) and systems that are Artificially Intelligent (Nathanielle 2025), demonstrate how algorithms can not only participate in, but can fully assume the creative act. Collins and d'Escriván (2007) explore these tensions within the wider field of electronic and computer music, noting that algorithmic systems function as both compositional partners and instruments, depending on the extent of human control and intervention. They argue that such collaborations necessitate a reframing of the authorial role as procedural and collaborative rather than expressive and centralised. These developments align strongly with post-humanist theory, which critiques the privileging of human subjectivity and instead emphasises distributed agency across human and non-human actors.

On how we became posthuman, Hayles (1992) argues that the traditional conception of the human as a discrete, autonomous subject is increasingly over ridden by a model in which the human is understood as an information-processing system. The human is a collaborator within a broader network of cognitive and computational agents. Hayles' reconfiguring of creative agency has had profound implications for computer music creation, particularly in the context of algorithmic composition, AI-assisted creativity, and interactive systems. Hayles endorses Actor Network Theory where the act of composing music is no longer the exclusive domain of a singular human intellect, the sole genius originator, but becomes a distributed process, shared across human and machine actors where creator

becomes curator, from author to orchestrator of and participator in generative processes. In Actor Network Theory there is a symmetrical credit ascribed to all actors. (Latour et al, 1987)

Haraway's Cyborg Manifesto (1991) proposes the figure of the cyborg as a metaphor for hybrid identities that surpass and exceed binary oppositions such as human and machine. The cyborg model encourages composers to embrace hybridity—not only in terms of media and tools but also in the blending of human intuition with machine-generated processes. In a musical context, this is manifest in George Lewis's Voyager, (1985-1987) where the computer acts as an autonomous improviser. The system is engaged in real-time decision-making, responding to human input not as a reactive tool but as a co-creative partner. Lewis asserts that "the computer system is not an instrument and therefore cannot be controlled by a performer. Rather, the system is a multi-instrumental player with its own instrument," (Bailey, 1993) leveraging its capacity to engage in real-time improvisation as an independent musical agent rather than a passive medium.

Karlheinz Essl's Realtime Composition Library RTC-lib (2022) and Lexicon Sonate (2020) MAX and Puredata Library which enables real-time algorithmic composition methodologies, is evidence that compositional intent can be encoded into algorithms, delegated to generative systems, or emerge through real-time interaction with autonomous software agents. These systems operate according to their own logics, constraints, and temporalities, and can often produce results that exceed or diverge from the composer's original expectations.

Holly Herndon's PROTO (2019) features an AI vocal agent named Spawn, trained on Herndon's voice and those of her collaborators. Rather than using AI as a tool for replication, Herndon treats Spawn as a co-performer, engaging in real-time vocal improvisation and

composition. By decoupling the voice from the individual, she repositions it as a computational artifact. The voice that emerges is neither fully human nor fully artificial—it is posthuman, existing in a liminal space between the two, resulting in a seemingly posthuman vocal identity. The voice is reimagined as a collaborative, computational, and semi-anthropocentric phenomenon. David Cope's intent for Experiments in Musical Intelligence (EMI) was initially conceived as a response to a compositional creative block, using it as a kind of provocateur to stimulate his own compositional thinking. (Garcia, 2015) In this complex interplay of agency, EMI was not intended to replace the composer, but to challenge, inspire, and interact with them. In the case of both Cope and Henderson, the machine is not simply executing instructions but actively shaping the musical output. The composer's role shifts from that of a sole creator to that of a system designer, curator, or collaborator, responsible for framing the conditions under which musical material is generated and interpreted.

Hayles's framework also invites a reconsideration of embodiment in music creation. If the composer is no longer a bounded individual but part of a cybernetic package, then musical expression becomes a product of interfacing bodies and systems—human, digital, and hybrid. This is particularly evident in the musical live coding works of "Algorave" founders Collins and Maclean (2014), networked media performances of Pamela-Z (2003), and sensor and haptic based live performances of Kirby (2017) , where the margins between performer, instrument, and environment are fluid and co-constitutive. Ultimately, Hayles's post-humanist lens encourages a move away from anthropocentric narratives of genius and authorship, toward a more ecological and relational understanding of creativity. In this way, music is not simply made by humans using machines, but emerges from the blend of human cognition, machine computation, and the affordances of technological systems.

Technological and computational and theoretical advancements in the latter half of last century, (Groupe Recherche Musicale, Paris, Elekronische studios Cologne and Bell USA) became charted as significant research centres for the composition of electronic and electroacoustic music, and by association software/ hardware designers, technicians and synthesis modellers and digital instrument researchers. Similarly, and more recently, current academic and research institutions can be seen in current creative software tool making, (Ircam suite, GrmTools , Beast-Tools, Integralive,, and the AJAXSTUDIO). What they all share in common is the construction and manipulations of mathematical relationships between expressive sonic parameters. Hamilton confirms that when designing an instrument, whether following a parallel design or maintaining two independent approaches, "the physical and logical separation of the input device from the sound production necessitates multiple ways of processing and mapping the information coming from the input device. Mapping becomes therefore an essential element in designing new instruments." (Jorda, 2002) Miller Puckette's Pure-Data [31] programming and mapping environment is a fundamental cornerstone of the creative work presented in this research, allowing for the acquisition, parsing, processing, through processing, function, manipulation and synthesis of any data representation, whilst at the same time keeping the artist acutely aware of the fully-duplex, simultaneous handshaking between the roles of researcher, software designer, composer, improviser, performer and mathematician. At a recent Tone.js Machine Learning & Music series webinar [32], I asked workshop leader Tero Parviainen which of these roles does he see himself and his practice as being, and whether he is primarily focused on one or a simultaneous agencies operating between these roles. Parviainen approach is user-centric affirming that he wanted the environments he designed to be intuitive as possible to the user, not necessarily the listener. His practice is driven and informed by the software

coding, it's logic, functionality, and interestingly, his desire to hide that coding process from the intended listener/ user. Grond and Hermann (2012) makes quite an opposite observation in that if there exists enough nuance in the process and means by which the information is to be sonified, the sonic output itself can serve as a pathway into a particular sonic aesthetic representation of the world, especially if the map (or data model) it is linked to is manifestly exposed to contextualise the sonification or audification. This also means multiple data audifications, using different learning, listening and processing models can sit together as different doorways into a plurality of aesthetic representations of the world, in complex synthesis patterns.

Development is of course the whole raison d'etre of the open-source movement and perhaps explains why most source code is packaged as libraries, where those connections are left to the artist to discover and uncover how musical that sonic experience will be. Parviainen encourages developers to "explore the full potential of the system". Conversely, producer and performer Deadmau5, on his electronic dance music performances, actively avoids the exploration of his system's real-time compositional potential, rejecting the instrument's virtuosity in favour of playback of precomposed tracks (Parkinson, Bell 2015). But to fulfil the exploration of the full potential of a computer music system, at least to my aesthetic desires, should require some attention to liveliness, its ability to exist and react in real-time, with or without the input of a human performer or interactor.

Essl's approach is more environment-centric. On Improvisation with computers (Essl,2002), his conceptual framework is the design of a system that allows a synthesis of compositional environments and performance paradigms, a hybrid workspace where the conventional demarcations between composition, performance, and instrument are

effectively hybridised. He emphasises the efficacy of a real-time interactive system that provides immediate auditory feedback, thereby enabling spontaneous and continuous sonic manipulation, drawing similarities with George Lewis's work. Whilst Essl and Lewis may have different opinions on the function of the instrument as a passive conduit for human expression, they both advocate for systems that dissolve the boundaries between composition and performance, enabling environments where improvisation and structure coexist fluidly.

Recent developments in computational music further complicate the distinction between anthropocentric authorship and distributed agency. Contemporary AI-based music systems, including deep learning and transformer-based architectures, demonstrate an increasing capacity to generate stylistically coherent musical material with minimal human intervention (Huang et al., 2021; Expert Systems with Applications, 2022). However, these systems remain fundamentally conditioned by the statistical properties and aesthetic biases embedded within their training datasets, which overwhelmingly reflect Western tonal and rhythmic conventions. As such, their outputs often reinscribe anthropocentric musical grammars, even as they appear autonomous.

Several scholars have argued that this form of algorithmic creativity constitutes a *weak* or *delegated* autonomy, in which agency is distributed but not structurally redefined (Browne, 2025). In these systems, the composer or system designer retains primary authorship by curating datasets, selecting models, and authorising outputs. The machine functions as an extrapolative agent rather than a formally constrained one. This distinction is significant when contrasted with systems in which autonomy arises not from probabilistic learning but from explicitly defined transformational grammars.

From a posthuman perspective, this raises critical questions concerning transparency, interpretability, and perceptual legibility. Recent work in structured generative music argues that systems grounded in formal constraint, rather than data-driven inference, offer a clearer redistribution of agency, as their internal operations remain intelligible and analytically traceable (Ni-Hahn et al., 2025). In such systems, autonomy is not emergent from opacity but is enacted through rule-based transformation, aligning more closely with post-serial and transformational compositional lineages.

Open-source machine learning frameworks for music generation, Google's Magenta (2016), Huann, Engell et al's Music Transformer models(2021), and Tegridy's Los Angeles Composer Library (2022), are built upon extensive corpora of human-generated musical data. These datasets, comprising thousands of hours of Western tonal music, encode deeply anthropocentric assumptions about musical structure, style, and expression. As such, the models trained on them inevitably reflect and re-evoke these human-centred musical grammars, even though they generate novel outputs. However, the creative potential of these systems lies not merely in their capacity to replicate existing styles, but in their ability to navigate latent musical spaces—regions of possibility that interpolate between known musical forms and speculative, emergent ones. In this sense, music machine learning systems can be seen as cyborg-centric collaborators, producing outputs that are simultaneously grounded in human musical tradition and suggestive of posthuman or hybrid aesthetics. Despite this generative capacity, the composer or system user remains the principal agent in the creative process. It is the human who curates, selects, refines, and ultimately authorises the musical material proposed by the machine.

This morphological dynamic positions the composer in the network, not as a passive recipient of machine output, but as an active interlocutor in a dialogic process of co-creation. The composer curator may well get lost in rejoicing in an algorithmic sublime or a scientific narcissism (Cole, 2020), but ultimately the consumption of the final audio creation returns to the anthropocentricity of the human ear. From this literature review, one persistent challenge in algorithmic approaches remains the issue of listenability. Whilst recent systems are capable of producing harmonically and stylistically coherent material (Pachet 2016) , the music they generate can often lack emotional depth, expressive nuance, and a convincing narrative arc and flow. Born's application of Actor Network Theory (2005) to music making notes that listenability emerges not from the algorithm alone, but from the broader network of actors, including humans, datasets, software tools, instruments, performance layers, and audiences, that together shape the musical outcome. Born shows that when these actors are weakly connected or absent (for example, when expressive performance is omitted or when audiences approach AI music with bias), outputs are perceived as sterile; conversely, when the network is richer and better aligned, algorithmic works become more engaging and listenable.

Models, particularly those based on statistical or machine learning approaches, operate effectively at a local level, producing plausible note sequences or textures but struggle with maintaining large-scale musical form (Huang, 2018, Brion et al, 2019). in their survey of deep learning techniques for music generation show that deep learning outputs in melodic contours and stylistic transfers can feel either directionless or overly formulaic. Moreover, the paper explicitly flags "lack of performance expressivity" and "difficulty in long-term structure" as critical gaps in deep-learning music systems. For example, systems like AIVA (46)and Los Angeles Composer(46a) demonstrate remarkable stylistic imitation, yet

their outputs are often critiqued for lacking the dynamic contour and expressive phrasing characteristic of human compositions (Colton et al 2020, Dhariwal et al 2019.) The issue is compounded by the absence of performance nuance in many algorithmic software systems. While note-level data can be generated convincingly, aspects such as timing flexibility, articulation, and expressive phrasing are frequently underdeveloped, unless a human-in-the-loop integration is applied. Yamaha's Dear Glenn project addresses this by incorporating performance modelling trained on recordings of Glenn Gould, enabling real-time musical interaction with expressive fidelity (Yamaha Corporation, 2019). Similarly, Google's Magenta project, particularly the Performance RNN and Music Transformer models, have advanced the ability to translate and represent temporal dynamics and long-range dependencies in piano performance data yet, even these more advanced systems seem to prioritise technical fluency over emotional or perceptual engagement, leading, at least to me, to listener fatigue due to either excessive complexity or lack of variation.

As algorithmic composition becomes increasingly central to my contemporary compositional practice, the question is no longer simply whether the designed system can produce musical material that is useful to me as a composer, but bears a more nuanced requirement of whether it can sustain musical interest, display virtuosity and artistic intent. For this researcher, this presents both a challenge and an opportunity: to design systems that are not only generative, but autonomous fluid and musically compelling on an anthropocentric level. The creative work undertaken in this research explores this collaborative relationship between algorithmic systems and human authorship and the challenges of creating fluidity in composition and the tension of owning an aesthetic control. The resultant musical outputs of this practice resonate with Schlomvic's (2024) challenge of the notion of the single artist voice, resulting, for me at least, in a liberating expansion of my

stylistic approaches in a multiplicity of musical and aesthetic compositional voices. These systems are not merely instruments but co-creative hybrid and non-human agents, capable of proposing fluid musical ideas that challenge me, the composer, to respond, adapt, apply and reimagine their own creative role.

3.3 Autonomous and Formal Systems in Algorithmic and Post-Serial Practice

While machine-learning-based music systems dominate much contemporary discourse, an alternative lineage of algorithmic composition prioritises formal autonomy through mathematical structure rather than statistical inference. This approach extends from post-war serialism and transformational theory into contemporary computational practice, where symmetry, permutation, and group-theoretic operations function as generative grammars rather than analytical abstractions.

Recent research has revisited the application of dihedral and cyclic symmetry groups within computational music systems, demonstrating how formal constraint can generate musically coherent variation without reliance on stylistic imitation (Chen, 2024; Luo, 2024). These systems embed transformational operations directly into the generative architecture, allowing musical structure to unfold through closed sets of rotations, reflections, and permutations. Importantly, such approaches align with Lewin's emphasis on transformations as primary musical objects, shifting focus away from the musical surface toward the relations governing change. This formalist trajectory resonates strongly with post-serial aesthetics, particularly total serialism's extension of ordered relations beyond pitch to encompass rhythm, articulation, and dynamics. However, whereas historical total serialist practices often relied on pre-compositional planning or score-based realisation,

contemporary computational systems enable these transformations to operate continuously and in real time. As Browne (2025) argues, autonomy in such systems emerges not from complexity alone but from the disciplined application of constraint, enabling perceptually intelligible musical behaviour without continuous human intervention. Within this context, the Scratch Engine aligns with a growing body of research advocating *structural autonomy* over adaptive intelligence. Its use of dihedral symmetry as a control grammar positions it alongside contemporary formal systems while distinguishing it from AI-driven generative models. Rather than learning musical style, the system navigates a finite transformational space, producing fluid musical output through the traversal of mathematically defined relationships. This approach frames transformation itself as an audible musical phenomenon, reactivating serialist principles within a live, performative computational framework.

4. Conceptual Framework: Structural Control, ATPS, and Posthuman Agency

The Scratch Engine departs from traditional sequencing paradigms, which typically operate by generating discrete musical events within temporal grids (Roads, 1996; Taube, 2009). In such paradigms, musical meaning emerges primarily through the temporal succession of notes, gestures, or control events, with higher-level structure inferred retrospectively. By contrast, the Scratch Engine adopts a fundamentally different ontological stance: it prioritises the transformation of structural relationships over the sequencing of individual events. Instead, it sequences transformations of control structure: six numerical values are assigned to six persistent parameter containers, corresponding to functional roles such as pitch, note length, harmonicity, pitch multiplier,

resonant filtering, and distortion. Once generated, these values remain invariant, and all musical change arises exclusively through permutation.

In operational mode, the system autonomously generates new vectors at fixed formal intervals (e.g., every sixteen bars) and traverses permutation space at user-defined resolutions (bar, half-bar, quarter-bar). This autonomy distinguishes the Scratch Engine from interactive systems predicated on continuous performer feedback (Rowe, 1993), embedding agency within the system's architecture rather than in moment-to-moment control. Musical meaning arises from structural traversal and transformation, not from reactive gesture. From a posthuman perspective, the system exemplifies a cyborgic coupling of human intention and machine autonomy (Haraway, 1991; Hayles, 1999; Braidotti, 2013). The performer configures constraints and selects temporal parameters, but the system itself executes transformations that are perceptually legible, formally coherent, and musically consequential. Structural continuity, symmetry, and relational invariance become central musical phenomena, experienced through autonomous, algorithmically driven motion rather than direct human manipulation.

4.1 Bridging Lewinian and Posthuman Theory

Lewin's transformational framework conceptualises musical structure as the relational action of groups upon musical objects, privileging continuity and relational meaning over static material (Lewin, 1987). Lewin extended a more parametric application of Schoenberg's Serialist technique, namely Group Theory Musical Transformation (GTMT) that extended the techniques to model musical transformative relationships. This relational ontology finds a theoretical complement in posthuman perspectives, which de-center the

human subject and distribute agency across networks of human and non-human actors (Haraway, 1991; Hayles, 1999; Braidotti, 2013). The Scratch Engine operationalises this convergence: dihedral permutations act autonomously upon numerical vectors, producing structural transformations that are perceptually intelligible and musically expressive, yet exceed direct human control. In doing so, the instrument embodies a cyborgic realisation of Lewinian transformational space, where musical agency emerges through the interaction of formal group action, algorithmic execution, and temporal unfolding.

4.2 Structural Transformation over Event Sequencing

Traditional sequencing paradigms in electronic and computer music prioritise the temporal ordering of discrete musical events: pitches, rhythms, dynamics, or parameter changes. Even in algorithmic systems, this frequently manifests as the generation or selection of events within a predefined temporal grid (Roads, 1996; Taube, 2009). The Scratch Engine departs from this paradigm by operating at a deeper structural level. Rather than sequencing events, it sequences transformations of control structure.

At any given moment, the instrument is defined by a fixed six-element numerical vector. These values are not treated as musical materials in themselves, but as abstract data capable of acquiring meaning only through their assignment to functional roles. Crucially, once generated, this vector remains invariant for a fixed formal duration. No stochastic processes, interpolations, or gradual parameter changes act upon the values themselves. All musical change arises exclusively through permutation. This approach evidences structural reassignment rather than parametric variation, aligning the system with compositional approaches that privilege invariance, constraint, and formal coherence over surface-level

change (Lewin, 1987; Morris, 1998). Musical motion is produced not by changing what the material is, but by changing what it does.

4.3 Autonomous Technological Performance Systems (ATPS)

The Scratch Engine functions as an autonomous technological performance system in which structural behaviour unfolds independently of continuous human input. In performance mode, the system generates a new six-element vector at fixed formal intervals (e.g., every sixteen bars) and subjects this vector to dihedral permutation at user-defined temporal resolutions (bar, half-bar, quarter-bar, etc.). The performer does not select individual permutations in real time; instead, the system traverses permutation space algorithmically, cycling through the twelve symmetries of the dihedral group. This autonomy reflects Rowe's distinction between interactive systems that respond to performer input and systems that assert their own internal musical logic over time (Rowe, 1993). Agency is embedded in the design of the system itself: in the choice of group structure, the restriction to a closed transformational vocabulary, and the temporal constraints governing permutation. Once activated, the system behaves consistently, predictably, and rigorously, producing musical output that is coherent precisely because it is constrained (Zicarelli, 2002). Such behaviour aligns with ATPS frameworks in which musical meaning arises from the interaction between formal systems and time, rather than from reactive performer–system dialogue. The Scratch Engine does not "listen" to the performer or adapt

expressively; instead, it asserts its own internal logic, inviting the performer and listener to engage with the unfolding of stylistic structure as an autonomous process.

4.4 Posthuman and Cyborg-Centric Perspectives

From a posthuman perspective, the Scratch Engine can be understood as a cyborg instrument in which human and machine agency are inseparably intertwined. The composer does not directly articulate musical gestures; rather, they configure a formal system whose behaviour exceeds immediate human control. Musical authorship is distributed across mathematical structures, algorithmic processes, temporal frameworks, and the act of listening itself (Haraway, 1991; Hayles, 1999). This conception resonates with posthuman theories that challenge anthropocentric models of creativity by recognising non-human actors—algorithms, formal systems, machines as active participants in artistic production (Braidotti, 2013). The instrument does not merely execute human intention; it produces outcomes that are shaped by its own internal constraints and affordances. The resulting musical phrases and lead lines are not fully predictable, yet never truly arbitrary. By rendering permutation audible as motion, and symmetry perceptible as musical continuity, the Scratch Engine reframes formalism as an embodied, temporal experience. Serial organisation is encountered not as an abstract matrix or score-based procedure, but as a continuous sonic process in which structure is perceived through transformation and conservation (Babbitt, 1965; Lewin, 1987).

4.5 Transformational Theory and Posthuman Agency: A Synthesis

Lewin's conception of musical structure as the action of groups upon musical objects provides a crucial theoretical bridge between formalist music theory and posthuman models

of creative agency (Lewin, 1987). In Lewin's framework, musical meaning arises not from static entities, but from the relationships and transformations that connect them. This relational ontology resonates strongly with posthuman theory, which similarly decentralises the human subject and distributes agency across networks of human and non-human actors (Haraway, 1991; Hayles, 1999; Braidotti, 2013). The Scratch Engine operationalises this convergence by rendering group action not merely as an analytical abstraction, but as a real-time, autonomous process that unfolds independently of continuous human intervention. Transformations are enacted by the system itself, yet remain perceptually intelligible and musically consequential. In this sense, the instrument can be understood as a cyborg realisation of Lewinian transformational space: a performable environment in which musical structure emerges through the interaction of formal group action, algorithmic execution, and embodied listening. Musical agency is thus relocated from the composer or performer alone to the dynamic field of relations enacted by the system, affirming both Lewin's relational theory of musical meaning and posthuman accounts of distributed creativity.

5. Parameter Containers and Instrument State

The instrument defines six persistent parameter containers, represented geometrically as the vertices of a regular hexagon. Each container corresponds to a stable control role within a synthesis or processing architecture, encompassing pitch, note length or rhythmic interval, harmonicity or spectral density, pitch multiplier or register, resonant filtering, and distortion or nonlinear processing. These roles remain spatially and conceptually fixed throughout performance, while the numerical value occupying each role changes over time.

A single interaction generates a six-element numerical vector, for example [54, 4, 32, 6, 40, 121]. Once generated, the values remain immutable; no stochastic or interpolative processes modify them. All subsequent musical change arises exclusively through permutation. This design prioritises structural transformation over parametric noise, privileging constraint, invariance, and formal coherence.

6. The Dihedral Group D_6 as Control Grammar

Structural transformation within the system is governed exclusively by the dihedral group D_6, the group of symmetries of a regular hexagon. D_6 consists of twelve elements, six rotations and six reflections, each realised as a permutation acting on the six-element ordered set. These permutations form the sole transformational vocabulary of the instrument. Rotational permutations enact cyclic reassignment of parameter roles, producing gradual and perceptually continuous shifts, while reflections introduce inversional symmetries that often yield more contrastive reconfigurations. The finite and closed nature of D_6 ensures that all transformations are known, repeatable, and structurally related, producing formal unity even under rapid interaction.

7. Interface Representations: Hexagon and Matrix

The Scratch Engine employs two isomorphic interface representations: a geometric hexagonal layout and a two-column, three-row matrix. In the hexagonal interface, the six parameter containers are arranged spatially and labelled with fixed semantic descriptors corresponding to their functional roles. The numerical values contained within these positions circulate under the action of the dihedral group D_6, enabling transformations to be perceived as structural reassignment rather than modification of material. The matrix

representation re-expresses the same group actions in a computationally tractable format. Rotational symmetry corresponds to cyclic row or column shifts, while reflectional symmetry manifests as matrix inversion operations. Each of the twelve dihedral symmetries maps directly onto a unique matrix state, preserving adjacency relationships and ordering constraints. This formal equivalence ensures that the matrix does not introduce an additional organisational layer but functions as an alternative formalism for representing the same transformational process. Importantly, the matrix serves as an intermediary between the geometric interface and the Pure Data (Pd) environment, facilitating the parsing, routing, and application of permutations in real time across diverse synthesis and signal-processing domains.

8. Methods: Computational and Performative Operation

From a transformational perspective, the Scratch Engine aligns with Lewin's conception of musical structure as the action of groups upon musical objects (Lewin, 1987). System states are represented as ordered lists of six numerical values, while musical meaning emerges from the continuity and trajectory of transformations applied to these lists over time. Within Pure Data, rotations are implemented as list shifts, reflections as reversals combined with offsets, and matrix traversal as indexed access. This approach allows permutations to be applied uniformly across all parameter domains, producing continuous structural variation without modifying the underlying material values.

The system architecture is designed as an autonomous real-time musical environment governed entirely by the action of D_6. Temporal behaviour is determined by

user-defined parameters, including global tempo via Ableton Link, temporal resolution for permutation application (bar, half-bar, quarter-bar), and the formal duration of each state vector, typically sixteen bars. Once initialised, the system advances automatically through permutation space, producing continuous transitions between structural states. Performer interaction is therefore configurational rather than moment-to-moment; musical agency is exercised primarily through the selection of constraints and operational parameters, while autonomous execution determines the unfolding of structure.

9. Sequencing as Structural Traversal

In its operational mode, the Scratch Engine functions as a live permutation sequencer. Each activation advances the system to a new structural state while preserving the underlying numerical vector. A cumulative log records all permutations applied during performance, effectively producing a structural score that may be replayed, analysed, or mapped onto other musical dimensions, including orchestration and spatialisation. During performance, the system synchronises with networked tempo and generates a six-element vector at the beginning of a cycle, which persists for a fixed formal duration. At each subsequent temporal step, the system traverses the twelve dihedral symmetries, interpolating between matrix states to produce continuous sonic transitions. Continuity is maintained because energy and material are conserved, even as functional relationships are repeatedly reconfigured. This approach operationalises serial principles as dynamic structural traversal rather than static organisation, situating the system within a performative, time-based paradigm.

10. Relation to Serialist and Algorithmic Practice

Historically, permutation has been central to serial and post-serial composition, where ordering operations were applied to pitch-class sets, rhythms, and other musical parameters (Babbitt, 1965; Morris, 1998). In such works, including Boulez's *Structures I* (1952) and Stockhausen's *Klavierstück XI* (1956), serial procedures were applied pre-compositionally or offline, creating rigorously organised but fixed sequences. Similarly, early electronic and post-serialist compositions by Subotnick, Xenakis, and Tenney explored algorithmic transformation of parameters, often mediated through notation or fixed computational frameworks, limiting the extent to which structure could emerge in real time (Xenakis, 1971; Tenney, 1988).

The Scratch Engine diverges from this tradition by permuting not the values themselves, but the functional roles to which values are assigned. Each six-element state vector remains invariant, while dihedral transformations of the hexagonal interface reassign parameter roles continuously. This distinction reconceptualises serialism as a live, performative practice rather than a static organisational scheme, situating formal transformations directly within the flow of musical time. Rotational and reflectional permutations enact cyclic and inversional reassignments, producing perceptually coherent but non-repetitive structural shifts that maintain material identity. In this sense, the system operationalises transformational theory in real time, producing structured musical output that is both audibly intelligible and formally rigorous without reliance on offline score generation (Lewin, 1987). By embedding group-theoretic constraints directly into the control architecture, the Scratch Engine realises an autonomous temporal framework in which formal behavior unfolds independently of continuous performer input, exemplifying

the principles of autonomous technological performance systems (ATPS) (Rowe, 1993; Zicarelli, 2002). Musical agency is distributed across the system: the performer defines high-level constraints such as temporal resolution and formal duration, while structural transformations occur algorithmically, continuously, and audibly.

By connecting the lineage of serialist and post-serialist practice to posthuman performance perspectives, the Scratch Engine demonstrates that formal compositional operations can operate as emergent, time-based phenomena. Whereas historical serialist software and score-based practices emphasised human-mediated control, the present system embeds serialist transformations into an autonomous, algorithmically governed environment. The result is a hybridised mode of musical agency, in which dihedral symmetry, permutation, and algorithmic execution converge to produce continuous melodic and rhythmic motion, extending the conceptual and practical scope of contemporary serialist and post-serialist composition (Haraway, 1991; Braidotti, 2013).

11 Symmetry, Serialism, and Group-Theoretic Lineages in Musical Composition

The relationship between symmetry and music is historically deep-rooted and conceptually foundational. Since antiquity, music has been understood through symmetrical and proportional frameworks, most notably in Pythagorean philosophy, where musical intervals were first formalised as ratios of whole numbers (Pythagoras; Plato et al. [3.7]). Fundamental intervals such as the octave (2:1), perfect fifth (3:2), and perfect fourth (4:3) exemplify intrinsic numerical symmetry. These ratios are not merely mathematical abstractions, but constitute the perceptual and structural basis of harmonic consonance, embedding symmetry at the core of Western musical thought. In the twentieth century, the conceptualisation of musical symmetry shifted from harmonic proportion to

transformational structure. Music theorists and composers including Babbitt [3.8] and Lewin [3.9] demonstrated that core compositional operations—transposition, inversion, and retrograde—can be rigorously formalised as group actions. Subsequent work by Fiore et al. [3.10] explicitly identified these operations as elements of the dihedral group of order twelve, corresponding to the symmetries of the regular 12-gon. Within this framework, pitch-class space is treated not as a collection of static entities, but as a structured domain upon which symmetrical transformations act.

Arnold Schoenberg's twelve-tone serialism represents a pivotal moment in this historical trajectory. While Schoenberg's method abandons the harmonic symmetry of Pythagorean ratios, it nonetheless embodies a strict transformational symmetry at the level of pitch-class ordering. Inversion of the tone row produces a reflection about a central axis, corresponding to vertical symmetry in geometric terms, while retrograde presents a temporal reflection of the original ordering. Retrograde inversion combines these operations, yielding composite reflective and rotational symmetries. However, Schoenberg's serial technique applies such formal constraints almost exclusively to pitch-class ordering, leaving rhythmic structure, metre, and dynamics largely unconstrained and subject to compositional discretion.

In the post-war period, this limitation was addressed through the emergence of total serialism. Building upon Schoenberg's pitch-centric model, composers and theorists extended serial principles to encompass duration, articulation, metre, and dynamics. Lewin's transformational theory formalised this shift by foregrounding the operations themselves rather than the musical objects upon which they act, framing composition as a network of transformations rather than as a fixed array of ordered elements (Lewin [3.9]).

This reconceptualisation marked a decisive movement away from object-centred serialism towards relational and process-oriented musical structures. From a mathematical perspective, Schoenberg's serial technique can be understood as a direct application of permutation group theory, as formalised by Cauchy in the nineteenth century ([3.12]), and specifically as an instantiation of dihedral group relations acting upon the twelve-tone scale ([3.13]). Parallel developments in more tonally oriented contexts further demonstrate the breadth of group-theoretic thinking in composition. Hugo Riemann's application of group theory to harmonic function introduced symmetrical transformations within triads through parallel, relative, and leading-tone exchanges. Fiore's analysis of works by Bach, Pachelbel, Wagner, and Ives demonstrates that such transpositions and inversions correspond to the symmetries of the regular 12-gon when applied to triadic voice leading [3.10]. Cohn further observes that these symmetries naturally extend to single-line voice-leading contexts, albeit under parsimonious constraints that limit allowable transformations [3.14]. In all cases, symmetry operates within clearly defined rule-based systems.

Despite the extensive historical repertoire available for symmetrical analysis, Babbitt anticipated that the mathematical application of permutation and symmetry would continue to offer "enormous scope for investigation for future generations of composers" [3.8]. The present work responds directly to this proposition by relocating symmetrical operations from pitch-class content to the permutation ordering of abstract data strings. When the symmetric group $S_n$of an n-gon is associatively mapped to the Scratch Engine's compositional software, a characteristic improvisatory musical flow emerges. This flow exhibits formal behaviours analogous to exposition, development, transition, recapitulation, cadential articulation, and anacrusis, effectively generating miniature sonata-like forms at

each iteration of permutation. A stylistically coherent body of digital audio compositions has been produced and analytically examined to substantiate the claim that serialist principles—and, by extension, symmetrical operations—can be applied not to pitch classes themselves, but exclusively to the ordering and reordering of datasets. Through this approach, discrete and exclusive symmetrical permutation states are identified, traversed, and rendered audible. Musical fluency arises from the exposure of commutable relationships within the data, articulated through associative and re-associative mapping rather than through direct parametric control. In this sense, the Scratch Engine extends post-serial practice by abstracting serial symmetry away from musical surface parameters and embedding it instead within the structural logic of real-time computational systems.

12. Sonic Output and Serial Continuity

The sonic output of the Scratch Engine consists of continuous melodic and rhythmic phrases characterised by rapid reordering, micro-variation, and gestural articulation. Rather than developing material through thematic variation, coherence emerges through the recontextualisation of immutable numerical vectors under changing functional roles.

This approach aligns conceptually with total serialism (Boulez, 1954; Babbitt, 1965), while departing from historical implementations by realising serial organisation as an emergent, performative process. Listeners perceive continuity because transformations are experienced as motion through a constrained structural space, consistent with Lewin's theory of transformational perception (Lewin, 1987). Several extended recordings demonstrate the system's capacity to sustain uninterrupted lead-line trajectories over prolonged durations, preserving structural consistency without post-hoc editing.

13. Computational and Design Implications

From a systems perspective, the Scratch Engine exemplifies a rigorous separation of concerns between material generation, structural transformation, and presentation. Its architecture ensures robust behaviour under rapid interaction and supports extension to additional group actions or parameter cardinalities. Embedding explicit group-theoretic constraints directly into a real-time interface demonstrates how formal mathematical structures can coexist with performative immediacy. The system operationalises a form of computational total serialism in which unified structural transformations are distributed across heterogeneous musical parameters. This design supports Zicarelli's observation that productive musical systems derive richness from minimal but rigorous rules (Zicarelli, 2002), illustrating how constraint and autonomy can facilitate expressive complexity.

14 Practical Applications in Contemporary Electronic and Generative Music

The Scratch Engine operates as a bridge between formal, mathematically rigorous compositional models and contemporary electronic music workflows, particularly those oriented towards live coding, generative systems, and real-time performance. Its core functionality—permuting functional parameter roles rather than the values themselves—resonates with the practices of contemporary producers who manipulate modular synth environments, algorithmic patches, or DAW-based generative devices. Unlike conventional sequencers or automation tracks in DAWs, which often operate on pre-programmed, value-specific timelines, the Scratch Engine identifies structural transformations as the primary musical material.

In a typical usage scenario, the system is deployed within a 4/4 rhythmic framework, producing continuous melodic and rhythmic content in a scratching-style lead line, at any given tempo. These streams emerge from the autonomous traversal of twelve dihedral symmetries applied to six-parameter vectors, resulting in non-repetitive yet perceptually coherent material. This mode of operation aligns with contemporary generative workflows in which musical structures are defined algorithmically and allowed to unfold over time, a practice common in live coding sessions and modular electronic setups.

For electronic music producers and composers, the instrument functions both as a sound design tool and a compositional engine. Its real-time permutation of parameters enables complex, multi-dimensional modulation across pitch, rhythm, timbre, and spectral processing simultaneously, reducing the cognitive load on the performer while preserving expressive control. Coders benefit from the explicit mapping of group-theoretic actions onto software objects, allowing integration with environments such as Max/MSP, Pure Data, or SuperCollider, where list operations and array manipulations can be directly applied to synthesis and processing routines. By embedding formal structures into live performance, the Scratch Engine exemplifies a workflow in which autonomy, algorithmic rigor, and performative immediacy coexist. Producers and composers can thus explore emergent patterns and gestural improvisation within highly constrained yet dynamically evolving musical spaces, creating material that is simultaneously formally intelligible and stylistically relevant to contemporary electronic music idioms.

14.1 Integration with Contemporary Music Software and Workflows

The Scratch Engine complements and extends conventional electronic music software environments by embedding structural permutation directly into real-time control architectures. While DAWs such as Ableton Live and Bitwig provide event sequencing, automation, and clip-based modulation, they typically require manual programming of value changes and offer limited mechanisms for the dynamic reassignment of parameter roles. Similarly, modular environments like VCV Rack or hardware-based Eurorack systems allow for generative modulation and algorithmic patching, but structural transformations are generally hardwired or externally sequenced.

By contrast, the Scratch Engine situates dihedral permutation as an intrinsic, performable process, capable of traversing a mathematically closed space of twelve symmetrical transformations in real time. Its interface and matrix representations enable immediate reconfiguration of multiple parameters—pitch, rhythm, timbre, spectral processing—without necessitating pre-compositional intervention. In practice, the system can be integrated alongside Ableton Link, Max/MSP, or Pure Data workflows, providing continuous, autonomous structural variation that is compatible with clip-based timelines, generative patches, and live coding sessions.

For contemporary producers, coders, and electronic musicians, this workflow allows a hybrid mode of creation in which algorithmic rigor coexists with performative spontaneity. Musical output retains the perceptual coherence of formal serial structures while remaining flexible and responsive to user-defined temporal frameworks. Consequently, the Scratch Engine functions as both a compositional engine and a performative instrument, bridging

the gap between formalist theory and practical, studio- or stage-based electronic music production.

15. Adaptive and Associative Mapping

The Data Selfie Album [3.1] is the musical outcome of a software coded environment for digital scratching. The lead solo instrument in this album is the primary vehicle to explore and manifest melodic and rhythmic compositional fluency from data sets and their permutational containerisation and reordering. . Video and audio Assets that chart the development of the project can be accessed at the project's homepage [3.2] In this chapter the design and implementation of the Scratching Engine [3.3] in its vanilla form is explained. Initially conceived as a live performance instrument employing gestural control over audio parameters [3.4], the Scratch Engine was designed to accept input from a range of MIDI-enabled environments. One such implementation utilised camera-based motion tracking, in which six continuous data streams—corresponding to the three-dimensional spatial coordinates (X, Y, Z) of the performer's left and right hands—were captured and mapped directly onto the six primary control parameters of the instrument's synthesis engine.

As a performance interface, the system affords a high degree of immediacy and playability [3.5]. Melodic and rhythmic fluency can be achieved rapidly, even by inexperienced users, due to the intuitive correspondence between bodily gesture and sonic response. However, following extensive rehearsal and live performance use, it became evident that sustained reliance on direct gestural mapping imposed significant limitations on the exploration of the instrument's full sonic and structural potential. In particular, the

phenomenon of muscular memory, ubiquitous among all instrumental performers that I am acquainted with, was observed to constrain the diversity of gestural input and, by extension, the range of sonic outcomes produced by the system. While direct mapping paradigms theoretically permit an unrestricted mapping between gestural and sonic parameters, in practice they remain bounded by the performer's habitual motor behaviours, physical constraints, and gestural imagination. Certain regions of the system's potential sonic space thus remain systematically unexplored, either because specific gestures are physically unachievable or because they fall outside the performer's embodied repertoire. As a result, a substantial portion of the instrument's latent expressive capacity remains dormant. In response to these observations, the methodological framework adopted in this research deliberately moves away from direct mapping strategies, instead foregrounding adaptive and associative mapping as a foundational design principle.

The creative implications of adaptive mapping within the Scratch Engine emerged initially through an unplanned configuration error. As is typical within software-based instrument design, the performer–programmer retains full agency over the assignment of input data streams to audio parameters at any given moment. In an early implementation of the camera-based tracking system, parameters associated with pitch and rhythmic articulation were intuitively assigned to the performer's left hand, reflecting both physiological handedness and a perceived dominance in gestural expressivity.

During a subsequent testing session, the camera system was inadvertently left in a horizontally flipped configuration, analogous to the distinction between front- and rear-facing camera modes in mobile devices. This inversion resulted in the reversal of left–right hand tracking within the software environment. Although the performer's physical gestures

remained unchanged, the polarity of the X-axis data streams was inverted prior to their application within the audio engine. Consequently, the dominant gestural control was transferred to the opposite hand, producing a distinct yet internally coherent mode of interaction with the instrument.

From a systems perspective, this incident exemplifies associative mapping, in which a single data stream is permitted to influence multiple parameters or to be reassigned dynamically across different functional roles. Extending this principle, subsequent development focused on systematically reconfiguring the associations between the six input axes and the six audio parameters. Rather than privileging left- or right-handed dominance, the system was restructured to explore the complete set of permutable associations between input data and control roles. Through this process, a defined subset of dihedral symmetrical and quasi-symmetrical permutations was embedded directly into the mapping architecture of the Scratch Engine. These permutations were applied automatically at metrically defined temporal resolutions—at the level of beats or rhythmic subdivisions—resulting in a continuous reconfiguration of gestural-to-sonic relationships. The resulting musical output exhibits extended, uninterrupted melodic and rhythmic phrases characterised by structural coherence and internal variation, displaying formal properties analogous to serial techniques and fugal procedures. Importantly, despite the absence of explicit pitch-class manipulation, the emergent melodic trajectories produced by the system can be subjected to conventional single-voice leading analysis [3.6], demonstrating that the instrument's output conforms to established analytical frameworks. The *Data Selfie Album* recordings accompanying this study constitute creative artefacts generated through this

methodology, serving as empirical demonstrations of the Scratch Engine's application of permutation group theory to the real-time composition of extended solo lines.

16 The Scratch Engine

The Scratch Engine is a computer music programme designed in Puredata [3.15]. It can be downloaded on this project's open source Github repository [3.16]. The Scratch Engine is locked to a tempo in a DAW via Ableton Link [3.17]. It comprises of a three parallel oscillator wavetable with six parameters of audio transformation for each oscillator denoted and referred to henceforth as A-F:

A) Pitch Frequency (not distinct pitch classes)

B) Note-length onset variance (modulus subdivisions of a 4/4 beat)

C) Harmonicity

D) Pitch Ramp Multiplier

E) Distortion

F) Resonance Filter

In the case of the camera tracked left and right-hand example given above, the 3 axes of each hand are mapped directly as follows:

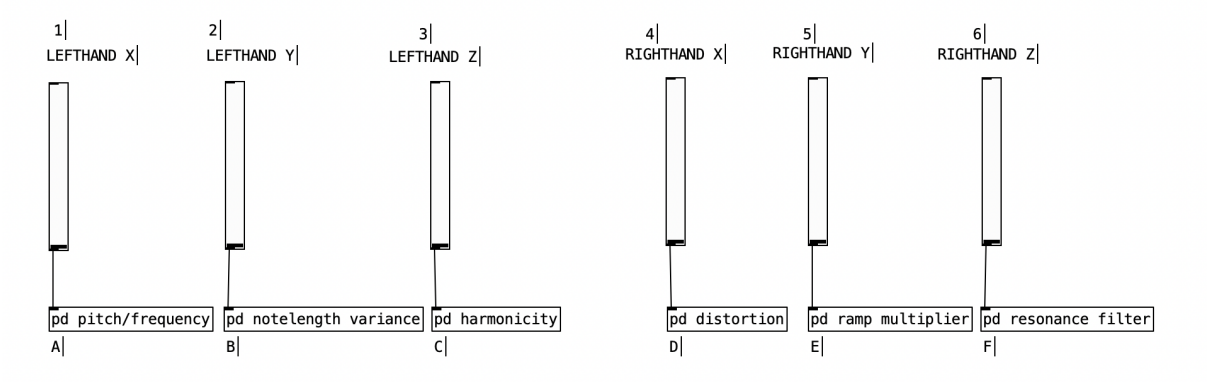

Fig 1 Left-handed dominancy

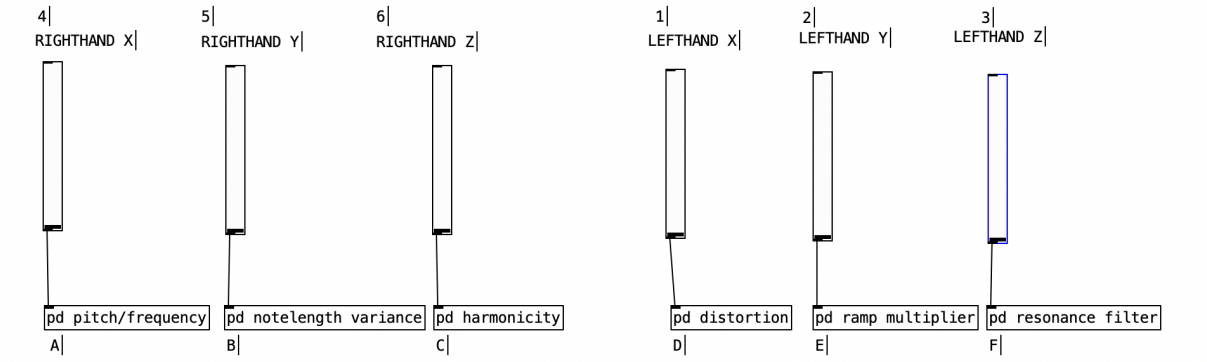

Fig 2 Right-handed dominancy

In figure 1 [1 2 3 4 5 6] = [A B C D E], whilst in figure 2 [4 5 6 1 2 3] = [A B C D E F] shows the permutation of the camera flipped order.

In the first instance, Let the data-string [1 2 3 4 5 6) be considered as six containers. (What data is held in each of those containers can, for now, be ignored.) In their re-ordering, there are factorial 6 (6!) = 720 unique ways of rearranging the order of the six containers,

where order does matter, and replacements are not allowed. These are all the possible permutations:

$$P(n,r)=?P(n,r)=?$$

$$P(n,r)=P(6,6)$$

$$=6!(6-6)!$$

$$= 720 \text{ permutations}$$

Permutations have many sub-groups. When considering the order of 6 strings, one can represents their order as an *n-gon*. In the examples in figure 1 and 2 the permutation can be viewed as a *6-gon*:

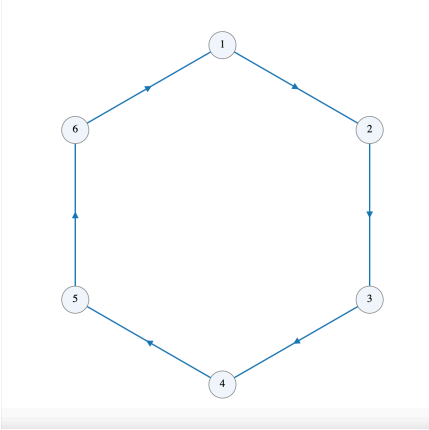

Fig 3 six pointed polygon

A [1 2 3 4 5 6] permutation to [4 5 6 1 2 3] is a cyclic group permutation as it can be achieved with three clockwise cycles, in other words a 180-degree transposition.

In cyclic notation 3([123456]) = [456123]

Cyclic permutations are considered abelian, that is to say, all elements commute. It is the simplest group permutations of order 1. Given it is cyclic, the transposition can be achieved through rotation. A linear visualisation of this permutation confirms that it belongs to the symmetrical sub-groups (S6).

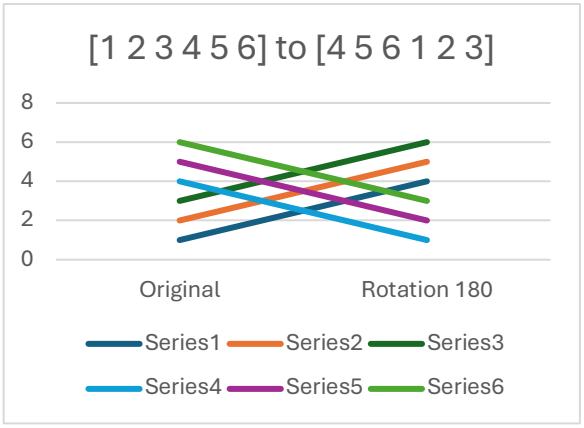

Fig 4 Braid trace graph 180 degree

There are 6 rotation permutations of a *6-gon*, 0, 60, 120, 180, 240 and 300 degrees. These permutations are transpositional. Additionally, there are 6 reflection symmetries of a *6-gon*. Reflections are also known as inversions.

These form the twelve symmetries in the dihedral sub-group (D6).

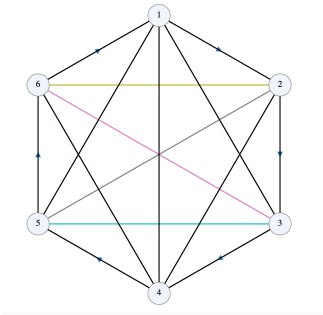

Fig 5 Visualisation of symmetric pathways of D6

Symetrical dihedric permutations of a 6-gon							
	A	B	C	D	E	F	cycle notation
Original	1	2	3	4	5	6	no change *(e)*
Rotation 60 degrees (clockv	2	3	4	5	6	1	(1 2 3 4 5 6)
Rotation 120 degrees	3	4	5	6	1	2	(1 3 5) (2 4 6)
Rotation 180 degrees	4	5	6	1	2	3	(1 4) (2 5) (3 6)
Rotation 240 degrees	5	6	1	2	3	4	(1 5 3) (2 6 4)
Rotation 300 degrees	6	1	2	3	4	5	(1 6 5 4 3 2)
Reflection around 2-5	1	5	3	4	2	6	(1 3) (4 6)
Reflection around 1-4	4	2	3	1	5	6	(2 6) (3 5)
Reflection around 3-6	1	2	6	4	5	3	(1 5) (2 4)
Flip A	2	1	6	5	4	3	(1 2) (3 6) (4 5)
Flip B	4	3	2	1	6	5	(1 4) (2 3) (5 6)
Flip C	6	5	4	3	2	1	(1 6) (2 5) (3 4)

A Linear mapping representation visualises the twelve symmetries, the commutability of the elements of D6 and their symmetrical pathways.

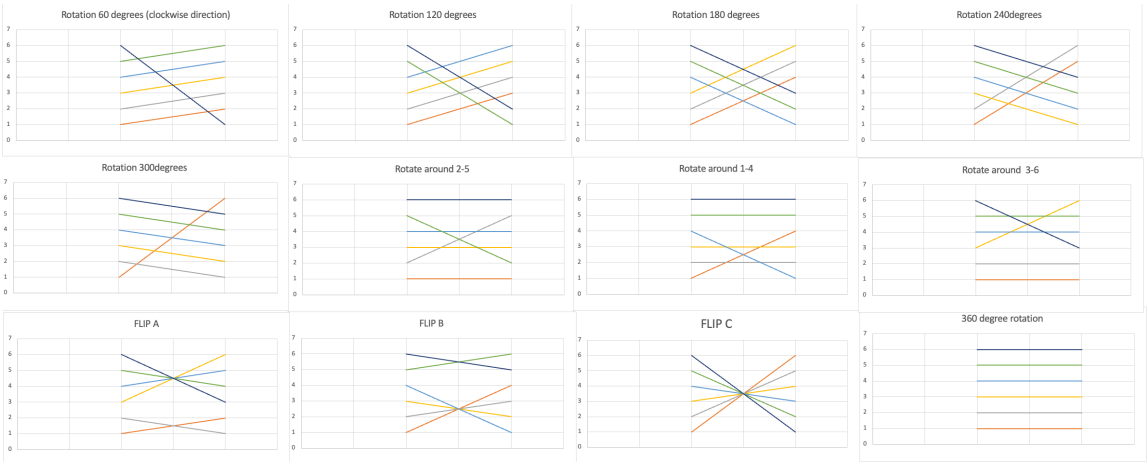

Fig 6 Braid trace remapping graphs

Matrix Implementation

The same permutation groups can be achieved through matrix row and column shifting. In Puredata, the Scratch Engine implements Zmoelnig et al's IEM matrix library [3.18], with which the data set of the 6 containers are placed into a 3X2 matrix.

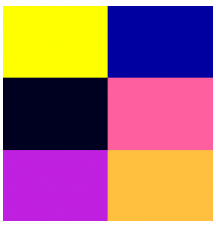

Fig 7*[1=Yellow, 2=Blue, 3=Black, 4=Pink, 5=Violet, 6=Orange]*

By inverting and transposing elements' position in a matrix (through applications of *mtx_roll, mtx_scroll, mtx_inverse* [3.19]), group permutation theory can therefore be preserved and qualified through matrix transformations in the following way:

Symetrical permutations of 3* 2 matrix						
	A	B	C	D	E	F
Original Matrix	1	2	3	4	5	6
Swap rows 1 and 2	3	4	1	2	5	6
Swap rows 1 and 3	5	6	3	4	1	2
Swap rows 2 and 3	1	2	5	6	3	4
Swap rows 1 and 2, then 2 and 3	3	4	5	6	1	2
Swap rows 1 and 3, then 1 and 2	5	6	1	2	3	4
Swap columns 1 and 2	2	1	4	3	6	5
Swap rows 1 and 2, then columns 1 and 2	4	3	2	1	6	5
Swap rows 1 and 3, then columns 1 and 2	6	5	4	3	2	1
Swap rows 2 and 3, then columns 1 and 2	2	1	6	5	4	3
Swap rows 1 and 2, swap rows 2 and 3	4	3	6	5	2	1
Swap rows 1 and 3, swap rows 1 and 2	6	5	2	1	4	3

In linear representation, the 3x2 matrix order permutations can be seen to be almost identical to the 12 symmetries of the *6-gon*.

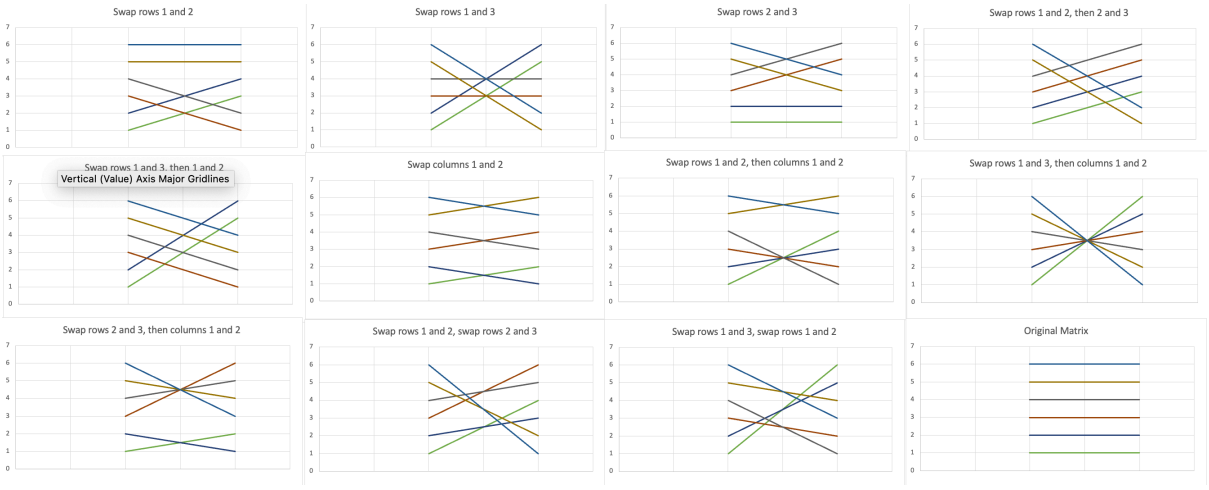

Fig 8 (Note: a 3x2 matrix is not square and doesn't translate on the diagonal or anti-diagonal axis which accounts for the two slight variational differences).

Twelve new 3x2 matrices are plotted from the twelve symmetries of the group permutation sub-group, and it is these order permutations that are categorised in this project as the first set of "data-selfies".

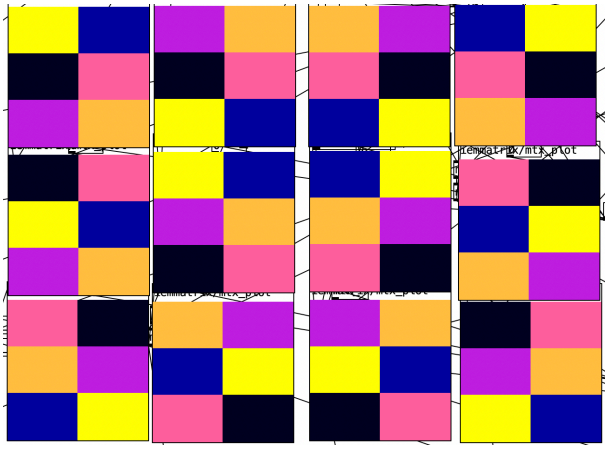

Fig 9 Twelve permutation group matrices

[1=Yellow, 2=Blue, 3=Black, 4=Pink, 5=Violet, 6=Orange]

Until now, the 6 elements of the data string have been referred to as containers. In midi terms, they can be thought of as individual discrete controller numbers [1-6]. In the Scratch Engine, the containers (sliders) contain data, specifically float or integers. In the following example let 1 contain the value 44, 2 contain the value 50, 3 contain the value 53, 4 contain the value 42, 5 contain the value 76, and 6 contain the value 15.

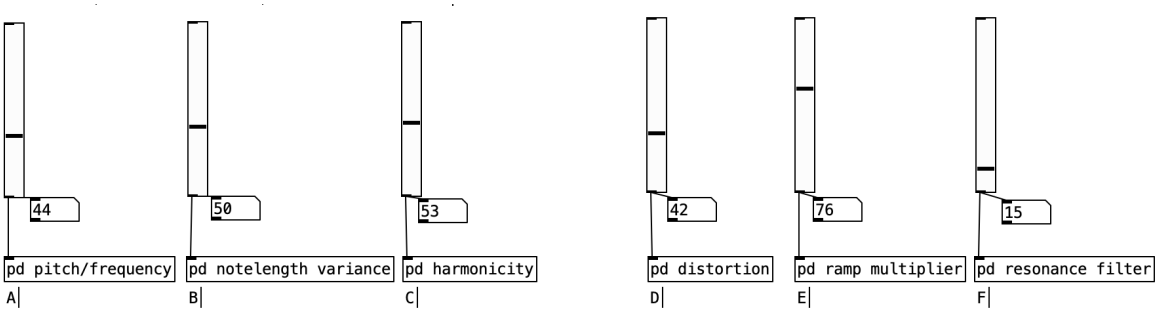

Fig 10 Filling the containers with number data

Now that [1 2 3 4 5 6] = [44 50 53 42 76 15], one can produce matrix group permutations that reorder the string as follows:

original order: [44 50 53 42 76 15]

original transposed: [44 42 50 76 53 15]

rolled+2: [50 53 44 76 15 42]

rolled+2 transposed: [50 76 53 15 44 42]

scrolled+1: [42 76 15 44 50 53]

scrolled+1 transposed: [42 44 76 50 15 53]

scrolledandrolled+1: [15 42 76 53 44 50]

scrolled+1 rolled+1 transposed: [15 53 42 44 76 50]

scrolledandrolledrolled+2: [76 15 42 50 53 44]

scrolled+1 rolled+1 transposed: [76 50 15 53 42 44]

rolled+1: [53 44 50 15 42 76]

rolled+1 transposed: [53 15 44 42 50 76]

The Scratch Engine allows for the composer to choose which permutation order to apply to the dataset being exposed to and associated with the fixed audio parameters A B C D E and F. In the production of the Data Selfie Album, 6 numbers are generated at random (replacing the data supplied by the hands) and populate the containers 1-6. This set of numbers live in the system for a 16-bar cycle, before the containers are repopulated. At each bar, or beat subdivision thereof, a new permutation order from the matrix set is selected thus reordering

the stream being fed into the audio Scratch Engine. Therefore, at every bar the exposed data remains empirical, but is developed and redeveloped through pure pre-reordering.

A typical 16 (or 8) bar structure from the Data Selfie Album:

1)Block of six string data is generated. (Exposition)

2)The data string is containerised into a 3x2 matrix.

3)Commutable ordering group permutations are generated.

4)Association Mapping A-F. One permutation order is selected at each beat or strict subdivision- (Development).

5)At the penultimate bar (for example at bar 15) a new block of data is generated (Cadenza and New Exposition). The cycle begins again.

As input data into the Scratch Engine, the six random numbers and their containers skip or hop every bar between the realational matrices, where their association and reassociation to the audio parameters are exposed. By triggering the next set of random numbers on the penultimate bar (7^{th} in an 8-bar cycle or 15th bar in a 16-bar cycle, the system interprets the last bar not only as a cadenza to the musical phrases but also as an exposition for the following set of phrases. These co-exist as a fluid and melodic anacrusis to the next cycle.

Furthermore, this penultimate transition is ramped (smoothed over time) and thus can be said to exist in latent spaces between permutation orders, momentarily for the duration of the ramp time. In this way, using this track **110bpmDSSoloDemo**

(http://kilshaw.duckdns.org/THE_DATA_SELFIE/THE_DATA_SELFIE_ALBUM/index.html) from the Data Selfie Album as an example, each 16 bar cycle's permutation score can be visualised and a reordering trajectory can be plotted.

Random six string generated	[15 31 42 100 6 90]] = [1 2 3 4 5 6]						
	A	B	C	D	E	F	
ANACRUSIS	1	4	5	2	3	6	Bars 1-16
BAR 1	3	4	5	6	1	2	
2	3	4	5	6	1	2	
3	2	1	6	5	4	3	
4	3	4	5	6	1	2	
5	5	6	3	4	1	2	
6	1	2	5	6	3	4	
7	3	4	5	6	1	2	
8	2	1	6	5	4	3	
9	5	6	3	4	1	2	
10	2	1	6	5	4	3	
11	3	4	5	6	1	2	
12	6	5	2	1	4	3	
13	1	2	5	6	3	4	
14	1	4	5	2	3	6	
15	2	1	6	5	4	3	
Random six string generated	[62 11 40 43 90 22]						
ANACRUSIS	5	6	3	4	1	2	Bars 17 32
BAR 1	3	4	5	6	1	2	
2	5	6	3	4	1	2	
3	5	6	3	4	1	2	
4	1	2	5	6	3	4	
5	2	1	6	5	4	3	
6	1	4	5	2	3	6	
7	1	2	5	6	3	4	
8	3	4	5	6	1	2	
9	6	5	2	1	4	3	
10	1	2	5	6	3	4	
11	3	4	5	6	1	2	
12	2	1	6	5	4	3	
13	1	4	5	2	3	6	
14	6	5	2	1	4	3	
15	5	6	3	4	1	2	
Random six string generated	[17 65 49 102 3 91]						
ANACRUSIS	6	5	2	1	4	3	Bars 33 -48
BAR 1	6	5	2	1	4	3	
2	2	1	6	5	4	3	
3	1	2	5	6	3	4	
4	6	5	2	1	4	3	
5	6	5	2	1	4	3	
6	5	6	3	4	1	2	
7	3	4	5	6	1	2	
8	2	1	6	5	4	3	
9	5	6	3	4	1	2	
10	6	5	2	1	4	3	
11	5	6	3	4	1	2	
12	6	5	2	1	4	3	
13	2	1	6	5	4	3	
14	6	5	2	1	4	3	
15	3	4	5	6	1	2	
Random six string generated	[16 50 118 9 88 31]						
ANACRUSIS	6	5	2	1	4	3	Bars 49-64
BAR 1	5	6	3	4	1	2	
2	3	4	5	6	1	2	
3	1	2	5	6	3	4	
4	6	5	2	1	4	3	
5	5	6	3	4	1	2	
6	6	5	2	1	4	3	
7	3	4	5	6	1	2	
8	1	2	5	6	3	4	
9	3	4	5	6	1	2	
10	1	4	5	2	3	6	
11	2	1	4	3	6	5	
12	1	2	3	4	5	6	
13	4	3	6	5	2	1	
14	1	2	5	6	3	4	
15	1	2	3	4	5	6	
Random six string generated	[53 61 12 83 4 16]						
ANACRUSIS	1	4	5	2	3	6	65-80
BAR 1	1	2	5	6	3	4	
2	6	5	2	1	4	3	
3	1	2	3	4	5	6	
4	6	5	2	1	4	3	
5	6	5	2	1	4	3	
6	1	2	5	6	3	4	
7	6	5	2	1	4	3	
8	5	6	3	4	1	2	
9	6	5	4	3	2	1	
10	1	2	5	6	3	4	
11	2	1	6	5	4	3	
12	1	2	3	4	5	6	
13	5	6	3	4	1	2	
14	6	5	2	1	4	3	
15	2	1	6	5	4	3	
Random six string generated	[32 13 114 62 66 43]						
ANACRUSIS	5	6	3	4	1	2	Bars 81 96
BAR 1	2	1	6	5	4	3	
2	1	2	5	6	3	4	
3	3	4	5	6	1	2	
4	1	2	3	4	5	6	
5	3	4	5	6	1	2	
6	2	1	6	5	4	3	
7	1	2	5	6	3	4	
8	6	5	4	3	2	1	
9	1	4	5	2	3	6	
10	4	3	6	5	2	1	
11	1	2	5	6	3	4	
12	3	4	5	6	1	2	
13	2	1	6	5	4	3	
14	3	4	5	6	1	2	
15	6	5	2	1	4	3	
Random six string generated	[107 4 56 30 33 18]						
ANACRUSIS	6	5	2	1	4	3	Bars 97- 112
BAR 1	1	2	5	6	3	4	
2	3	4	5	6	1	2	
3	6	5	2	1	4	3	
4	6	5	2	1	4	3	
5	3	4	5	6	1	2	
6	2	1	6	5	4	3	
7	6	5	2	1	4	3	
8	2	1	6	5	4	3	
9	6	5	2	1	4	3	
10	2	1	6	5	4	3	
11	6	5	2	1	4	3	
12	6	5	2	1	4	3	
13	2	1	6	5	4	3	
14	6	5	2	1	4	3	
15	2	1	6	5	4	3	
Random six string generated	[6 83 43 50 77 3]						
ANACRUSIS	2	1	6	5	4	3	Bars 113-128
BAR 1	3	4	5	6	1	2	
2	1	2	3	4	5	6	
3	2	1	6	5	4	3	
4	3	4	5	6	1	2	
5	6	5	2	1	4	3	
6	1	2	5	6	3	4	
7	3	4	5	6	1	2	
8	2	1	6	5	4	3	
9	6	5	2	1	4	3	
10	4	3	6	5	2	1	
11	3	4	5	6	1	2	
12	5	6	3	4	1	2	
13	1	4	5	2	3	6	
14	2	1	4	3	6	5	
15	3	4	5	6	1	2	

Fig.11 All 128 bar of **<u>110bpmDSSoloDemo</u>**

The twelve matrix symmetries discussed above are extended through diagonal matrix (*mtx_diag*) permutations to create six additional quasi-symmetrical permutations, producing 18 permutation variations for associative mappings. It is from these specific matrices that the Data Selfie Album is composed.

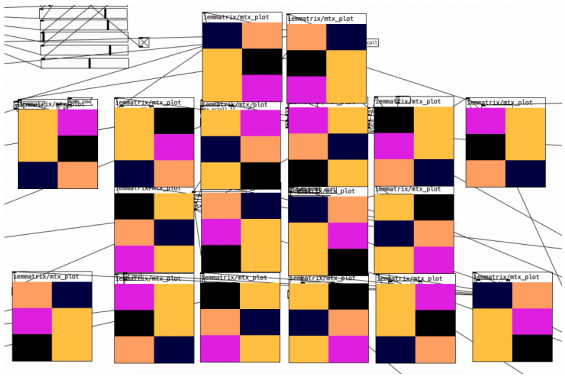

Fig 12 The eighteen quasi symetrical matrix containerised permutations.

16.1 Melodic and Rhythmic Analysis

As well as the Scratch Engine's distinctive sound of the audio synthesis, the environment is programmed to generate midi note outs. Therefore, the outputted rhythm and pitches from the Scratch Engine can not only be re-voiced in DAWs and other environments but can be further analysed by a midi translation of onset rhythms, silences or rests and pitches. In this way one can understand the observable underlying structure and phrase patterns generated by the system. For this analysis, as well as visualising the melodic contours, it is also helpful to employ Salzer's structural hearing approach [3.20], to aesthetically ascertain the patterned conjuncts between notes as the system constructs linear melodic phrases. From the output, one observes some clear principles of voice leading theory, namely that of smoothness in stepwise motion, the lead voice is independent and balanced. Once a fundamental line is established, initial ascents can begin the elaboration.

Neighbour notes or passing notes link the phrased elements through decoration between the spans of top and bottom notes. A linear fluency in the melodic arc is observed. Occasional interruptions are allowed and form distinct sub motifs of the fundamental line being developed.

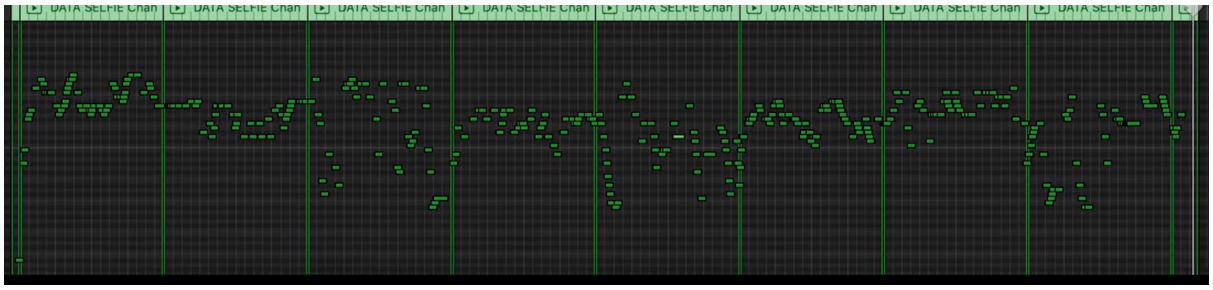

Fig 13 : https://www.youtube.com/watch?v=z3fawBnic3w

(8 cycles of the Scratch Engine 100 bpm)

Understanding this structurally, in terms of the group permutation matrix actions, each launch of a set of new random numbers instigates a cadenza and an exposition at the same time, identified figure n below in red. It resolves the previous cycle and introduces or exposes the next prime data stream. It can be heard to act as an anacrusis to the new prime data stream highlighted in blue. The prime data stream as an exposition instigates the fundamental line, where a new musical idea and the subsequent elaborations are developed as the system plays out the order permutation parameters, identified in green. These elaborations are littered with phrases that are clearly related to the prime data's fundamental and is when the phrases are melodically and rhythmically at their most fluent. At the penultimate bar the cycle returns to red, to jointly provide a cadenza and a new exposition and distinctively provides the anacrusis into the next new musical idea, and so on.

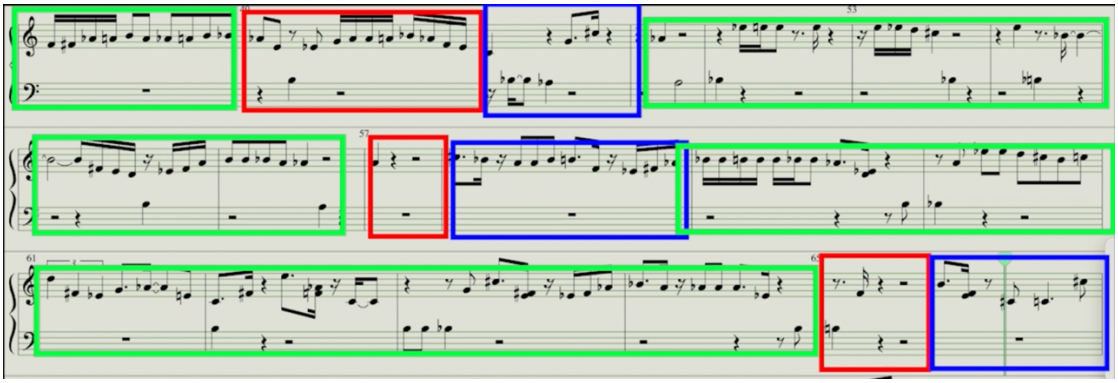

Fig 14 Score : 9m.41s -11m.30s https://www.youtube.com/watch?v=2U42qyU4Mh0&t=2s

The Scratch Engine will generate the lead instrument solo ad nauseum. [21] By generating a new set of six random numbers to populate the six containers every 16 bars (rather than the performer's input data) the instrument can play in automatic mode, ever evolving, but always congruently fluid through its internal logic. There is a spontaneous and expressive fluidity between beats, between bars and between larger scale cycles.

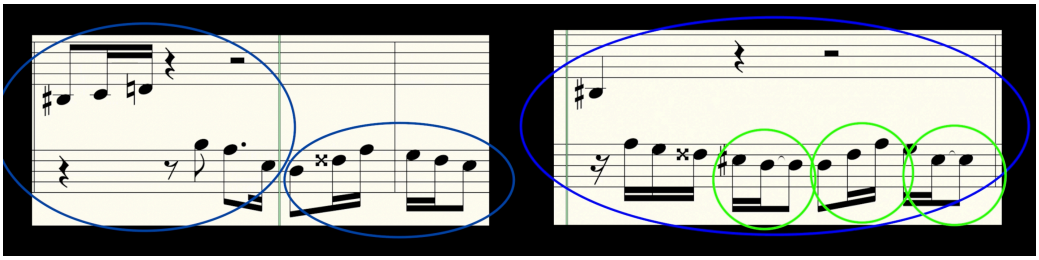

Fig 16 left (single bar: motion two melodic ascents and two descents with transposition).

Fig 17 right (single bar: mirrored rhythm over melodic arc(green)).

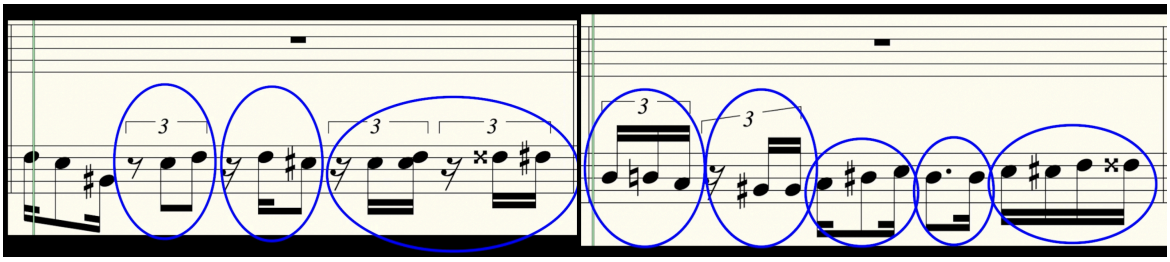

Fig 18 left(single bar: rests as "ghost" downbeats inside triplet cells).

Fig 19 right(single bar: fluid rhythmn modulation from triplets to straight on beat within a bar).

The scratching patterns outputted by the software have direct audible comparisons to established DJ scratch patterns and techniques, with particular reference to the Scratching Pattern Taxonomy laid out by DJ QBert [3.22]. These traditionally turntablist techniques (chirps, squiggles, orbits, baby scratches, transformers, autobahns, flares, tears, multi click crabbing) are all recognisably manifest in the Data Selfie Album, particularly in **130bpmSoupGlitchDemo** , albeit with its own distinctive prosody.

The Scratch Engine's midi output capabilities allow for parallel instrumental comping and scaled symmetrical noodling. In this revoiced example, [3.23] a midi Smokey-Clav comps over an oblique single chord loop, in a melodically harmonically congruent way. Pentatonic scaling in this example [3.24] allows the voice leading muted guitar to relate harmonically to the chord progressions in the track. Furthermore, when midi onsets are applied to midi gating techniques, some interesting symmetrical interjections and embellishments become an additional rhythmic feature to the production of the track.[3.25] Stylistic and revoiced experiments exist on the project's homepage.

16.2 Evaluation

The Data Selfie Album as an artefact, is the audio manifestation of applying dihedral group permutation theory to associative mapping and rhythmic remapping. In this way, containerising the elements of data strings is a pre-composition and pre-production

technique. Representationally, the six audio parameters in these given examples (A-F) have dictated the size of the *n-gon*, and as a corelative, the size of the matrices. It stands that the methodology is scalable. By letting the total number of target end-chain audio transformation parameters dictate the size of the *n-gon*, (and therefore it's possible symmetric and quasi-symmetric permutations), a containerised approach to permutation mapping will stand. By example, an associative mapping methodology could extend to an instance of an 8 element (parameter) granular synthesiser :

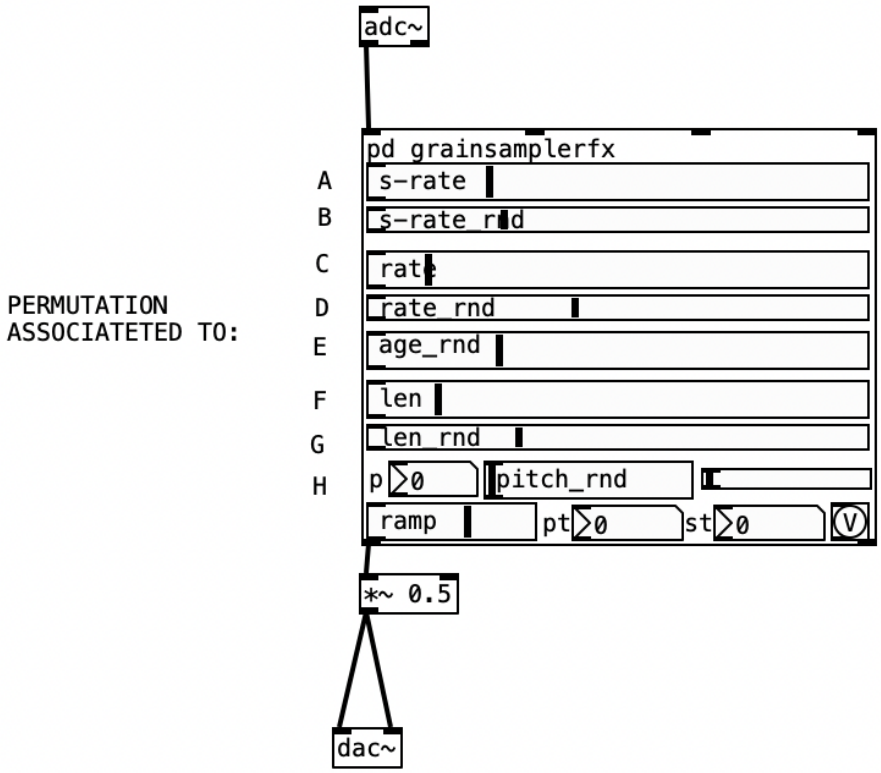

Fig 20 from Grainsamplerfx.pd (Brinkmann's Library)

Where, as containers, A-H could be mapped from the 8 rotations and 8 reflections of D8.

SYMMETRICAL GROUP PERMUTATIONS D8	A	B	C	D	E	F	G	H
ORIGINAL	1	2	3	4	5	6	7	8
Rotation by 45°	8	1	2	3	5	5	6	7
Rotation by 90°	7	8	1	2	3	4	5	6
Rotation by 135°	6	7	8	1	2	3	4	5
Rotation by 180°:	5	6	7	8	1	2	3	4
Rotation by 225°	4	5	6	7	8	1	2	3
Rotation by 270	3	4	5	6	7	8	1	2
Rotation by 315°	2	3	4	5	6	7	8	1
Reflection vertices 1 and 5:	1	8	7	6	5	4	3	2
Reflection vertices 2 and 6:	7	2	1	8	5	6	3	4
Reflection vertices 3 and 7:	5	4	3	2	1	8	7	6
Reflection vertices 4 and 8:	3	4	5	6	7	8	1	2
Reflection midpoints sides 1-2 and 5-6:	2	1	8	7	6	5	4	3
Reflection midpoints sides 2-3 and 6-7:	8	7	6	5	4	3	2	1
Reflection midpoints sides 3-4 and 7-8:	6	5	4	3	2	1	8	7
Reflection midpoints sides 4-5 and 8-1:	4	3	2	1	9	7	6	5

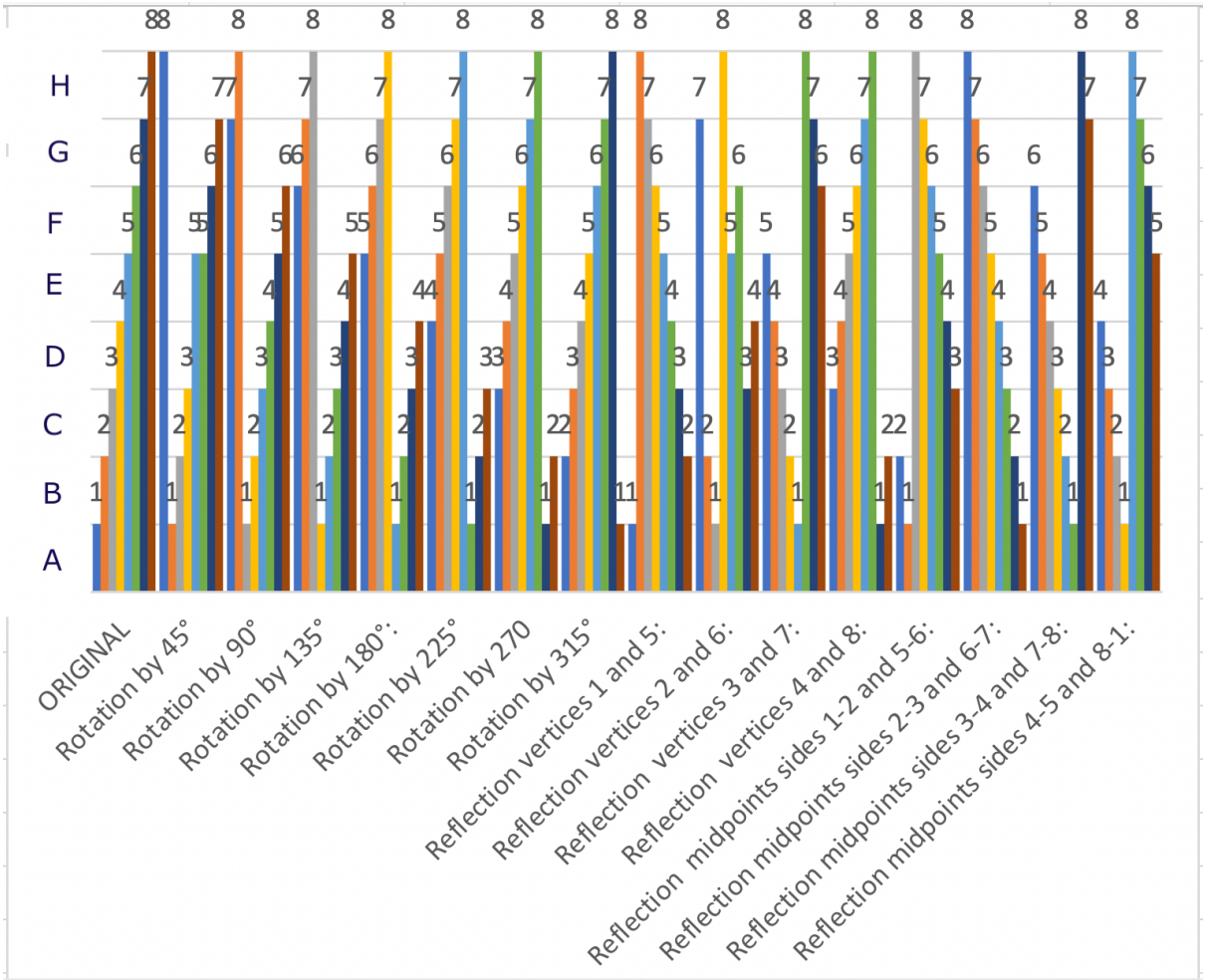

Fig 21 Example (Dihedral symmetries of an 8-gon) Proposed possible Octo-phonic speaker array mapping

This serialist mapping methodology is extendable across many computer music domains, compositional sound worlds, genres and data applications and is intended not just as a software instrument presented above, but as an accurate and efficient creative framework for data-composers and artists. An extended variety of stylistic genres with lead line soloing derived from the Scratch Engine are presented here:

http://kilshaw.duckdns.org/prosodic_development/indexb.html

17. Creative Outcomes: Album as Practice-Based Evidence

17.1 The Album as a Formal Outcome of the System

The album produced as part of this research constitutes a primary creative outcome and functions as material evidence of the Scratch Engine's operational and theoretical claims. Rather than being conceived as a collection of discrete compositions authored in the conventional sense, the album documents the behaviour of the system operating autonomously under defined constraints. Each track captures a sustained traversal through the system's permutation space, rendering audible the structural processes described in Chapters 6–11. In this context, the album should be understood not as a representational artefact that illustrates prior compositional decisions, but as a performative trace of an internally governed transformational process. Once configured, the Scratch Engine generates numerical state vectors and applies dihedral permutations at predefined temporal resolutions without continuous human intervention. The recorded output therefore reflects the system's capacity to maintain formal coherence over extended durations through the conservation of material and the systematic reassignment of

functional roles. This framing aligns with practice-based research methodologies in which creative artefacts operate as sites of knowledge production rather than objects of aesthetic validation. The album functions as a temporal exposition of the system's structural logic, making audible the real-time execution of dihedral group action upon invariant material. In this sense, the tracks may be read as structural studies, each articulating a particular pathway through the twelve symmetries of the hexagon while preserving a consistent internal identity. The album thus serves as an empirical manifestation of the system's formal design, confirming that the theoretical principles outlined earlier in this thesis are not merely abstract, but operational, audible, and sustainable in practice.

17.2 Listening as Perception of Transformation

Listening to the album invites transformation as the primary perceptual dimension, rather than thematic development or motivic variation. Because the underlying numerical material remains invariant for extended formal durations, change is experienced not as the introduction of new content, but as the reconfiguration of relationships between stable elements. This mode of listening aligns with Lewin's conception of musical meaning as emerging from transformational paths rather than from fixed musical objects (Lewin, 1987).

The continuity of the sonic output is maintained through conservation: energy, density, and material identity persist even as functional roles are repeatedly reassigned. As a result, musical motion is perceived as circulation within a constrained space rather than progression toward a goal. This produces a listening experience characterised by flow without teleology, in which structural change is continuous yet non-narrative. The listener encounters permutation as motion, symmetry as stability, and variation as relational

displacement rather than replacement. Such perceptual conditions resonate with post-serial and transformational listening practices, in which attention is directed toward processes, mappings, and structural affordances rather than surface-level events. The album therefore invites a mode of engagement that is analytical as much as experiential, encouraging listeners to apprehend structure through repetition with difference. Importantly, perceptual intelligibility is not achieved through simplification, but through the rigorous restriction of transformational possibilities. The closed nature of the dihedral group ensures that all perceived change remains structurally related, reinforcing coherence even under rapid permutation.

17.3 Aesthetic Character and Structural Fluency

While the album is not evaluated in terms of aesthetic preference, its sonic character can be described in structural and operational terms. The output consists of continuous, uninterrupted streams of melodic and rhythmic material articulated within a stable metric framework, typically in common time. This temporal regularity functions as a stabilising grid against which structural transformation is rendered perceptually legible, allowing rapid reconfiguration of functional roles without disorienting the listener. The gestural quality of the music, which can be observed to be scratching or turntable-like in character, emerges not from direct physical manipulation but from the systematic reassignment of control parameters. Articulation, contour, and rhythmic emphasis are products of permutation rather than expressive gesture, resulting in phrasing that appears fluid and performative despite its autonomous generation. This apparent paradox underscores the system's capacity to produce musically coherent output without reliance on embodied virtuosity. Crucially, the album demonstrates that structural rigor and sonic fluency are not mutually

exclusive. Extended passages sustain continuity without stasis, and contrast without rupture, confirming that permutation-based systems can support long-form musical trajectories. Rather than developing material through thematic elaboration, the music maintains interest through the ongoing recontextualisation of invariant elements. In doing so, the album exemplifies a form of operational serialism in which structure is not imposed retrospectively, but unfolds dynamically as an audible, time-based process.

18 Limitations and Future Work

The Scratch Engine is intentionally framed as a constrained formal system, grounded in dihedral symmetry, autonomous operation, and the permutation of functional roles rather than musical values. While these constraints underpin the system's conceptual coherence and aesthetic focus (Chapters 3–5), they also define the boundaries of the present research. This chapter articulates those limitations explicitly and outlines directions for future work that extend, rather than contradict, the theoretical position established earlier.

18.1 Serialism Beyond Pitch: Conceptual Boundaries

As discussed in Chapter 8 (Relation to Serialist and Algorithmic Practice), historical serialism emerged primarily as a response to the perceived exhaustion of tonal pitch organisation, with pitch-class ordering functioning as its central structural concern (Schoenberg, 1923; Babbitt, 1965). By contrast, the Scratch Engine abstracts serial logic away from pitch classes and applies it to the permutation of parameter roles (Chapters 2 and 6). This abstraction constitutes a conceptual limitation insofar as it departs from the historical specificity of serialist practice. While transformational theory explicitly permits such abstraction by

focusing on the action of groups rather than the musical objects they transform (Lewin, 1987; see Chapter 4.4), critics may argue that the system risks generalising serialism into a neutral permutation logic. Future research could address this by hybridising role-based permutation with pitch-class serial constraints, allowing historically grounded serial techniques and post-serial abstraction to coexist within a single system.

18.2 Symmetry, Determinism, and Expressive Constraint

The exclusive reliance on the dihedral group D_6, detailed in Chapter 6, provides mathematical closure and perceptual legibility but also introduces the risk of expressive determinism. Post-serial composers such as Ligeti and Lachenmann have explicitly critiqued the aesthetic consequences of rigid symmetrical systems, arguing that excessive regularity can suppress perceptual tension and formal rupture (Ligeti, 1968; Lachenmann, 1993). Although the Scratch Engine mitigates this risk through temporal subdivision and continuous interpolation (Chapter 8.4), its transformational vocabulary remains finite and closed. Future work could explore controlled symmetry-breaking strategies, such as dynamic group switching, probabilistic deformation of transformations, or multi-group interaction, extending the formal language while preserving the system's structural clarity.

18.3 Autonomy and Compositional Authorship

As established in Chapters 1 and 4, the Scratch Engine is conceived as an Autonomous Technological Performance System (ATPS), operating independently once initial constraints are defined (Rowe, 1993). This design emphasises structural autonomy over performer responsiveness, deliberately minimising moment-to-moment human intervention (Chapter 8.4). A limitation of this approach is that autonomy remains procedurally bounded. While

the system autonomously traverses permutation space, it does not generate or revise its own transformational grammar. Future research could investigate meta-transformational systems capable of modifying their governing group structures over time, aligning more closely with adaptive and evolutionary models of musical agency (Miranda, 2009).

18.4 Performer Agency and Embodiment

The reduction of direct gestural control, discussed in Chapters 4.2 and 13 (Methodology), represents a deliberate challenge to performer-centric models of digital musical instruments (Wanderley & Battier, 2000). While this aligns with posthuman and cyborg-centric perspectives on distributed agency (Haraway, 1991; Hayles, 1999; see Chapter 4.3), it may be perceived as limiting embodied expressivity. This limitation is conceptual rather than technical. Future work could explore layered agency models in which human gesture modulates higher-order system parameters—such as permutation density or temporal resolution—without reintroducing the gestural determinism explicitly rejected in the current methodology (Chapter 13.1).

18.5 Stylistic and Cultural Specificity

Although formally general, the system's musical outputs are stylistically situated within Western metric frameworks, particularly 4/4 time, and draw heavily on scratching practices and electronic improvisation (Chapter 11). This stylistic anchoring limits the generalisability of the system across broader cultural and musical contexts. Future research could apply the Scratch Engine to non-metric temporal structures, alternative tuning systems, or non-Western rhythmic frameworks, testing whether dihedral permutation retains perceptual coherence beyond its current idiomatic domain (Born, 2005).

18.6 Analytical and Perceptual Validation

While Chapters 8 and 11 demonstrate structural coherence through transformational and voice-leading analysis (Lewin, 1987; Cohn, 2012), the present research does not include empirical listener studies. As a result, claims regarding perceptual intelligibility remain analytically rather than empirically grounded. Future work could incorporate listener-based evaluations, psychoacoustic testing, or comparative studies with other generative systems to assess how dihedral permutation is perceived over extended durations.

These limitations define the scope of a deliberately focused intervention rather than deficiencies in execution. By constraining itself to a closed symmetry group, autonomous temporal unfolding, and abstract parameter roles, the Scratch Engine articulates a precise theoretical position within post-serial, transformational, and posthuman music discourse (Chapters 3–4). Future work will extend this position outward, testing the resilience of its principles under expanded musical, cultural, and computational conditions.

19. Conclusion

The Scratch Engine represents a performable musical instrument in which dihedral symmetry, serial logic, and real-time computation converge to produce a coherent autonomous system. By embedding the full action of the dihedral group D_6 directly into the control architecture, the system reconceptualises permutation as a continuous, audible, and manipulable process rather than a pre-compositional abstraction. Its principal contribution

lies in reconfiguring serial principles for real-time musical practice. By permuting functional roles rather than the numerical values themselves, the system enables structural variation while preserving material identity, producing a form of operational total serialism that unfolds dynamically. Serial organisation is thus retained at the level of relational structure while liberated from static matrices or score-based realisation.

Operating autonomously once configured, the Scratch Engine challenges conventional models of interaction in digital musical instruments. Musical form emerges from the execution of a constrained transformational grammar rather than continuous performer–system dialogue. Performer agency is exercised through system design and configuration, whereas autonomous execution governs temporal unfolding, foregrounding structure itself as a musical phenomenon in accordance with transformational theory (Lewin, 1987) and Xenakis' conception of composition as navigation through formal spaces (Xenakis, 1971). Artistically, the instrument produces uninterrupted streams of melodic and rhythmic material characterised by rapid re-contextualisation rather than thematic development. Extended performances demonstrate its capacity to sustain long-form musical trajectories without external segmentation or post-hoc editing. More broadly, the system provides a model for how abstract mathematical structures can be rendered perceptually explicit and musically productive within real-time systems, illustrating the convergence of formal rigour, performative immediacy, and posthuman musical agency.

20. Bibliography References

 Montague J (2017): Journal: Frontiers in Sociology.How Music and Instruments Began: A Brief Overview of the Origin and Entire Development of Music, from Its Earliest Stages University of Oxford

 Pinch, T.& Trocco, F.(2002).Analog Days: The Invention and Impact of the Moog Synthesizer.Harvard University Press.

 Blades, J.(1992).Percussion Instruments and Their History. Bold Strummer Ltd.

 Théberge, P.(1997).Any Sound You Can Imagine: Making Music/Consuming Technology.Wesleyan University Press.

 Magnusson, T (2021): The migration of musical instruments: On the socio-technological conditions of musical evolution. Journal of New Music Research, 50:2, 175-183

 Homer Homer, Odyssey 8.62–100.& Homer, Iliad 9.186–191.

 Bucknell, B. (2020). Aesthetics, music, noise. In A. Snaith (Ed.), Sound and Literature. Cambridge University Press.

 Marinetti T et al (1913) Futurists Manifesto Passerino Editore

 Russolo's Intonarumori: Musical Innovation at the Beginning of the Twentieth Century.(2012). International Yearbook of Futurism Studies,2(1). https://doi.org/10.1515/futur-2012.0020

 Russolo, L.(1913) The Art of Noises (L'arte dei rumori).

Originally written as a letter to Francesco Balilla Pratella, March 11, 1913.

First published as a booklet by the Futurist Movement, Milan, July 1913

 Stiegler, B.(1998). Technics and Time, 1: The Fault of Epimetheus (G.

Collins & R. Beardsworth, Trans.).Stanford University

 Skeldon, Kenneth D., et al. "Physics of the Theremin." American Journal of Physics, vol.66, no.11, 1998, pp.945–955. doi:10.1119/1.19004

 Schaeffer P: Solfege de'lobjets sonores 1948 InaGRM distributions

 Chowning, John M. "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation." Journal of the Audio Engineering Society, vol.21, no.7,1973, pp.526–534

 Cage, John. Imaginary Landscape No.1. Composed in 1939 performed by the Percussion Ensemble directed by Jan Williams, recorded May 28 – June 1, 1995, at Slee Concert Hall, University at Buffalo, and released by Hat Hut Records in 1995

 Tenney, James. Collage No.1 (Blue Suede), 1961

 Varèse, Edgard. Ionisation.Composed 1929–1931.Premiered March 6, 1933, at Carnegie Chapter Hall, New York City, conducted by Nicolas Slonimsky

 Ian Simon, Anna Huang, Jesse Engel, Curtis "Fjord" Hawthorne https://colab.research.google.com/drive/1XQjebKPHWWHcVps7V2dXpFovnEnoordc

 Nathanielle, T: Music Machine Learning : https://www.interviewquery.com/p/music-machine-learning

 Collins, N., & d'Escriván, J.(Eds.).(2007).The Cambridge Companion to Electronic Music.Cambridge University Press.

Agostini, A. & Ghisi, D. (2013). Gesture-based live electronics: Theoretical and practical issues. Proceedings of the International Computer Music Conference.

Babbitt, M. (1965). The use of computers in musicological research. Perspectives of New Music, 3(2), 74–83.

Boulez, P. (1954). Structures I. Paris: Heugel.

Haraway, D. (1991). Simians, Cyborgs, and Women: The Reinvention of Nature. New York: Routledge.

Hayles, N. (1999). How We Became Posthuman. Chicago: University of Chicago Press.

Lewin, D. (1987). Generalized Musical Intervals and Transformations. New Haven: Yale University Press.

Morris, R. (1998). Composition with pitch-classes: A theory of compositional design. New Haven: Yale University Press.

Roads, C. (1996). The Computer Music Tutorial. Cambridge, MA: MIT Press.

Rowe, R. (1993). Interactive Music Systems. Cambridge, MA: MIT Press.

Taube, H. (2009). Notes from the Metalevel: An Introduction to Algorithmic Music Composition. Lisse: Swets & Zeitlinger.

Xenakis, I. (1971). Formalised Music: Thought and Mathematics in Composition. Bloomington: Indiana University Press.

Zicarelli, D. (2002). How I learned to love a program that does nothing. Computer Music Journal, 26(4), 44–51.

 Hayles K: How we became posthuman. Virtual Bodies in Cybernetics, Literature, and Informatics. The University of Chicago Press Chicago er London 1992

 Latour, B. (1987). Science in Action. Harvard University Press.

 Haraway, D.(1991). A Cyborg Manifesto: Science, Technology, and Socialist-Feminism in the Late Twentieth Century.In Simians, Cyborgs and Women: The Reinvention of Nature (pp.149–181).New York: Routledge

 Lewis D: Voyager (continuing) https://youtu.be/wGfeh7MNTpU?si=nuf4QY9kZ8j8cC2O

Bailey, Derek.1993.Improvisation: Its Nature And Practice In Music. Da Capo Press.

 Essl K, Lexicon Sonate (2020) https://www.essl.at/works/Lexikon-Sonate.html

 Essl K, RTClib (2022): https://www.essl.at/works/rtc.html

 Holly Herndon PROTO (2019) https://hollyherndon.bandcamp.com/album/proto

 Cope, D : https://computerhistory.org/blog/algorithmic-music-david-cope-and-emi/)

 Collins, N., & McLean, A.(2014). Algorave: Live Performance of Algorithmic Electronic Dance Music. In Proceedings of the International Conference on New Interfaces for Musical Expression (NIME), pp.355–358.Goldsmiths, University of London

 Pamela Z Voci https://youtu.be/PqGD2N1FLGg?si=oCkiXCzyDHaSlSRP

 Kirby J Dr: Phonetics https://youtu.be/k1FznkrjpWg?si=knrZP554FVWn8hlD

 Ircam suite https://www.ircamlab.com/

 GrmTools https://inagrm.com/en/store

 Beast-Tools https://soundcloud.com/birminghamelectroacousticsoundtheatre

 IntegraLive. https://integra.io/portfolio-items/integralive/

 AJAXSOUND. Piquet and Belanger's https://github.com/belangeo/cecilia5

 Article in Computer Music Journal · September 2002 Jorda S: FMOL: Toward User-Friendly, Sophisticated New Musical Instruments (2006)

 PureData)http://puredata.info

  (BitRate GrayArea workshop), BitRate Series: Building An Interactive Machine Learning Orchestra Using ml5.js Workshop Attended 8/14/2020

 Parviainen 2025 https://github.com/teropa

 Grond, F., & Hermann, T.(2012). Aesthetic strategies in sonification. AI & Society, 27(2), 213–222. https://doi.org/10.1007/s00146-011-0341-7

 https://www.essl.at/bibliogr/improvisation-e.html#komp-improv

 Magenta https://magenta.tensorflow.org/

 Music Transformer models https://colab.research.google.com/drive/1XQjebKPHWWHcVps7V2dXpFovnEnoordc

 Los Angeles Music Composer https://github.com/asigalov61/Los-Angeles-Music-Composer

 https://paperswithcode.com/datasets?task=music-generation

 https://magenta.tensorflow.org/music-vae

 Ross Cole 2020 The Problem with AI Music: Song and Cyborg Creativity in the Digital Age Cambridge University press

 Pachet, F.et al.(2016).Flow Machines: Creative Music Generation. Sony CSL

RNN:https://kilshaw.duckdns.org/Squash_steps_weighted_rnn/index.html

 Born, G. (2005). On musical mediation: Ontology, technology and creativity. Twentieth-Century Music, 2(1), 7–36.

 (Huang, C.-Z. A., Vaswani, A., Uszkoreit, J., et al.(2018). Music Transformer: Generating Music with Long-Term Structure.

 Briot, J.-P., Hadjeres, G., & Pachet, F.-D.(2019). Deep Learning Techniques for Music Generation – A Survey. Springer

 AIVA: https://www.aiva.ai/

 Los Angeles Composer: https://colab.research.google.com/github/asigalov61/Los-Angeles-Music-Composer/blob/main/Los_Angeles_Music_Composer.ipynb

 Colton, S., López de Mántaras, R., & Stock, O.(2020). AI and Creativity: An Introduction to the Special Issue.AI Magazine 41(3), 5–7.Dhariwal, P., Jun, H., Payne, C., et al.(2019).

MuseNet.OpenAI.Retrieved from https://openai.com/research/musenet

Yamaha Corporation.(2019). Dear Glenn: AI Piano Performance System.

Retrieved from https://www.yamaha.com/en/about/ai/dear_glenn/

Shlomowitz M. IT'S NOT ABOUT YOU: DO WE STILL NEED AN 'ARTISTIC VOICE'? Tempo. 2024;78(308):38-45. doi:10.1017/S0040298223000980

 Chen, J. (2024). Dancing with math: Using Klein's quartic for music generation. Theoretical and Natural Science, 39, 23–42.

 Luo, J. (2025). Comparison of Algorithmic Music Composition: Translational Models, Mathematical Models, and AI Tools. Applied and Computational Engineering, 160, 45–54.

 Luo, W. (2024). Music102: An D_{12}-equivariant transformer for chord progression accompaniment.

 Browne, E. (2025). The Shape of Surprise: Structured Uncertainty and Co-Creativity in AI Music Tools.

 Ni-Hahn, S., Yang, C. P., Ma, M., Rudin, C., Mak, S., & Jiang, Y. (2025). ProGress: Structured Music Generation via Graph Diffusion and Hierarchical Music Analysis.

 Stefánsdóttir, H. S. et al. (2025). Of altered instrumental relations: agency through musical performance with neural audio synthesis and violin. Frontiers in Computer Science.

 Systematic review of AI-based music generation. Expert Systems with Applications (2022).