Gesture Before Symbol: Recovering Latent Structure in Keyboard Input

Flyxion
Independent Researcher
github.com/standardgalactic

April 26, 2026

Abstract

Keyboard input is conventionally modelled as a discrete symbolic stream: keystrokes are recorded, ordered, and emitted as a linear sequence of characters that serve as the primitive units of computation and communication. This model is practically successful but structurally lossy. A skilled typist does not produce a sequence; they produce a coordinated, overlapping, temporally extended gesture distributed across both hands. Standard input systems discard most of this signal, reducing a high-bandwidth relational object to an ordered list of keydown events. This paper formalises the latent structure that is lost. We introduce a hierarchical gesture space in which individual finger states, intra-hand coordination, and inter-hand geometry constitute three distinct levels of representation. A canonicalisation operator maps raw event streams onto equivalence classes that preserve topological features—overlap, pivot, and segmentation—while discarding incidental timing variation. Outputs such as text and music are then defined as projection functors from this canonical gesture space, not as defining representations. We further introduce an admissibility functional that constrains which gesture–projection pairs are coherent, yielding a two-stage structure of representation and validation. The central claim is that gesture is the primary object, symbolic output is a projection of it, and admissibility is the constraint layer that governs which projections can be stably realised. The instrument required to read this signal does not demand a new form factor; it demands that existing hardware be read at higher temporal and relational resolution than current pipelines permit.

1. Introduction

Every modern keyboard interface is built on a simplifying assumption: that human input is fundamentally a sequence of discrete symbols. Keystrokes are recorded, ordered, and emitted as a linear stream of characters, which then serve as the primitive units for computation, communication, and control. This assumption has proven enormously practical [7], but it is also structurally lossy.

A skilled typist does not produce a sequence. They produce a coordinated, overlapping, temporally extended gesture distributed across both hands. Multiple fingers are active simultaneously; contact is sustained or released with varying duration; trajectories migrate across local regions; and reversals of motion introduce structured inflection points. These features are not incidental—they are stable, reproducible aspects of trained motor behaviour [9, 8]. They are, however, almost entirely discarded by existing input systems.

The conventional keyboard pipeline performs a systematic flattening. Continuous and concurrent motor activity is reduced to discrete key events; overlapping states are serialised; temporal structure is compressed into ordering; and relational information between fingers and hands is eliminated. What remains is a symbolic stream that is easy to process but no longer reflects the structure of the gesture that produced it.

This paper argues that the discarded structure is not noise but signal. The hands already generate a rich, low-dimensional, relational object that can be described independently of any particular symbolic interpretation. Rather than treating gesture as an intermediate artefact to be resolved into text, we treat it as the primary object of representation. Text, music, and other modalities are then understood as projections of this object, not as its defining form.

To make this claim precise, we introduce a formalisation of keyboard input as a hierarchical gesture space. At the lowest level, individual fingers operate over constrained local domains, producing time-indexed states with activation, position, and directional change. These are aggregated into hand-level structures capturing intra-hand coordination and dominant motion patterns. At the highest level, inter-hand relations—convergence, divergence, synchronisation, and asymmetric anchoring—define the global structure of the gesture. A canonicalisation operator maps raw event streams onto equivalence classes that preserve topological features such as overlap, pivot, and segmentation, while discarding incidental timing variation.

Within this framework, meaning is defined over equivalence classes of relational trajectories rather than over sequences of discrete symbols. Projection functions map canonical gestures into text, musical structures, or other output domains. These mappings may vary without altering the underlying gesture space, allowing a single learned motor vocabulary to support multiple modalities simultaneously. A further admissibility layer constrains which projections are coherent, completing the picture: gesture is represented, rendered through projection, and validated through admissibility.

The central claim is therefore not that new input techniques must be invented, but that an existing signal must be recognised. The gesture is already present in ordinary typing practice; what is missing is an instrument capable of reading it. Crucially, this absence is not a matter of software interpretation alone: standard keyboard hardware reports key events through matrix-scanned, debounced circuits that prioritise reliable symbol entry over high-resolution concurrent state, effectively collapsing overlapping and time-sensitive structure at the point of capture. By articulating the structure of this latent signal and the consequences of its systematic loss, we aim to reframe keyboard input not as a problem of symbol entry, but as a problem of gesture representation.

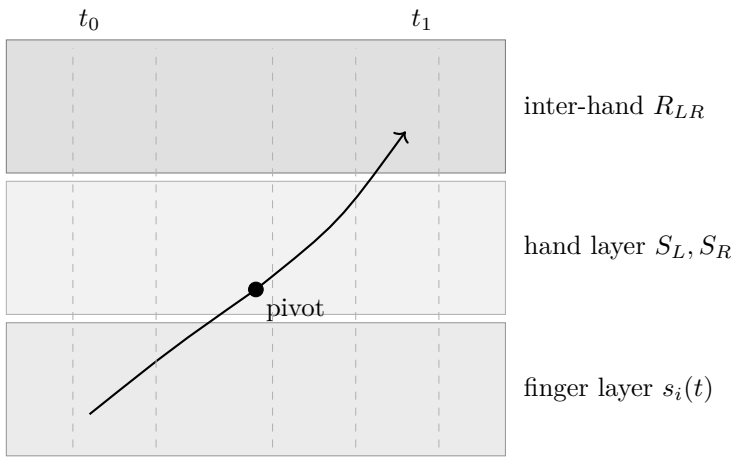


Figure 1: A gesture G as a trajectory through three hierarchical layers of coordination. The pivot (red) marks a reversal under sustained contact—the primary structural feature preserved by canonicalisation [13].

2. Gesture Before Symbol: Developmental and Evolutionary Context

The title of this paper carries a double meaning. In the technical sense developed throughout, "gesture before symbol" refers to the argument that the relational motor structure of keyboard input precedes and grounds its symbolic interpretation: $\hat{G}$ is the primary object, and $\Phi_i(\hat{G})$ is the projection. But the phrase also names an empirical finding from developmental psychology and evolutionary anthropology that independently supports the same logical priority.

2.1 Ontogenetic Precedence

Research in developmental psychology establishes that communicative gestures systematically precede symbolic communication in human infants [16]. Between approximately ten and fourteen months, infants develop deictic gestures—pointing, showing, reaching—that function as requests or references to objects before spoken words serve the same function. Crucially, gesture-symbol combinations (pointing while vocalising) appear before symbol-symbol combinations (two-word utterances). The gesture does not merely accompany emerging language; it scaffolds it. The child first establishes a communicative act through motor action, and the symbol is learned into that already-functional gestural

slot [17].

This developmental sequence has a direct structural parallel in the framework of this paper. The gesture $\hat{G}$ is not derived from or dependent on the projection Φ_{text}. The projection is learned into the gesture. A child learning the word "bottle" already has a functional pointing gesture that selects the referent; the word fills that pre-existing communicative role. The canonical gesture precedes and grounds the symbol in exactly the same sense that $\hat{G}$ precedes $\Phi_i(\hat{G})$.

2.2 Evolutionary Precedence

The pattern extends across species. Language-enculturated great apes — chimpanzees and bonobos trained to use lexigrams or sign language — reliably use reaching and showing gestures to indicate intent well before they use the corresponding symbolic lexigrams for the same purpose, and continue to use gestural reference in contexts where symbolic access is uncertain [18]. This cross-species commonality suggests that gestural communication is not merely a human developmental convenience but a phylogenetically earlier and more fundamental mode of intentional reference. Symbol use is a later, higher-cost layer built over a gestural substrate that is both evolutionarily older and cognitively more basic.

Lashley's foundational analysis of serial order in behaviour [9] provides a further dimension of this argument. The problem Lashley identified—that the surface sequential order of action cannot be the primary representational format, because sequences are planned and executed from a pre-sequential hierarchical structure—applies equally to communicative gesture and to keyboard input. In both cases, the temporal sequence is a projection of a deeper, non-sequential organisation.

2.3 The Shared Logical Structure

The developmental and evolutionary literature on gesture therefore provides independent empirical support for the core claim of this paper, arrived at from a completely different direction. Whether one begins from developmental observations of infants, comparative observations of enculturated apes, or the analysis of keyboard event streams, the same conclusion emerges: gesture is the primary representational object, and symbol is a downstream projection. The infant's point, the ape's reach, and the typist's slingshot arc are structurally analogous: each is a relational motor act that grounds and precedes the symbolic layer built over it.

This convergence is not coincidental. It reflects a general principle: any system capable of intentional reference will exhibit gesture before symbol because gestural acts are computationally and evolutionarily cheaper, more continuous with basic motor action, and more directly grounded in the relational structure of the environment. Symbolic communication is a compression of that gestural substrate, not its replacement.

3. The Existing Signal

3.1 Touch Typing as Structured Motor Behaviour

Touch typing is commonly described as a method for efficient text entry, but this description obscures the structure of the underlying motor activity. A skilled typist does not operate by selecting keys independently in sequence. Instead, typing is executed as a coordinated, distributed process across both hands, in which multiple fingers are active simultaneously and transitions are planned in advance of discrete key events.

Each finger operates within a constrained spatial domain defined by keyboard layout and learned motor habit. These domains are stable over time: the left pinky reliably services the set {`q, a, z`}, while the right pinky services {`p, ;, /`}. This partitioning is not merely instructional but becomes internalised as a persistent mapping between fingers and regions of action. Within these regions, movement is not discrete but continuous, even though the sensing apparatus reports only binary contact events.

Crucially, the fingers do not act independently. Skilled typing exhibits overlapping activation, in which multiple keys are depressed within short temporal intervals, often with one or more keys held while others are actuated. These overlaps are not accidental but arise from biomechanical efficiency: minimising unnecessary lift and repositioning allows for higher throughput and smoother motion [8].

The result is that, at any given moment, the typist's hands are in a composite state consisting of multiple active contacts, partial releases, and anticipatory positioning. This state evolves continuously over time, forming a trajectory that cannot be reduced to a simple sequence of independent actions.

3.2 Temporal Structure: Overlap, Duration, and Release

The temporal dimension of typing extends beyond the order of key presses. Each key activation has a duration, and the relative timing of presses and releases across fingers forms a structured temporal pattern. Empirically, three features are consistently observable:

- **Overlap windows.** Keys pressed within short intervals (on the order of tens of milliseconds) are often intended as part of a single coordinated gesture rather than independent events.
- **Sustained contact vs. pass-through.** Some keys are held briefly while others are engaged, creating layers of activation that persist across transitions.
- **Release ordering.** The order in which fingers disengage from keys carries structure, often reflecting the resolution of a movement or the completion of a local gesture.

These features are stable across repeated executions of the same word or phrase. A typist does not merely reproduce the same sequence of letters but tends to reproduce the same timing relationships between fingers [4, 5]. This temporal structure is directly

perceptible to the typist, manifesting as rhythm, flow, and coordination rather than as discrete symbolic decisions. Standard input systems ignore these features entirely, treating only the initial keydown events as meaningful.

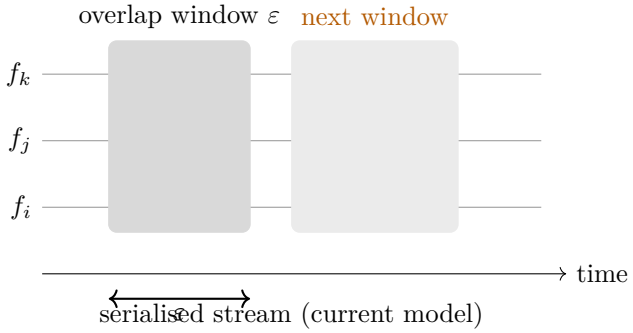


Figure 2: Simultaneity as a volumetric overlap region. Serialisation collapses each shaded volume into a linear sequence, destroying the equivalence structure $k_i \sim k_j \iff |t_i - t_j| < \varepsilon$.

3.3 Biomechanical Constraints and Hand Asymmetry

The structure of typing is further constrained by the biomechanics of the hands. The two hands do not contribute symmetrically; they exhibit distinct and complementary roles:

- **Anchoring.** One hand maintains a relatively stable position or sustained contact while the other executes more dynamic movement.
- **Alternation.** Rapid transitions between hands distribute load and reduce strain.
- **Convergence and divergence.** The hands move toward or away from each other depending on the spatial distribution of upcoming keys.

These patterns are not consciously controlled but emerge from physical constraints and the optimisation of movement over time. They are consistent across typists trained in standard touch-typing technique and are reinforced through practice. The result is that typing is not simply a collection of finger actions but a coordinated two-handed system in which the relationship between the hands encodes additional structure beyond what any single finger contributes.

3.4 The Key Switch as a Constraint Accumulator

A mechanical key switch is not a binary sensor in the ordinary sense. It is a continuous pressure device with a discrete output threshold. The actuation point—the position along the travel at which electrical contact is made—is a singular event in an otherwise continuous pressure curve. What the firmware receives as a "keydown" event is in reality the crossing of a constraint boundary by a continuous motor action.

This observation matters for the gesture framework because the *approach* to actuation carries information. The velocity of the finger at the moment of contact, the dwell time before actuation, and the rate of subsequent release all reflect properties of the

underlying gesture. Tactile and clicky switches make the actuation boundary physically perceptible, which is why experienced typists report different "feel" from different switch types—not because the switches change the symbolic output, but because they change the haptic feedback at the constraint boundary. In this sense, the key switch acts as a constraint accumulator: it integrates continuous pressure into a discrete event, discarding the approach curve in the process [11]. A gesture-aware capture layer would ideally record the crossing timestamp at sub-millisecond resolution alongside the event, preserving as much of the approach curve as the hardware permits.

3.5 Stability and Reproducibility of Gesture

The features described above—overlap, duration, and inter-hand coordination—are not incidental artefacts but form a stable component of skilled typing behaviour. A typist repeating a familiar word or phrase will tend to reproduce similar patterns of overlap between keys, similar durations of contact for specific fingers, and similar coordination patterns between the hands. This reproducibility suggests that the gesture underlying typing is learned as a coherent motor pattern, not reconstructed from discrete symbolic decisions each time [9]. In other domains, such as musical performance, such patterns are treated as primary objects of analysis and interpretation. In typing, they are systematically ignored.

3.6 Summary

The act of typing generates a structured signal with the following properties: concurrent activation across multiple fingers; continuous temporal variation in duration and release; hierarchical coordination within and between hands; and constraint-crossing events at the key switch level. This signal is stable, reproducible, and directly experienced by the typist. It exists independently of any symbolic interpretation and precedes the extraction of discrete characters. The subsequent sections show that current input systems do not merely simplify this signal but eliminate most of its structure, and that recovering it requires treating gesture—not symbol—as the primary object of representation.

4. What Current Systems Discard

The flattening performed by standard keyboard interfaces can be understood as a sequence of projections that systematically eliminate dimensions of the original motor signal. The result is a representation that is computationally convenient but information-theoretically impoverished relative to the gesture that produced it. This section isolates three dimensions of loss: simultaneity, temporal structure, and inter-hand relations.

4.1 Simultaneity and Its Collapse

Human typing routinely produces near-simultaneous key activations across multiple fingers. Keystrokes within a window on the order of tens of milliseconds are perceptually and motorically co-produced, forming what are effectively micro-chords. Standard input

systems do not preserve this simultaneity. Keyboard controllers scan key matrices sequentially and emit discrete keydown events that are timestamped and serialised by the operating system. When multiple keys are actuated within a short interval, their ordering in the resulting event stream is determined by scan order, interrupt timing, or driver behaviour rather than by any meaningful property of the gesture itself.

Formally, let $K = \{k_1, k_2, \ldots\}$ be the set of keydown events, each with timestamp t_i. The original gesture contains equivalence classes

$$k_i \sim k_j \iff |t_i - t_j| < \varepsilon$$

for some small ε representing simultaneity. The serialisation process replaces these classes with a total order:

$$k_{i_1} \prec k_{i_2} \prec \cdots \prec k_{i_n},$$

thereby destroying the equivalence relation. The information loss is structural: a partially ordered set is reduced to a linear order, and chordal or synchronous grouping becomes unrepresentable.

At the hardware level, this collapse is exacerbated by the key matrix rollover limit. Most consumer keyboards report at most six simultaneous key activations ("6KRO"), with the remainder silently dropped. This means the full combinatorial space of up to ten simultaneous contacts—the theoretical maximum for a two-handed touch typist—is not merely serialised but partially destroyed before the event stream reaches the operating system. Gaming and mechanical keyboards with true N-key rollover (NKRO) eliminate this truncation, but the serialisation of simultaneous events into a total order remains a software-level constraint regardless.

4.2 Duration and Release Structure

Each key event has a corresponding release, and the duration between onset and release encodes a second temporal signal largely independent of ordering. Conventional text input ignores this structure almost entirely: keyup events are detected at the hardware level but rarely utilised by higher-level software. Text entry is defined exclusively in terms of keydown order, and durations $\Delta t_i = t_i^{\uparrow} - t_i^{\downarrow}$ are discarded [3, 4]. This removes hold vs. pass-through distinctions, release ordering within a cluster, and temporal layering across fingers.

The discarding of keyup events is so complete that it is invisible to most developers. A standard event listener in any major programming environment exposes `keydown` and `keypress` as primary events, with `keyup` treated as a cleanup signal of no semantic significance. This is a design choice, not a physical constraint: the hardware reports both transitions with equal fidelity. The software stack then systematically discards half the available signal.

In the gesture formalism introduced below, duration and release are essential for identify-

ing pivots and segment boundaries; their removal forces segmentation to be reintroduced artificially via a dedicated spacebar rather than emerging from the physical gesture.

4.3 Inter-Hand Relations

The most consequential loss occurs at the level of inter-hand coordination. The two hands form a coupled system with stable, learnable patterns of interaction: convergence, divergence, asymmetric anchoring, and synchronisation. These patterns align with higher-level structure in both language and music, yet current input systems provide no representation of them. Because input is reduced to independent key events, there is no observable variable corresponding to

$$R_{LR}(t) = \text{relation}\big(H_L(t),\, H_R(t)\big),$$

where $H_L(t)$ and $H_R(t)$ are the aggregated states of the left and right hands. This collapse removes an entire layer of structure. While per-finger events remain observable in degraded form, the geometry of the hands as a coordinated system is rendered invisible.

4.4 The Spacebar as Vestigial Organ

A concrete illustration of the cost of discarding release structure is the spacebar. In a gesture-first system, word boundaries emerge naturally from the termination event $t_1 = \inf\{t \mid \forall i, \text{active}_i(t) = 0\}$: the moment all fingers disengage simultaneously. Full release is an unambiguous physical event that partitions the gesture stream into units without requiring an additional key.

The spacebar exists precisely because the serial symbol model cannot use this signal. Having discarded release structure, the system has no mechanism for detecting word boundaries and must reintroduce them as an explicit symbol. The spacebar is therefore not a fundamental input primitive but an artefact of the lossy compression performed upstream. In stenographic systems, which preserve chord structure, the stroke boundary plays an analogous role to full release; the steno writer never needs a dedicated spacebar [6].

4.5 Summary of Loss

The original signal includes concurrent activation across channels, continuous duration and release structure, and hierarchical coordination within and between hands. The retained signal includes only ordered keydown events. Certain structures—pivots defined by reversal under sustained contact, phrase boundaries defined by coordinated release—cannot be expressed within the serialised representation. They are not difficult to detect; they are unrepresentable.

5. Gesture and Canonical Representation

5.1 Finger Channels and Local Domains

Define the hand as a fixed set of channels $F = \{f_1, \ldots, f_8\}$, one per finger. Each finger f_i operates over a small local domain D_i (its constrained key region). At time t, each finger has a state

$$s_i(t) = \big(\text{active}_i(t),\ p_i(t),\ \dot{p}_i(t)\big),$$

where $\text{active}_i(t) \in \{0, 1\}$, $p_i(t)$ is the discrete or interpolated position within D_i, and $\dot{p}_i(t)$ is the inferred direction of migration. The full system state is $S(t) = \{s_1(t), \ldots, s_8(t)\}$.

5.2 Hierarchical Structure

The eight channels are not symmetric. They are organised into two hands:

$$H_L = \{f_1, f_2, f_3, f_4\}, \qquad H_R = \{f_5, f_6, f_7, f_8\}.$$

Hand-level states $S_L(t)$ and $S_R(t)$ are obtained by aggregating finger states within each hand, extracting the active support set, dominant direction of motion, and internal coherence (whether fingers are acting as a unit or independently). The inter-hand relation $R_{LR}(t) = \langle S_L(t), S_R(t) \rangle$ captures convergence, divergence, synchronisation, and role asymmetry $\text{role}(t) \in \{\text{left-anchor, right-anchor, co-active}\}$.

A gesture is therefore a time-indexed trajectory through this three-level structure:

$$G = \big\{\big(S_L(t),\ S_R(t),\ R_{LR}(t)\big) \mid t \in [t_0, t_1]\big\}.$$

5.3 Relational Manifold

Define derived relational features over G:

$$\begin{aligned}
A(t) &= \{i \mid \text{active}_i(t) = 1\} && \text{(support set)} \\
\text{sync}(i, j, t) &= \text{active}_i(t) \wedge \text{active}_j(t) && \text{(synchronisation)} \\
\text{pivot}_i(t) &\iff \dot{p}_i(t^-) \cdot \dot{p}_i(t^+) < 0 \wedge \text{active}_i(t) = 1 && \text{(reversal under contact)}
\end{aligned}$$

The termination event is $t_1 = \inf\{t \mid \forall i, \text{active}_i(t) = 0\}$. The gesture G is a trajectory through a low-dimensional relational manifold induced by F; it is not reducible to any sequence of key events.

6. Canonicalisation and Gesture Equivalence

6.1 The Canonicalisation Operator

Define a canonicalisation operator $\mathcal{C} : G \to \hat{G}$ such that small timing perturbations are discarded while overlap, pivot structure, and segmentation by full release are preserved. Formally, we impose a metric d on the raw gesture space such that if $d(G_1, G_2) < \delta$ then

$\mathcal{C}(G_1) = \mathcal{C}(G_2)$, where δ is a jitter tolerance threshold calibrated to the polling rate of the capture hardware. Two raw gestures G_1 and G_2 are equivalent if $\mathcal{C}(G_1) = \mathcal{C}(G_2)$. The canonical gesture $\hat{G}$ is the object the user actually learns.

Canonicalisation proceeds hierarchically:

$$\mathcal{C} = \mathcal{C}_{\text{inter-hand}} \circ \mathcal{C}_{\text{hand}} \circ \mathcal{C}_{\text{finger}}.$$

At the finger level, timing jitter is suppressed and local pivots are detected. At the hand level, dominant motion patterns are identified and fingers are clustered into coherent units. At the inter-hand level, convergence/divergence and anchor roles are extracted. The hierarchical composition can be visualised as a sequence of structure-preserving collapses, each removing distortion while retaining the invariant geometry of the gesture [14].

6.2 Topological Features

The canonical gesture preserves three topological features: **overlap** (which channels were simultaneously active), **pivot** (where reversals occurred under sustained contact), and **segmentation** (where full release divided one gesture from the next). These features are invariant under spatial distortion of the keyboard, temporal jitter, and changes to output mapping. They constitute the stable vocabulary the motor system learns.

6.3 Learnability Condition

A system is learnable if and only if the canonicalisation is stable under the distortions the user is likely to encounter. Formally:

$$\mathcal{C}(G_{\text{motor}}) = \hat{G}$$

must hold under spatial distortion of D_i, temporal jitter, and parameter variation in the output mapping. This ensures that motor equivalence classes coincide with canonical equivalence classes—the condition under which muscle memory transfers reliably.

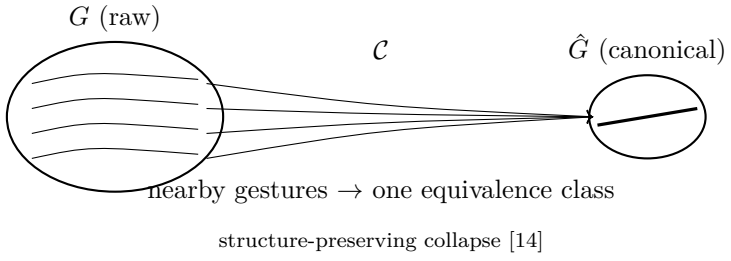


Figure 3: Canonicalisation as a collapse of nearby raw trajectories onto a single equivalence class. Distortion is removed; topological structure (overlap, pivot, segmentation) is preserved. The fiber-bundle intuition: each point in $\hat{G}$ has a fibre of equivalent raw gestures in G.

Reference: Gesture and Canonicalisation

Object	Definition
Finger state	$s_i(t) = (\text{active}_i(t),\ p_i(t),\ \dot{p}_i(t))$
System state	$S(t) = \{s_1(t), \ldots, s_8(t)\}$
Gesture	$G = \{S(t)\}_{t \in [t_0, t_1]}$
Support set	$A(t) = \{i \mid \text{active}_i(t) = 1\}$
Pivot	$\text{pivot}_i(t) \iff \dot{p}_i(t^-) \cdot \dot{p}_i(t^+) < 0 \ \wedge\ \text{active}_i = 1$
Termination	$t_1 = \inf\{t \mid \forall i,\ \text{active}_i(t) = 0\}$
Canonicalisation	$\mathcal{C} : G \to \hat{G}$, preserving overlap, pivot, segmentation
Learnability	$\mathcal{C}(G_{\text{motor}}) = \hat{G}$ stable under jitter and mapping variation

Key structural claim: $G \not\equiv$ sequence $\implies \hat{G} =$ primary object. Typing is a trajectory in a low-dimensional relational manifold; symbols are projections of that trajectory; canonicalisation extracts the stable topology of motor knowledge.

Figure 4: Reference sheet: gesture field objects and canonicalisation. Timing jitter is removed; overlap, pivot, and segmentation are preserved.

7. Gesture Grammar

7.1 Gesture Units and Slingshot Arcs

The canonical gesture decomposes into pivot-centred units:

$$\hat{G} = (\gamma_1, \gamma_2, \ldots, \gamma_n),$$

where each γ_k is a *slingshot arc*: a convergence phase, a pivot, and a divergence phase,

$$\gamma_k = (\text{approach},\ \text{pivot},\ \text{release}).$$

Pivot points are gesturally prominent: they are the moments of maximal constraint, where one or more fingers reverse direction while remaining in contact. They correspond naturally to syllabic peaks in speech and harmonic anchors in music.

The slingshot arc also provides a formal basis for why keystroke dynamics work as biometric identifiers [4, 5]. Biometric systems typically treat timing as a hidden feature; the present framework promotes it to a first-class structural component. The pivot density and arc shape of a gesture are as characteristic as a signature, and for the same reason: they reflect the compiled motor patterns of an individual user.

7.2 Composition

Gesture units compose by concatenation across release boundaries or by sharing a pivot. The word *machine*, for instance, decomposes into three slingshot arcs: `m`↔`a`, `a`↔`ch`, and `i`↔`n`↔`e`, where each pivot letter serves simultaneously as the endpoint of one arc and the origin of the next. This decomposition is not imposed on the gesture but emerges from its physical structure.

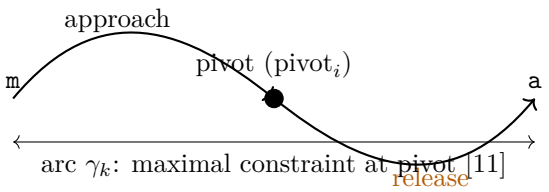


Figure 5: Slingshot arc $\gamma_k = (\text{approach}, \text{pivot}, \text{release})$. The pivot is the moment of maximal constraint: fingers reverse direction while remaining in contact. This is not a symbolic event but a bodily one [13].

7.3 Anchor–Pivot Asymmetry

A refinement of the slingshot arc concerns the role asymmetry between hands. In many gestures, one hand acts as an anchor—maintaining a stable or slowly evolving position—while the other hand executes the high-frequency movement that carries most of the local information. The anchor hand provides the *context* within which the pivot hand's movements are interpreted. Formalising this:

$$\gamma_k = \big(\text{anchor}(H_L \text{ or } H_R),\ \text{trajectory}(H_{\neg\text{anchor}}),\ \text{pivot}\big)$$

This structure is directly analogous to the relationship between bass and melody in tonal music: the anchoring voice establishes harmonic context, and the melodic voice moves within it [13]. It also explains why some word-initial chords feel natural and others do not—the hand assignment at the onset of each arc constrains what subsequent movements are biomechanically smooth.

8. Projection Framework

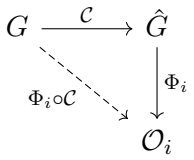
8.1 Projection Functors

Let $\hat{\mathcal{G}}$ denote the space of canonical gestures. Define projection functors

$$\begin{aligned}\Phi_{\text{text}} &: \hat{\mathcal{G}} \to \Sigma^* \\ \Phi^{\theta}_{\text{music}} &: \hat{\mathcal{G}} \to \mathcal{M} \\ \Phi_{\text{visual}} &: \hat{\mathcal{G}} \to \mathcal{V}\end{aligned}$$

where Σ^* is the space of strings, $\mathcal{M}$ is a pitch-time structure, and $\mathcal{V}$ is a space of visual forms. Each functor depends only on $\hat{G}$, not on the raw event stream.

The following commutative diagram illustrates the relationship between the raw gesture, canonicalisation, and projection:



The dashed arrow confirms that whether one first canonicalises and then projects, or composes both operations directly from the raw stream, the result is the same output $\mathcal{O}_i$. This commutativity is the formal expression of the claim that canonical gesture is the correct intermediate representation: it does not lose information that matters for projection.

8.2 Metric Independence

Let θ be a mapping parameter (pitch range, tuning, text dictionary). Then $\Phi^{\theta}_{\text{music}}(\hat{G})$ varies with θ while $\hat{G}$ remains fixed. This formalises the design principle: the mapping layer may change freely; the gesture grammar must remain invariant. A single learned motor vocabulary can project into multiple modalities without relearning.

8.3 Core Claim

Meaning is a function of gesture equivalence classes, not of symbol sequences:

$$\text{meaning} : \hat{\mathcal{G}} \to \text{output domain}.$$

Text, music, and visual form are projections of a shared underlying structure. The gesture is the primary object; the projections are secondary.

9. Admissibility as a Meta-Constraint Structure

The projection framework defines mappings from canonical gesture space to output domains. A projection specifies *how* a gesture is rendered; but it does not specify *whether* the result is structurally valid. This distinction motivates a further layer: admissibility.

9.1 The Admissibility Functional

Define an admissibility functional

$$\mathcal{A} : \hat{\mathcal{G}} \times \{\Phi_i\} \to \mathbb{R}_{\geq 0}$$

A projected output is admissible if

$$\mathcal{A}(\hat{G}, \Phi_i) > \tau$$

for some threshold τ. Admissibility captures conditions such as: whether a gesture closes coherently under projection (e.g., a gesture that maps to a fragment of a word rather than a complete word has low admissibility under Φ_{text}); whether the projected output is

globally consistent (e.g., a musical phrase that violates the current harmonic constraint has low admissibility under Φ_{music}); and whether the equivalence class is unambiguous (e.g., a gesture that maps to multiple equally plausible words has reduced admissibility because the projection is not injective at that point).

9.2 The Admissibility Log

Rather than treating admissibility as a static predicate, we define the admissibility log as the temporal trace of this evaluation:

$$\mathcal{L} = \{(\hat{G}_t,\ \Phi_i,\ \mathcal{A}_t,\ \epsilon_t)\}_t$$

where $\hat{G}_t$ is the canonical gesture at time t, Φ_i is the active projection, $\mathcal{A}_t$ is the admissibility score, and ϵ_t is an obstruction term encoding failure of closure. The obstruction captures incompatibility between local gesture segments under projection, failure of global consistency (non-gluable outputs), and ambiguity across equivalence classes.

9.3 Non-Reductive Character

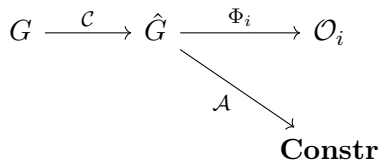
Unlike the projection functors Φ_i, which reduce the dimensionality of the gesture representation, the admissibility structure is non-reductive: it introduces additional structure rather than collapsing it, encodes global constraints across time, and tracks the history of constraint satisfaction and failure. It is therefore better understood as a functor into a category of constraint-evaluated objects rather than into an output domain:

$$\mathcal{A} : \hat{\mathcal{G}} \to \mathbf{Constr}$$

where objects in **Constr** are pairs $(\hat{G}, \Phi_i)$ equipped with coherence conditions.

9.4 Extended Diagram

The full three-stage structure can be represented as:



Canonicalisation identifies what the gesture *is*; projection specifies how it is *rendered*; admissibility evaluates whether the rendering is *valid*. Together they form a complete pipeline from raw motor input to validated symbolic or musical output.

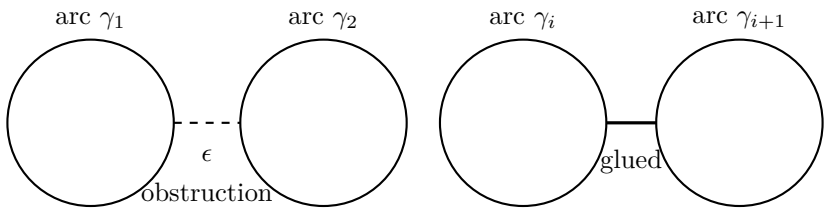


Figure 6: Failure of local gesture arcs to glue into a globally admissible structure (left) versus successful gluing (right). The obstruction ϵ is not a local error but a global topological invariant.

Reference: Projection and Admissibility

Object	**Definition**
Projection functor	$\Phi_i : \hat{\mathcal{G}} \to \mathcal{O}_i$ (text Σ^*, music $\mathcal{M}$, visual $\mathcal{V}$)
Admissibility	$\mathcal{A} : (\hat{G}, \Phi_i) \to \mathbb{R}_{\geq 0}$; admissible iff $\mathcal{A}(\hat{G}, \Phi_i) > \tau$
Admissibility log	$\mathcal{L} = \{(\hat{G}_t, \Phi_i, \mathcal{A}_t, \epsilon_t)\}_t$
Obstruction	$\epsilon \in H^1(\hat{\mathcal{G}}, \mathcal{F})$: non-trivial cohomology class
Metric independence	$\Phi^{\theta}_{\text{music}}(\hat{G})$ varies with θ; $\hat{G}$ invariant
Commutativity	$\Phi_i \circ \mathcal{C}$ depends only on $\hat{G}$, not raw stream

Two-layer structure: projection reduces dimensionality; admissibility adds structure and records constraint history. **Sheaf interpretation:** global coherence exists iff $\epsilon = 0$; failure of meaning $\equiv$ non-gluable local structure.

Figure 7: Reference sheet: projection functors, admissibility functional, and obstruction.

10. Amplitwist Cascades: Recursive Geometric Evaluation

The admissibility framework introduced above defines a constraint layer over canonical gestures and their projections. It does not yet specify the *internal geometry* through which a gesture is evaluated prior to projection. This section introduces the amplitwist cascade as the geometric mechanism underlying that evaluation [14].

The central claim is that a canonical gesture $\hat{G}$ does not map directly to an output domain. Instead, it induces a trajectory in an epistemic manifold M equipped with RSVP fields $(\Phi, \vec{v}, S)$, and is evaluated through a sequence of recursive deformations of that manifold. The cascade is not a pipeline between different spaces; it is progressive warping of the same manifold under accumulated Lie action.

10.1 The Base Amplitwist Operator

On the RSVP manifold M, let $\Phi : M \to \mathbb{R}$ be the scalar constraint-density field, $\vec{v} : M \to TM$ the plenum velocity field, and $S : M \to \mathbb{R}_+$ the entropy field. Define the base amplitwist operator:

$$\mathcal{A}(\vec{x}) = \|\vec{v}(\vec{x})\| \cdot \exp\Big(i \cdot \arccos\Big(\frac{\vec{v}(\vec{x}) \cdot \nabla\Phi(\vec{x})}{\|\vec{v}(\vec{x})\| \, \|\nabla\Phi(\vec{x})\| + \varepsilon} \Big) \Big),$$

where $\varepsilon > 0$ ensures numerical stability. This operator decomposes into two geometrically distinct components:

$$\rho(\vec{x}) = \|\vec{v}(\vec{x})\| \qquad \text{(amplitude: intensity of conceptual flow)}$$
$$\theta(\vec{x}) = \arccos\left(\frac{\vec{v} \cdot \nabla\Phi}{\|\vec{v}\|\,\|\nabla\Phi\| + \varepsilon}\right) \qquad \text{(phase: alignment with semantic gradient)}$$

The amplitwist encodes a local rotation and scaling in the tangent space of M, generalising the complex derivative interpretation of conformal maps [14] to the RSVP setting.

10.2 Recursive Lie-Algebraic Deformations

Let $\mathfrak{so}(n)$ denote the Lie algebra of skew-symmetric transformations on M. Each semantic layer introduces an infinitesimal deformation $T_j \in \mathfrak{so}(n)$ scaled by $\epsilon_j \in \mathbb{R}$. Define the accumulated deformation:

$$\mathfrak{R}_k(\vec{x}) = \vec{x} + \sum_{j=1}^{k} \epsilon_j T_j(\vec{x}).$$

This is not a sequence of mappings between distinct spaces but a progressive warping of M that preserves differentiable structure. The composition $\mathfrak{R}_k$ defines a flow on M generated by $\mathfrak{so}(n)$:

$$M \;\mapsto\; \mathfrak{R}_1(M) \;\mapsto\; \cdots \;\mapsto\; \mathfrak{R}_k(M).$$

10.3 Layered Amplitwist Evaluation

The amplitwist at layer k is evaluated at the deformed point $\mathfrak{R}_k(\vec{x})$ with entropy weighting:

$$\mathcal{A}^{(k)}(\vec{x}) = w_k(\vec{x}) \cdot \mathcal{A}(\mathfrak{R}_k(\vec{x})), \qquad w_k(\vec{x}) = e^{-\lambda S(\vec{x})}.$$

Deformation accumulates forward; entropy suppresses amplitude multiplicatively. The cascade $\mathcal{A}^{(0)}, \mathcal{A}^{(1)}, \ldots, \mathcal{A}^{(k)}$ is a recursive evaluation over a continuously deformed manifold in which amplitude, phase, and entropy jointly determine the stability of interpretation.

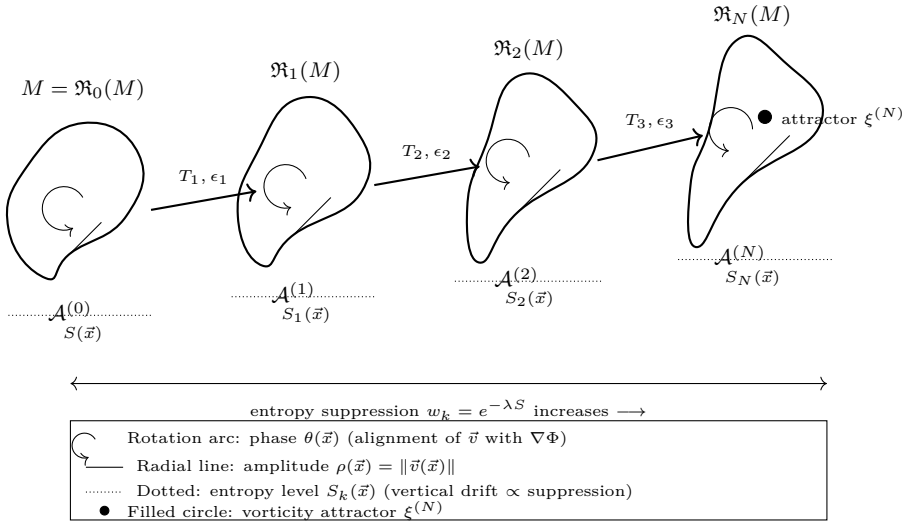


Figure 8: Amplitwist cascade: recursive deformation of manifold M under Lie-generated flows $\mathfrak{R}_k$, with layer-k amplitwist $\mathcal{A}^{(k)}(\vec{x}) = w_k(\vec{x})\,\mathcal{A}(\mathfrak{R}_k(\vec{x}))$. Rotation arcs encode phase θ; radial lines encode amplitude ρ; dotted lines mark entropy levels; vertical drift encodes suppression. The filled circle marks the emergent attractor $\xi^{(N)}$ [11].

10.4 Emergent Vorticity and Attractors

Let $\hat{\vec{v}}^{(N)}$ denote the velocity field transported through the cascade. Define the emergent vorticity:

$$\xi^{(N)}(\vec{x}) = \|\nabla \times \hat{\vec{v}}^{(N)}(\vec{x})\|.$$

Regions of high vorticity are stable attractors in the epistemic manifold—points at which recursive deformation and amplitwist alignment converge. A gesture induces a trajectory whose cascade evaluation either converges to an attractor (high admissibility) or disperses (low admissibility). Admissibility can therefore be reinterpreted as:

$$\mathcal{A}(\hat{G}, \Phi_i) \;\sim\; \text{stability of } \mathcal{A}^{(N)} \text{ under cascade flow.}$$

Theorem 3.1 of the RSVP programme guarantees that $\xi^{(N)}$ converges when $\epsilon_j < \epsilon_{\text{crit}}$ [11].

10.5 Integration with Gesture and Projection

The gesture-first framework defines:

$$G \xrightarrow{\mathcal{C}} \hat{G} \xrightarrow{\Phi_i} \mathcal{O}_i.$$

The amplitwist cascade inserts an internal geometric evaluation stage:

$$\hat{G} \;\longrightarrow\; \mathfrak{R}_k(\hat{G}) \;\longrightarrow\; \mathcal{A}^{(k)} \;\longrightarrow\; \Phi_i.$$

Canonical gesture $\hat{G}$ specifies initial conditions on M; recursive deformation $\mathfrak{R}_k$ defines the trajectory in epistemic space; amplitwist $\mathcal{A}^{(k)}$ evaluates local alignment and intensity;

projection Φ_i renders the resulting structure. Admissibility then operates over the entire cascade:

$$\mathcal{A}(\hat{G}, \Phi_i) \;\equiv\; \text{coherence}(\{\mathcal{A}^{(k)}\}_k).$$

The key objects of the cascade are summarised in Table 1.

Object	**Definition**	**Role**
Base operator	$\mathcal{A}(\vec{x}) = \|\vec{v}\| \exp(i\theta)$	Amplitude from velocity; phase from alignment
Recursive layer	$\mathfrak{R}_k = \vec{x} + \sum_j \epsilon_j T_j(\vec{x})$, $T_j \in \mathfrak{so}(n)$	Lie-algebraic deformation
Layer-k amplitwist	$\mathcal{A}^{(k)} = w_k \, \mathcal{A}(\mathfrak{R}_k)$, $w_k = e^{-\lambda S}$	Entropy-weighted evaluation
Vorticity	$\xi^{(N)} = \|\nabla \times \hat{\vec{v}}^{(N)}\|$	Emergent attractor structure

Table 1: Key objects of the amplitwist cascade.

10.6 Conceptual Closure

The amplitwist cascade provides a unifying interpretation of the framework developed in this paper. It replaces static representation with recursive deformation: meaning is not read off from a gesture but emerges through successive geometric transformations. Entropy suppresses amplitwist magnitude, biasing the system toward low-energy attractors. Admissibility reduces to geometric stability: a projection is valid if and only if the cascade converges to a stable attractor.

The cascade integrates with the broader RSVP programme: the scalar, vector, and entropy fields $(\Phi, \vec{v}, S)$ define the local geometry; the amplitwist operator encodes alignment within that geometry; and the cascade describes how local structure propagates across scales. It is not an additional component of the system but the internal geometry through which gesture, projection, and admissibility become coherent [11, 14, 13].

Interpretation is not a mapping from symbol to meaning, but a trajectory through a recursively deformed geometric field whose stable attractors constitute what is recognised as meaning.

11. CLIO as Gradient Flow on the Amplitwist Cascade

CLIO (Cognitive Loop via In-Situ Optimization) is a framework introduced by Cheng, Broadbent, and Chappell [15] in which large language models self-formulate approaches to problems, adapt when self-confidence is low, and allow external observation and correction of internal reasoning states. The key features of CLIO—open uncertainty representation, graph-structured belief states, and scientist interjectability—align structurally with the

gesture-first framework developed in this paper. In the gesture setting, the "scientist" is the user, the "reasoning process" is the amplitwist cascade, and "steerability" is precisely the intervention parameter Δt_{ctrl} of Section 17.

We reinterpret CLIO here as a gradient flow on the cascade energy, connecting its adaptive self-confidence mechanism to the geometric stability of the amplitwist attractor. The original CLIO paper operates over LLM belief states; the present interpretation generalises this to any recursive geometric evaluation system, of which gesture-driven projection is a specific instance.

The amplitwist cascade defines the geometry of interpretation. It does not yet specify how the system *evolves* toward admissibility. This role is fulfilled by the gradient flow on the cascade energy.

11.1 Energy Functional

Let $\{\mathcal{A}^{(k)}(\vec{x})\}_{k=0}^{N}$ be the amplitwist cascade. Define the cascade energy:

$$\mathcal{E}(\vec{x}) = \sum_{k=0}^{N} \alpha_k \left(\|\nabla\theta^{(k)}(\vec{x})\|^2 + \beta \, \|\nabla\rho^{(k)}(\vec{x})\|^2 \right) + \mu S(\vec{x}),$$

where $\rho^{(k)}$ and $\theta^{(k)}$ are the amplitude and phase of $\mathcal{A}^{(k)}$, α_k weights cascade layers, β balances amplitude vs. phase smoothness, and μ penalises entropy. This functional measures incoherence across the cascade: rapid spatial variation in phase or amplitude corresponds to misalignment; high entropy corresponds to instability.

11.2 Gradient Flow Dynamics

The gradient flow on $\mathcal{E}$ is:

$$\frac{d\vec{x}}{dt} = -\nabla\mathcal{E}(\vec{x}).$$

This implements a continuous repair process: given a gesture $\hat{G}$, the system evolves its position in the epistemic manifold until the amplitwist cascade becomes stable. It moves toward regions of aligned phase (low $\|\nabla\theta\|$), smooths amplitude variation (low $\|\nabla\rho\|$), and descends toward lower entropy (low S).

11.3 Relation to Control Resolution

The control resolution parameter Δt_{ctrl} determines how this flow is realised. In high-intervention regimes ($\Delta t_{\text{ctrl}} \to 0$), the user perturbs $\vec{x}$ directly during the flow—analogous to the scientist interjectability feature of CLIO [15]. In low-intervention regimes, the system integrates the flow to a fixed point before exposing the result. Both regimes follow the same gradient flow but differ in how frequently external intervention modifies the trajectory.

11.4 Fixed Points and Admissibility

A point $\vec{x}^*$ is a fixed point if $\nabla\mathcal{E}(\vec{x}^*) = 0$. These correspond to stable attractors of the amplitwist cascade. A gesture $\hat{G}$ is admissible if its induced trajectory under this flow converges to a fixed point:

$$\text{admissibility} \iff \text{convergence of gradient flow.}$$

Admissibility is therefore not a binary predicate but a dynamical property of the system.

12. Amplitwist Loss as an Admissibility Functional

The admissibility functional $\mathcal{A}(\hat{G}, \Phi_i)$ has thus far been defined abstractly. The amplitwist cascade provides a natural instantiation in terms of cross-system alignment.

12.1 Definition of Amplitwist Loss

Let $\mathcal{A}^{(k)}_{\text{sys}}$ denote the cascade induced by a system and $\mathcal{A}^{(k)}_{\text{ref}}$ a reference cascade (human gesture dynamics or a canonical semantic field). Define:

$$\mathcal{L}_A = \sum_{k=0}^{N} \left\| \mathcal{A}^{(k)}_{\text{sys}}(\vec{x}) - \mathcal{A}^{(k)}_{\text{ref}}(\vec{x}) \right\|^2.$$

This measures discrepancy at the level of recursive geometric evaluation, not final output.

12.2 Decomposition by Amplitude and Phase

Expanding $\mathcal{A}^{(k)} = \rho^{(k)} e^{i\theta^{(k)}}$:

$$\mathcal{L}_A = \sum_{k=0}^{N} \left(\|\rho^{(k)}_{\text{sys}} - \rho^{(k)}_{\text{ref}}\|^2 + \gamma \|\theta^{(k)}_{\text{sys}} - \theta^{(k)}_{\text{ref}}\|^2 \right),$$

where γ balances amplitude and phase alignment. This separates two failure modes: **amplitude mismatch** (incorrect intensity) and **phase mismatch** (misalignment with semantic gradients).

12.3 Admissibility as Bounded Loss

Admissibility is now quantitative:

$$\mathcal{A}(\hat{G}, \Phi_i) > \tau \iff \mathcal{L}_A < \varepsilon.$$

A projection is admissible not because it satisfies symbolic constraints but because its recursive geometric evaluation aligns with the reference cascade.

12.4 Relation to CLIO

The gradient flow and the loss are dual:

$$\nabla \mathcal{E} \to 0 \;\Rightarrow\; \mathcal{L}_A \to 0.$$

Convergence of internal dynamics produces alignment with the reference cascade. Together, CLIO (internal repair) and $\mathcal{L}_A$ (external alignment) complete the admissibility structure: the cascade defines the geometry of interpretation, the gradient flow defines the dynamics of repair, and the amplitwist loss defines the criterion of alignment.

13. Variational Principle for the Amplitwist Cascade

The amplitwist cascade can be derived from a variational principle. Rather than treating the recursive operators and amplitwists as definitional, we regard them as extrema of an action functional.

13.1 Cascade Energy

Define:

$$\mathcal{E}_{\text{cas}} = \sum_{k=0}^{N} \int_M \left[\alpha_k \|\nabla \mathcal{A}^{(k)}\|^2 + \beta_k \|\mathcal{A}^{(k)} - \mathcal{A}^{(k-1)}\|^2 + \mu_k S |\mathcal{A}^{(k)}|^2 \right] d\mu,$$

(with the second term omitted for $k = 0$). Writing $\mathcal{A}^{(k)} = \rho^{(k)} e^{i\theta^{(k)}}$, the gradient term decomposes as:

$$\|\nabla \mathcal{A}^{(k)}\|^2 = \|\nabla \rho^{(k)}\|^2 + (\rho^{(k)})^2 \|\nabla \theta^{(k)}\|^2.$$

The three terms penalise local roughness, layer discontinuity, and entropy-weighted amplitude respectively.

13.2 Full Action

$$\mathcal{S}_{\text{amp}} = \int_{t_0}^{t_1} \left(\mathcal{E}_{\text{cas}} + \mathcal{E}_{\text{RSVP}} + \mathcal{E}_{\text{adm}} \right) dt,$$

where the RSVP field energy is:

$$\mathcal{E}_{\text{RSVP}} = \int_M \left[a_\Phi \|\nabla \Phi\|^2 + a_v \|\nabla \vec{v}\|^2 + a_S \|\nabla S\|^2 + b\, S \|\vec{v}\|^2 \right] d\mu,$$

and the admissibility energy penalises obstruction:

$$\mathcal{E}_{\text{adm}} = \int_M \|\epsilon(\vec{x})\|^2 \, d\mu.$$

13.3 Stationary Condition

The realised interpretation is obtained by extremising the action:

$$\delta \mathcal{S}_{\text{amp}} = 0,$$

which yields stationarity conditions with respect to Φ, $\vec{v}$, S, and T_j simultaneously. These express the same principle at four levels: scalar salience, vector flow, entropy modulation, and recursive deformation must jointly stabilise.

13.4 Central Identity

The variational formulation gives the cascade a field-theoretic status: meaning is not a lookup but a constrained relaxation. The core principle of the paper can now be stated as a single equation:

$$\boxed{\text{meaning} = \arg\operatorname*{ext}_{X,\{T_j\}} \mathcal{S}_{\text{amp}}[X, \{T_j\}]}$$

Meaning is the stationary structure of an entropy-weighted, gesture-driven amplitwist cascade [11, 14].

Reference: Control, CLIO, and Amplitwist Optimisation

Object	**Definition**
Control resolution	Δt_{ctrl}: minimum interval for gesture to modify $x(t)$
Tool regime	$\Delta t_{\text{ctrl}} \to 0$: continuous coupling, external repair
Agent regime	large Δt_{ctrl}: single-step mapping, internal repair
CLIO energy	$\mathcal{E} = \sum_k \alpha_k(\|\nabla\theta^{(k)}\|^2 + \beta\|\nabla\rho^{(k)}\|^2) + \mu S$
Gradient flow	$d\vec{x}/dt = -\nabla\mathcal{E}(\vec{x})$ (CLIO [15] as descent)
Fixed point	$\nabla\mathcal{E} = 0 \Rightarrow$ stable interpretation
Amplitwist loss	$\mathcal{L}_A = \sum_k \|\mathcal{A}_{\text{sys}}^{(k)} - \mathcal{A}_{\text{ref}}^{(k)}\|^2$
Admissibility	$\mathcal{L}_A < \varepsilon \iff$ admissible
Variational identity	$\text{meaning} = \arg\operatorname{ext}_{X,\{T_j\}} \mathcal{S}_{\text{amp}}[X, \{T_j\}]$

System closure: $\hat{G} \to$ cascade $\to$ CLIO flow $\to$ fixed point $\to \Phi_i$. CLIO provides internal repair; $\mathcal{L}_A$ provides external alignment; Δt_{ctrl} determines coupling. **Key principle:** Meaning = stable fixed point of constrained geometric flow.

Figure 9: Reference sheet: control resolution, CLIO gradient dynamics, amplitwist loss, and the variational identity.

Unified RSVP–Amplitwist–Admissibility System

Layer	Content
RSVP base fields	$X = (\Phi, \vec{v}, S)$: scalar potential, vector flow, entropy
Gesture coupling	$\hat{G} \to \vec{v}(\vec{x}, t)$: gesture as boundary condition on flow
Amplitwist	$\mathcal{A}(\vec{x}) = \|\vec{v}\|e^{i\theta}$: local rotation-scaling in TM
Cascade	$\mathcal{A}^{(k)} = e^{-\lambda S}\mathcal{A}(\mathfrak{R}_k)$: entropy-weighted recursive deformation
Vorticity	$\xi^{(N)} = \|\nabla \times \hat{v}^{(N)}\|$: emergent attractor structure
Projection	$\Phi_i : \hat{\mathcal{G}} \to \mathcal{O}_i$: rendering into output domain
Admissibility	$\mathcal{A}_{\text{adm}} \sim \int \|\nabla\theta^{(N)}\|^2 + \mu S$: coherence measure
Constraint closure	$\epsilon = 0 \iff$ global coherence

Full system flow: $\hat{G} \to \vec{v} \to \mathcal{A}^{(k)} \to \xi^{(N)} \to \Phi_i \to \mathcal{A}_{\text{adm}}$.

Core identity:

$$\text{Meaning} = \text{entropy-weighted stable flow structure} = \underset{X,\{T_j\}}{\arg\operatorname{ext}}\, \mathcal{S}_{\text{amp}}$$

Figure 10: Unified reference: RSVP fields, amplitwist cascade, gesture input, and admissibility as a single constrained dynamical system.

14. Hardware and System Constraints

14.1 Keyboard Matrix Limitations

Standard keyboards scan key matrices at rates of approximately one to eight kilohertz, with hardware debounce filters that introduce latency on the order of milliseconds. N-key rollover is absent on many consumer devices, limiting the number of simultaneously reported keys to six or fewer. These constraints mean that concurrent gesture structure is collapsed at the point of capture, before any software layer has an opportunity to interpret it.

The USB HID protocol, which governs how keyboards communicate with host systems, was designed around a 125 Hz polling rate (8 ms per report), though USB high-speed mode supports 1000 Hz (1 ms). Most operating systems default to 125 Hz unless explicitly configured otherwise. At this polling rate, two keystrokes separated by less than 8 ms are indistinguishable from simultaneous. The simultaneity equivalence window ε in our formalism is therefore bounded below by the polling period: any implementation must account for this hardware floor when calibrating δ in the canonicalisation metric.

14.2 Required Capabilities

Capturing the gesture signal described in this paper does not require a new form factor. It requires: true N-key rollover across all keys; high-resolution timestamps on keydown and keyup events (sub-millisecond resolution); and access to raw keyup events at the application level. These capabilities exist in current gaming and mechanical keyboard hardware with QMK-compatible controllers (such as the Teensy series running at 1000 Hz polling), but are not exposed by standard input pipelines.

Developers working in this space should note that `evdev` on Linux provides raw key event timestamps at kernel-level resolution and exposes both keydown and keyup events with equal fidelity. The `libevdev` C library and `python-evdev` wrapper provide direct access to this stream without the semantic filtering imposed by higher-level input frameworks. On macOS, the `CGEventTap` interface provides analogous low-level access. On Windows, `SetWindowsHookEx` with `WH_KEYBOARD_LL` captures low-level keyboard events including keyup with system timestamps, though the timestamp resolution depends on the timer resolution set via `timeBeginPeriod`.

14.3 Minimal Instrument Specification

A gesture-capable input layer requires: a capture module delivering raw timestamped keydown and keyup events; a channel mapping from physical keys to finger identities; a feature extractor computing overlap, duration, direction, and inter-hand relations; a canonicalisation pipeline implementing $\mathcal{C}$; and a pluggable projection layer implementing Φ_i. No new hardware is required. The instrument is a software layer that reads what the hands are already producing.

15. Relationship to Existing Systems

Existing input systems can be situated within the gesture framework by examining how much of the relational signal each preserves. A useful prior step is to observe that these systems differ along two independent axes: *continuous vs. symbolic* (whether the signal is a spatial path or a discrete command element) and *low-dimensional vs. high-dimensional* (the size of the space the input navigates). Trace keyboards are continuous and low-dimensional: the finger draws a path through a flat letter grid [2]. Symbolic systems such as Vim, Bash, or Emacs are discrete and high-dimensional: each keypress or chord is an element in a large combinatorial space. These two axes are orthogonal, and the positions systems occupy on them predict which dimensions of gesture they preserve and which they discard.

A second structural observation concerns the relationship between chorded and sequential input. Chords and key sequences are not categorically different; they are dual encodings of the same underlying structure—a prefix tree, or trie, over possible intentions (i.e., a hierarchical decision structure in which each node narrows the space of valid next actions). A chord selects a node in that tree in parallel; a sequence walks a path to the same node one step at a time. Systems such as Spacemacs make this equivalence explicit: the binding `SPC f s` (space, file, save) is a sequential traversal of the same hierarchy that a single chord might compress into simultaneity. GUI menus, shell directory traversal (`cd /usr/local/bin`), and chorded keybindings are all encodings of hierarchical navigation; what differs is whether the tree is externalised in the display or internalised in motor memory. The `which-key` plugin in Emacs and Spacemacs is precisely a mechanism for rendering the implicit trie visible on demand—collapsing the tree back into a menu when the user needs to explore rather than execute. This trie structure is also present in the

gesture grammar of Section 6: pivot-centred arcs are the nodes, and composition by shared pivot is traversal through the tree.

Standard typing collapses the gesture entirely to ordered keydown events: $G \to \{k_{i_1} \prec k_{i_2} \prec \cdots\}$ [7]. Stenography preserves simultaneity within a stroke but discards duration and release ordering [6]. Swype-style trace keyboards preserve single-finger spatial trajectory but discard multi-finger parallelism and inter-hand structure [2]. Musical instruments preserve duration and timing but treat inter-finger coordination as implicit in the acoustic output rather than as an explicit signal. General gesture recognisers have explored trajectory-based representations at the level of single-pointer input [1], but without the multi-channel relational structure described here.

No existing system treats inter-hand relational structure as a primary signal. The present framework is distinguished by defining meaning at the level of inter-channel coordination rather than individual channel events.

The developmental and evolutionary literature reviewed in Section 2 provides a further axis of comparison. Existing input systems, without exception, assume the symbol as primitive and treat gesture as an encoding problem: how do we get from motor action to symbolic output as efficiently as possible? The gesture-before-symbol framework inverts this. In developmental terms, input system design has been proceeding as though children learn to point by first learning words—when in fact the arrow runs the other way. The correct design question is not "how do we recognise the symbol the user intends?" but "how do we preserve enough of the gestural structure that the symbol can be unambiguously projected from it?" [16, 17].

16. Implications

If gesture is the primary object, several consequences follow. First, a single learned motor vocabulary can project simultaneously into text, music, and visual form without requiring separate training for each modality—the same gesture that types a word produces a musical phrase and a visual contour. Second, skill transfer between typing and musical performance becomes structurally grounded rather than merely analogical: both exploit the same hierarchical relational structure [8]. This explains why some people find the transition between piano and keyboard intuitive: the underlying relational manifold is the same; only Φ is swapped. Third, the design space for input systems expands from symbol-entry optimisation to gesture-representation fidelity, shifting the engineering question from *how do we recognise keystrokes faster?* to *how much of the gesture do we preserve?* Isomorphic keyboard designs, which seek layout-invariant chord shapes [10], represent one existing step in this direction, though they remain committed to fixed pitch mappings rather than the dynamic projection framework described here.

A fourth consequence bears on how intelligence is distributed in human-computer systems. Much current discourse assumes that intelligence must be centralised in a general-purpose agent—a model that interprets inputs, reasons over them, and produces outputs. The

gesture-first framework suggests a different picture. If a well-designed interface preserves the relational structure of the user's intention and projects it faithfully into multiple domains, then a large fraction of what is normally attributed to the intelligence of the *system* is better understood as intelligence of the *coupling*. Tools like Vim, Bash, or a well-designed trace keyboard are not intelligent in the agent sense; they are intelligent in the sense that they preserve and amplify the structure the user brings. The keyboard in this framework is not a switchboard but a high-bandwidth sensor whose full signal has never been read. The present framework extends this principle: by treating gesture as primary and projection as secondary, it locates expressive power in the trained motor system of the user rather than in any centralised interpreter. The interface becomes not a black box that translates intention into output, but a transparent medium that preserves the structure of gesture across modalities.

17. Intervention and Control: Tools and Agents as a Continuum

The distinction between "tools" and "agents" is commonly framed in terms of autonomy. Within the gesture-first framework, this distinction can be expressed more precisely as a property of the *control loop* linking gesture, projection, and admissibility.

Given a canonical gesture $\hat{G}$ and a projection Φ_i, the system produces an output trajectory $x(t)$. The admissibility functional $\mathcal{A}$ evaluates this trajectory and yields an obstruction signal ϵ_t when coherence fails. The critical question is how this obstruction is resolved—and by whom.

17.1 External Repair (High-Intervention Systems)

In high-intervention systems, the repair loop remains external. The user observes ϵ_t and updates the gesture:

$$\hat{G}_{t+1} = \text{repair}(\hat{G}_t,\ \epsilon_t).$$

Projection is applied incrementally, and admissibility is enforced through continuous user intervention. The system dynamics are tightly coupled to gesture:

$$x(t + \delta t) = F(x(t),\ \hat{G}_{[t,\, t+\delta t]}).$$

Editors, shells, and direct manipulation interfaces operate in this regime. The user retains control over the trajectory at fine temporal resolution, and admissibility is achieved through iterative refinement rather than autonomous internal computation.

17.2 Internal Repair (Low-Intervention Systems)

In low-intervention systems, the repair loop is internalised. A gesture $\hat{G}$ is treated as an initial condition, and the system applies an internal process that attempts to resolve admissibility without further input:

$$x(t_1) = F(x(t_0),\ \hat{G}).$$

The user observes only the final outcome and may intervene only by aborting or restarting. Large language models, batch compilers, and automated pipelines exemplify this regime. Note that the internalisation of the repair loop does not increase the expressive power of the projection—it relocates where constraint satisfaction occurs.

17.3 Control Resolution

The distinction between these regimes is not categorical but continuous. Let Δt_{ctrl} denote the minimum interval at which gesture can modify the evolving trajectory $x(t)$. High-intervention systems satisfy

$$\Delta t_{\text{ctrl}} \to 0,$$

while low-intervention systems exhibit large Δt_{ctrl}, approaching a single-step mapping from initial condition to final output. The critical parameter is not the complexity of the internal model but the temporal resolution at which the user's gesture remains coupled to system dynamics.

17.4 Coupling Strength and Expressive Power

Expressive power is not solely a property of Φ_i but of the *coupling* between $\hat{G}$ and system dynamics. When Δt_{ctrl} is small, the relational structure of $\hat{G}$ is continuously reflected in $x(t)$, and admissibility is achieved through direct manipulation. When Δt_{ctrl} is large, the mapping $\hat{G} \mapsto x(t_1)$ is mediated by internal dynamics, and the original gesture structure is partially obscured.

This explains an otherwise puzzling phenomenon: systems with minimal internal modelling can feel more powerful than systems with substantial internal modelling. The difference is not capability but coupling. A tightly coupled simple system preserves the gesture structure all the way to the output; a loosely coupled complex system substitutes its own internal dynamics for the user's intent.

17.5 Interpretation

Tools and agents are not fundamentally distinct classes. They are points along a continuum defined by Δt_{ctrl} and the location of admissibility repair. The gesture-first framework suggests that increasing capability does not require increasing autonomy. It requires preserving the user's ability to intervene in the evolving trajectory and to participate in the repair process that enforces admissibility.

The design question shifts from "how autonomous should the system be?" to "at what temporal and structural resolution should gesture remain coupled to system dynamics?"

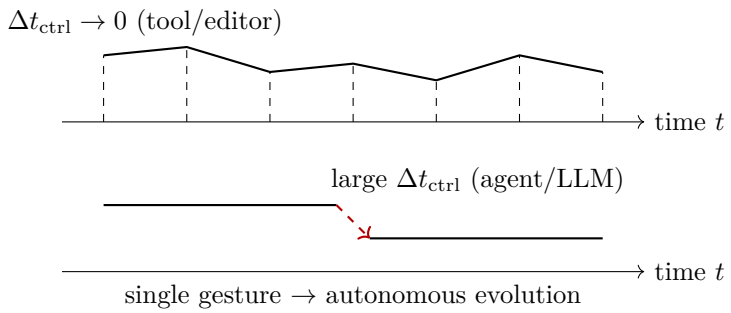


Figure 11: Continuous coupling (tool regime, top) versus discrete jump (agent regime, bottom). In the tool regime, gesture can modify $x(t)$ at fine resolution; in the agent regime, $\hat{G}$ becomes an initial condition for an autonomous process the user cannot steer mid-flight.

18. Future Directions

18.1 Optimal Control Resolution

The introduction of Δt_{ctrl} as a continuous parameter suggests a research programme for optimising control resolution in gesture-coupled systems. Fully autonomous systems and fully manual tools may both be suboptimal extremes: the former sacrifice expressive coupling for convenience; the latter place the full repair burden on the user. Hybrid systems in which admissibility repair is partially external and partially internal may offer a more favourable balance. Characterising the Pareto frontier between responsiveness and autonomy—and identifying which gesture structures are most sensitive to control resolution—is an open problem.

18.2 Admissibility Log as a Design Primitive

The admissibility log $\mathcal{L}$ introduced in Section 8 records the temporal evolution of ϵ_t across gesture–projection pairs. A natural extension is to expose this log to the user as an interactive surface: a mechanism by which the history of constraint satisfaction and failure becomes visible and manipulable. In such a system, the user would intervene not only in the gesture itself but in the dynamics of admissibility evaluation, steering the trajectory toward coherence at the meta-level.

This extends the gesture-first framework from representation and projection to a third layer: the user's relationship to the constraint structure governing valid output. Whether this layer can be made navigable through gesture—treating admissibility itself as a space to explore—is an open question with implications for programming environments, musical improvisation systems, and structured writing tools.

18.3 Esoteric Input Languages

The gesture grammar introduced in Section 6 is defined over the biomechanical constraints of standard keyboards. A natural extension is to explore alternative constraint

structures: non-standard keyboard layouts such as Dvorak or Colemak alter the finger-domain partition without changing the relational manifold; chorded keyboards such as stenographic systems alter the simultaneity structure; and isomorphic instruments such as the Lumatone alter the pitch-mapping geometry.

More speculatively, the framework opens the possibility of intentionally designed "esoteric" input languages—gesture grammars where the canonical equivalence classes are defined not by phonological or musical convention but by arbitrary structural criteria. Just as esoteric programming languages explore the boundary between computation and symbol manipulation, gesture languages could explore the boundary between motor control and semantic production. The canonicalisation operator $\mathcal{C}$ would remain invariant; only the projection Φ_i would change, mapping the same gesture vocabulary into a novel output domain.

18.4 Cryptographic Applications

The stability and individuality of gesture patterns under canonicalisation has a direct application in authentication and cryptographic key derivation. Keystroke dynamics as a biometric have been studied primarily at the level of timing sequences [3, 4]; the present framework suggests that the richer relational structure of canonical gestures—overlap patterns, pivot density, anchor-hand asymmetry, and inter-hand synchronisation—could provide substantially higher entropy for biometric identification or continuous authentication.

Because $\hat{G}$ is defined over topological features rather than raw timestamps, canonicalisation introduces a structural inversion with cryptographic implications. On the eavesdropper's side, timing-based side-channel attack yields only an equivalence class under $\mathcal{C}$ rather than a unique signature: the jitter tolerance δ acts as a privacy parameter, collapsing nearby timing signatures into a single canonical form. On the authenticator's side, the same equivalence structure increases internal distinctiveness, because two users with similar timing profiles may nonetheless produce distinct canonical gestures if their overlap patterns, pivot structure, or anchor-hand asymmetry differ. The system trades raw timing precision for higher-level structural entropy. Developing a cryptographically grounded treatment of gesture-based authentication—including formal bounds on the entropy of $\hat{G}$ and the privacy loss introduced by $\mathcal{C}$—is a natural direction for future work.

18.5 Gesture Space as a Dynamical System

The present framework treats G as a trajectory, but has not fully analysed the space of gestures as a dynamical system in its own right. A natural extension is to define a flow on gesture space:

$$\frac{dG}{dt} = \mathcal{F}(G,\ u(t)),$$

where $u(t)$ is the motor input signal and $\mathcal{F}$ encodes biomechanical and coupling constraints. Under this view, the gesture space acquires attractors (commonly repeated words,

phrases, or musical motifs), repellers (biomechanically difficult configurations), and stable limit cycles (rhythmic patterns in music or typing). Learning then corresponds to the convergence of motor trajectories toward the basin of attraction of a canonical gesture class, and expertise is characterised by reduced variance around that attractor. The canonicalisation operator $\mathcal{C}$ can be interpreted as a projection onto the attractor manifold, and the learnability condition of Section 5 becomes a condition on the width of that manifold relative to the noise in $u(t)$.

This dynamical framing connects the gesture-first framework to the broader programme in which trajectories—not states—are the primary objects of analysis. The RSVP amplitwist cascade formalises this connection precisely. Define the base amplitwist operator on the canonical gesture manifold:

$$\mathcal{A}(\hat{G}) = \|\vec{v}(\hat{G})\| \cdot \exp\left(i \, \arccos \frac{\vec{v}(\hat{G}) \cdot \nabla\Phi(\hat{G})}{\|\vec{v}\| \, \|\nabla\Phi\| + \varepsilon} \right),$$

where $\vec{v}$ is the RSVP velocity field, Φ is the scalar constraint-density field, and $\varepsilon > 0$ ensures stability. The phase angle $\theta = \arccos(\cdots)$ encodes the misalignment between conceptual velocity and the semantic salience gradient; the amplitude $\|\vec{v}\|$ encodes the magnitude of the gesture's kinetic energy.

Recursive semantic layers $\mathfrak{R}_k$ deform the base manifold via accumulated Lie-algebraic rotations:

$$\mathfrak{R}_k(\hat{G}) = \hat{G} + \sum_{j=1}^{k} \epsilon_j T_j(\hat{G}), \qquad T_j \in \mathfrak{so}(n),$$

and the layer-k amplitwist is evaluated at the deformed point with an entropy suppression weight:

$$\mathcal{A}^{(k)}(\hat{G}) = w_k(\hat{G}) \cdot \mathcal{A}(\mathfrak{R}_k(\hat{G})), \qquad w_k = e^{-\lambda S(\hat{G})}.$$

As k increases, the manifold warps under the accumulated Lie action, and emergent vorticity $\xi^{(N)} = |\nabla \times \hat{v}|$ identifies stable attractor regions—the gesture-space analogue of common words, phrases, or musical motifs. Theorem 3.1 of the RSVP programme establishes that $\xi^{(N)}$ converges when $\epsilon_j < \epsilon_{\text{crit}}$, guaranteeing that the cascade reaches a stable attractor rather than diverging [11].

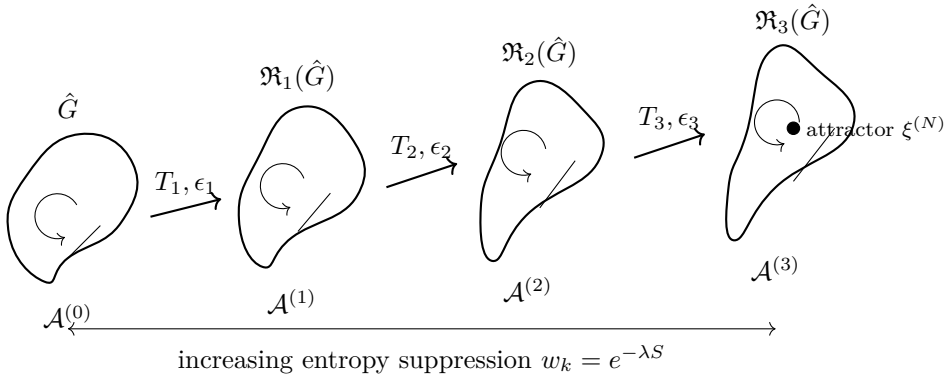


Figure 12: Amplitwist cascade: recursive deformation of canonical gesture manifold $\hat{G}$ under Lie-algebraic rotations $T_j \in \mathfrak{so}(n)$. At each layer k, entropy weight $w_k = e^{-\lambda S}$ suppresses high-uncertainty regions. Arcs encode phase angle θ; radial lines encode amplitude $\|\vec{v}\|$. The filled circle marks the attractor identified by vorticity $\xi^{(N)}$—the fixed point toward which gesture learning converges [11, 14].

18.6 Information-Theoretic Loss

The claim that current systems are "lossy" is made throughout this paper but not yet quantified. A rigorous extension would measure the mutual information lost at each stage of the serialisation pipeline.

Define the compression operator $\Pi : G \to K$ mapping the full gesture to the standard keydown stream, and the information loss as:

$$\mathcal{L}_{\text{info}} = I(G) - I(\Pi(G)).$$

Because Π discards simultaneity, duration, and inter-hand relations, $I(\Pi(G)) \ll I(G)$ for typical typing gestures. Estimating this gap empirically—using entropy measures over recorded keystroke streams compared against richer gesture logs—would transform the structural argument of Section 3 into a quantitative one. It would also provide a principled basis for comparing input systems: stenography recovers more of $I(G)$ than standard typing; a full gesture-capture system recovers more still. The information-theoretic loss $\mathcal{L}_{\text{info}}$ thus serves as a universal metric for comparing the expressive efficiency of any input interface.

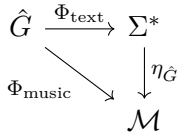
18.7 Category-Theoretic Extension: Natural Transformations

The projection framework currently defines functors $\Phi_i : \hat{\mathcal{G}} \to \mathcal{O}_i$ independently for each output domain. A stronger categorical treatment would define morphisms *between* projections, capturing structure-preserving relationships across modalities.

Specifically, define a natural transformation:

$$\eta : \Phi_{\text{text}} \Rightarrow \Phi_{\text{music}}$$

such that for every canonical gesture $\hat{G}$, the diagram



commutes. Such a natural transformation would formalise cross-modal translation as a structure-preserving map rather than a reinterpretation: the musical rendering of a gesture is related to its textual rendering by a systematic, gesture-independent transformation. This framework would clarify when two modalities are genuinely isomorphic projections of the same gesture (the transition feels natural) versus merely coincidentally related (the transition requires new motor learning).

18.8 Admissibility as Obstruction Theory

The admissibility log $\mathcal{L}$ is already structurally close to sheaf cohomology. The connection can be made precise. Let the canonical gesture space $\hat{\mathcal{G}}$ be covered by local patches corresponding to individual gesture arcs γ_k. A projection Φ_i defines local sections over each patch. Global admissibility requires that these local sections agree on overlaps—the standard sheaf gluing condition. When they fail to agree, the obstruction is not merely a practical error but a cohomological invariant:

$$\epsilon \in H^1(\hat{\mathcal{G}},\ \mathcal{F}),$$

where $\mathcal{F}$ is the sheaf of locally admissible projections. In this language, "failure to produce a valid word" is not a mistake but a non-trivial cohomology class: the local arcs are individually coherent, but they cannot be glued into a global section [12]. This reframing has practical implications: the obstruction class ϵ can in principle be computed locally and used to guide the repair process before the full projection is attempted. It also connects the gesture-first framework to a body of mathematical tools—Čech cohomology, derived functors, and spectral sequences—that have not previously been applied to input system design.

19. Limitations and Open Questions

Several questions remain open. The canonicalisation operator $\mathcal{C}$ is defined structurally but not yet implemented; the robustness of pivot detection under real typing noise is an empirical question that keystroke dynamics research has only partially addressed [3, 5]. The gesture grammar sketched here is descriptive rather than generative; a full formal grammar with a well-defined derivation relation remains to be developed. Inter-user variability in gesture patterns raises questions about whether canonical equivalence classes are stable across individuals or must be personalised [4]. The admissibility functional $\mathcal{A}$ is defined abstractly; specifying concrete admissibility conditions for natural language and musical projection will require empirical grounding. Finally, the relationship between

the present framework and existing work in motor neuroscience [8], continuous gesture recognition [1], and isomorphic keyboard design [10] deserves systematic treatment.

A structural gap that deserves explicit acknowledgement is the *injectivity problem* of the admissibility functional. The framework as stated assumes that $\mathcal{A}$ selects a coherent projection, but does not specify what happens when multiple projections are simultaneously admissible at comparable levels:

$$\exists\ \Phi_i,\ \Phi_j \ \text{ s.t. } \ \mathcal{A}(\hat{G}, \Phi_i) \approx \mathcal{A}(\hat{G}, \Phi_j).$$

This is not an edge case. For sufficiently expressive gesture vocabularies, many gestures will be simultaneously admissible under text and music projections, or will map to multiple equally plausible words within a single projection. Three resolution strategies are available. First, admissibility can be treated as inducing a distribution rather than a predicate, with output being a weighted family $\mathcal{P}(\mathcal{O}_i \mid \hat{G}) \propto \mathcal{A}(\hat{G}, \Phi_i)$; ambiguity is then a first-class state rather than a failure mode. Second, admissibility can be made path-dependent, with the user resolving ambiguity by continuing the gesture or modifying the control loop—consistent with the high-intervention regime of Section 13. Third, each projection can carry its own admissibility bias, effectively embedding a type system into the functional. The paper does not commit to any of these; establishing which resolution strategy is most consistent with the gesture-first framework is an open theoretical problem.

Skill acquisition also receives an incomplete treatment. The learnability condition of Section 5 states that $\mathcal{C}(G_{\text{motor}}) = \hat{G}$ must be stable under perturbation, but does not define a metric on the space of motor gestures that would make this precise. A natural proposal is to define skill as the reduction of variance in canonical equivalence classes:

$$\text{skill} \ \propto \ \frac{1}{\text{Var}\big(\mathcal{C}(G_{\text{motor}})\big)}.$$

This reframes expertise not as speed or accuracy but as *topological consistency*: the capacity to produce the same canonical gesture across varied conditions and with minimal jitter. Whether this definition aligns with empirical measures of typing proficiency is an empirical question, but it suggests a testable prediction: expert typists should exhibit lower variance in the relational features (overlap patterns, pivot timing) of their canonical gestures than novices, independent of overall keystroke rate.

20. Conclusion

Standard keyboard interfaces discard most of the signal the hands produce. The gesture is present in every act of skilled typing: in the overlapping contacts, the durations and releases, the convergence and divergence of hands, and the pivots that structure movement into phrases. Current systems reduce this to an ordered list of keydown events and treat everything else as noise.

This paper has argued that the discarded structure is the primary object, and that text and music are projections of it. The formalisation introduced here—hierarchical gesture space, canonicalisation by topological equivalence, projection functors, and the admissibility layer that constrains which projections are coherent—provides a framework for recovering that structure without requiring new hardware or new motor learning. The problem is not how to type faster. It is how to stop ignoring what the hands are already saying.

A. Example Gesture Decomposition

The word *machine* illustrates the slingshot arc structure. The gesture decomposes into three arcs:

1. **Arc 1** (`m` ↔ `a`): left index and left middle approach a shared region; `m` is the onset, `a` the pivot.
2. **Arc 2** (`a` ↔ `ch`): pivot on `a`, hands spread to the `ch` cluster; the `h` serves as the secondary pivot.
3. **Arc 3** (`i` ↔ `n` ↔ `e`): `n` is the turnaround pivot between `i` and `e`; full release terminates the gesture and signals the word boundary.

The pivot letters are each simultaneously the endpoint of one arc and the origin of the next. No spacebar is required; the word boundary emerges from the release structure of the gesture itself.

B. Minimal Implementation Sketch

A prototype gesture capture layer requires the following pipeline:

1. **Capture.** Intercept raw keydown/keyup events with sub-millisecond timestamps using a low-level keyboard hook (`pynput`, `evdev`, or equivalent).
2. **Channel mapping.** Assign each physical key to a finger channel f_i based on standard touch-typing finger ownership.
3. **Feature extraction.** Compute overlap groups, per-finger durations, migration directions, and pivot events from the timestamped stream.
4. **Canonicalisation.** Apply $\mathcal{C}$ to suppress jitter and extract topological features.
5. **Projection.** Apply the appropriate Φ_i to produce text, MIDI, or other output.
6. **Admissibility evaluation.** Score the projected output under $\mathcal{A}$ and log the result to $\mathcal{L}$.

The bottleneck is step 4. Steps 1–3 and 5–6 are straightforward with existing libraries.

C. Proof-of-Concept Event Capture

The following Python sketch demonstrates the capture and channel-mapping layers using `pynput`. It records timestamped keydown and keyup events and assigns each key to a finger channel based on standard QWERTY touch-typing zones.

Listing 1: Gesture capture skeleton (`pynput`)

```
from pynput import keyboard
import time

# QWERTY finger-channel assignment (0=left-pinky, 7=right-
    pinky)
FINGER_MAP = {
    'q':'LP','a':'LP','z':'LP',
    'w':'LR','s':'LR','x':'LR',
    'e':'LM','d':'LM','c':'LM',
    'r':'LI','f':'LI','v':'LI','t':'LI','g':'LI','b':'LI',
    'y':'RI','h':'RI','n':'RI','u':'RI','j':'RI','m':'RI',
    'i':'RM','k':'RM',',':'RM',
    'o':'RR','l':'RR','.':'RR',
    'p':'RP',';':'RP','/':'RP',
}

events = []  # raw event log: (time, key, direction, channel)

def on_press(key):
    t = time.perf_counter()
    k = getattr(key, 'char', str(key))
    ch = FINGER_MAP.get(k, 'UNKNOWN')
    events.append((t, k, 'DOWN', ch))

def on_release(key):
    t = time.perf_counter()
    k = getattr(key, 'char', str(key))
    ch = FINGER_MAP.get(k, 'UNKNOWN')
    events.append((t, k, 'UP', ch))
    # Detect full release (all channels UP) as word boundary
    active = {e[3] for e in events if e[2]=='DOWN'} - \
             {e[3] for e in events if e[2]=='UP'}
    if not active:
        print(f"[BOUNDARY] word boundary at t={t:.4f}")

with keyboard.Listener(on_press=on_press,
                       on_release=on_release) as listener:
    listener.join()
```

This is the minimal capture layer. The canonicalisation step ($\mathcal{C}$) would consume the `events` list, group entries within the simultaneity window ε, detect pivots from direction

reversals in each channel, and produce a canonical gesture object $\hat{G}$ ready for projection.

D. MIDI Projection Sketch

The music projection $\Phi^{\theta}_{\text{music}}$ maps canonical gesture to a pitch-time structure parameterised by θ (pitch range, tuning, interval mapping). The following sketch demonstrates a simple interval-stretching mapping where the left-hand channels control vertical harmonic position and the right-hand channels control horizontal melodic movement, with the full keyboard spanned across a configurable interval range.

Listing 2: MIDI projection with interval mapping (`mido`)

```
import mido

def build_pitch_map(root_midi=60, interval_semitones=7,
                    n_keys=88):
    """Map key index linearly across [root, root+interval]."""
    return [
        int(root_midi + (interval_semitones * i / (n_keys - 1)
            ))
        for i in range(n_keys)
    ]

def project_to_midi(canonical_gesture, pitch_map,
                    port_name='gesture_out'):
    """Send canonical gesture as MIDI note events."""
    with mido.open_output(port_name, virtual=True) as port:
        for arc in canonical_gesture.arcs:
            for event in arc.events:
                pitch = pitch_map[event.key_index]
                velocity = int(event.duration_ms * 2)  # map
                    hold -> velocity
                msg_on  = mido.Message('note_on',
                                       note=pitch,
                                       velocity=min(velocity,
                                           127))
                msg_off = mido.Message('note_off',
                                       note=pitch,
                                       velocity=0)
                port.send(msg_on)
                # sustain for arc duration
                time.sleep(event.duration_ms / 1000)
                port.send(msg_off)
```

Changing θ (the `interval_semitones` parameter) stretches or compresses the pitch space without altering the canonical gesture $\hat{G}$, illustrating metric independence. Spanning the full keyboard across a fifth ($\theta = 7$) makes harmonic motion local; spanning it across a semitone ($\theta = 1$) produces a microtonal surface with very fine pitch resolution.

E. Master Diagram: The Unified Pipeline

The following diagram shows the entire gesture-first framework as a single geometric structure. Raw gesture trajectories are mapped to canonical equivalence classes by $\mathcal{C}$, lifted into output domains by projection functors Φ_i, and evaluated in a constraint space by the admissibility functional $\mathcal{A}$. The control loop Δt_{ctrl} governs temporal coupling between gesture and projection.

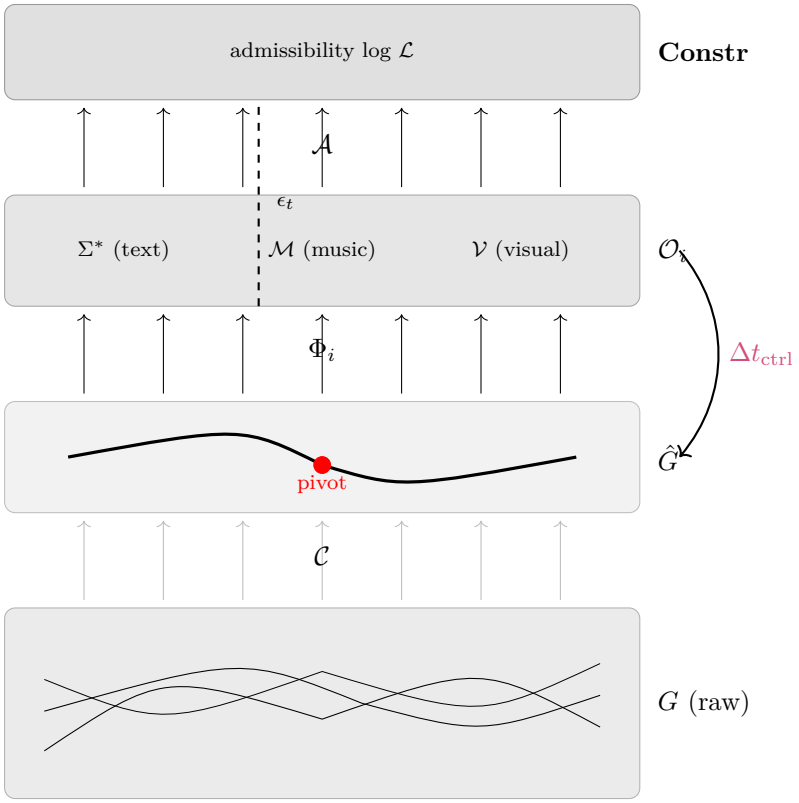


Figure 13: Unified geometric structure of the gesture-first framework. Bottom to top: raw gesture space G; canonical space $\hat{G}$ (fibers collapse under $\mathcal{C}$); output space $\mathcal{O}_i$ (lifted by Φ_i); constraint space **Constr** (evaluated by $\mathcal{A}$, logged in $\mathcal{L}$). The dashed line shows an obstruction ϵ_t propagating through layers. The purple feedback arc represents the control loop Δt_{ctrl} by which gesture can modify the trajectory during projection.

References

[1] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. In *Proceedings of UIST*, 2007.

[2] Shumin Zhai and Per Ola Kristensson. The word-gesture keyboard: Reimagining keyboard interaction. *Communications of the ACM*, 55(9), 2012.

[3] Kevin S. Killourhy and Roy A. Maxion. Comparing anomaly-detection algorithms for keystroke dynamics. In *Proceedings of DSN*, 2009.

[4] Fabian Monrose and Aviel D. Rubin. Keystroke dynamics as a biometric for authentication. *Future Generation Computer Systems*, 16(4), 2000.

[5] Daniele Gunetti and Claudia Picardi. Keystroke analysis of free text. *ACM Transactions on Information and System Security*, 8(3), 2005.

[6] The Plover Project. Open-source stenography software, 2010–present. `https://www.openstenoproject.org/plover/`

[7] I. Scott MacKenzie and R. William Soukoreff. Text entry for mobile computing: Models and methods, theory and practice. *Human-Computer Interaction*, 17(2–3), 2002.

[8] Marc H. Schieber. Individuated finger movements: Limb kinematics, physiology, and motor control. *Journal of Neurophysiology*, 2001.

[9] Karl Lashley. The problem of serial order in behavior. In *Cerebral Mechanisms in Behavior*, 1951.

[10] Lumatone Music. Lumatone isomorphic keyboard documentation, 2021. `https://www.lumatone.io/`

[11] Alicia Juarrero. *Dynamics in Action: Intentional Behavior as a Complex System.* MIT Press, 1999.

[12] Alicia Juarrero. Context changes everything: How constraints create coherence. *Emergence*, 4(1–2):99–110, 2002.

[13] Arnie Cox. *Music and Embodied Cognition: Listening, Moving, Feeling, and Thinking.* Indiana University Press, 2016.

[14] Tristan Needham. *Visual Complex Analysis.* Oxford University Press, 1997.

[15] Newman Cheng, Gordon Broadbent, and William Chappell. Cognitive loop via in-situ optimization: Self-adaptive reasoning for science. arXiv:2508.02789, 2025. `https://arxiv.org/abs/2508.02789`

[16] Jana M. Iverson and Susan Goldin-Meadow. Gesture paves the way for language development. *Psychological Science*, 16(5):367–371, 2005.

[17] Susan Goldin-Meadow. *Hearing Gesture: How Our Hands Help Us Think*. Harvard University Press, 2003.

[18] E. Sue Savage-Rumbaugh, Stuart G. Shanker, and Talbot J. Taylor. *Apes, Language, and the Human Mind*. Oxford University Press, 2000.