Live Notation for Patterns of Movement

Alex McLean and Kate Sicchio

Notation in the Algorithmic Arts

Notations are by nature incomplete, often describing continuous expressions in terms of discrete, symbolic operations. How can abstract symbols generalize gestures, with all of their nuanced timing and qualities? This question gets to the heart of how we can notate movement. Choreography can be seen as the organization of gestures in space and time, and yet despite well-known attempts, there is no widely adopted choreographic notation. Reflecting on this, we look to develop notation for algorithmic choreography that is specific to a particular dance. Bringing computer programming closer to dance can connect coding and moving in a creative feedback loop. However, we first need to lay conceptual ground for our approach to this work by challenging some pervasive assumptions that we believe hold the algorithmic arts back.

Establishing Dichotomies

There are three pairs of concepts that have become entangled and conflated when discussing the algorithmic arts: computation vs automation, determinism vs predictability, and sequencing vs patterning. Teasing apart these terms reveals notation as a live choreographic interface for working with patterns through *nonautomated computation* and *unpredictable determinism*.

Computation vs Automation

These two terms are too often needlessly conflated, but here we are mainly interested in considering computation *without* automation. Once we separate these two terms, we open up all the possibilities of human-action-as-computation, including its long, partially hidden history in traditional

Alex McLean (Then Try This) is based in Sheffield, UK, working on algorithmic patterns including in music, textiles, and dance. He has been working professionally in creative technology since 2000, including as live coding musician and software artist, festival curator/producer, and research fellow. He created the popular free/open source live coding environment TidalCycles, and cofounded the TOPLAP live coding and Algorave movements, and the AlgoMech festival. He coedited the Oxford Handbook on Algorithmic Music *(2018), and coauthored* Live Coding: A User's Manual *(2022). He now holds a UKRI Future Leaders Fellowship. alex@thentrythis.org*

Kate Sicchio (Virginia Commonwealth University) is a choreographer, media artist, and performer whose work explores the interface between choreography and technology with wearable technology, live coding, and real time systems. Her work has been shown in the US, Germany, Australia, Belgium, Sweden, and the UK at venues such as PS122 (NYC), Banff New Media Institute (CAN), and Arnolfini Arts (UK). She coedited Intersecting Art and Technology in Practice: Techne/Technique/Technology *with Camille Baker (2017). She has given invited talks at countless universities and events, including: EU Parliament, Eyeo, Resonate, Node Code, and Expo 74. She is currently Associate Professor of Dance and Media Technology. ksicchio@vcu.edu*

TDR 68:1 (T261) 2024 https://doi.org/10.1017/S1054204323000576

craft culture. After all, humans have always computed things, as is clearly evident in how handcrafts such as weaving and braiding are full of algorithms for generating patterns. Craftspeople have explored computation well before automation, and while the Jacquard device[1] is often cited in connecting weaving with computing, handweaving has always been computational. Jacquard's device only stands for automating the computation, thereby taking it out of human hands (Harlizius-Klück 2017).

Human action-as-computation is also seen in dance and choreography, wherever rule-based systems are used to create and perform movements of the body. A mark of this is where simple instructions are used to generate complex choreographic movements and relationships. One such computational approach was used by Trisha Brown for *Locus* (1975), which does not involve electronic computers, but does involve a distinct system for performing movement through a diagrammatic cube that assigns movement in space around a dancer (Graham 2016). In *Locus*, the focus is on spatial placement of gestures and not the movement itself, resulting in different sections of the piece having different movements (Sulzman 1978).

Algorithmic systems for organizing movement are common in social dance and folk traditions too, such as European maypole dances and the comparable but more complex Tamil dance of Pinnal Kollattam. In both cases each dancer is given instructions to follow, dancing in and out in opposing directions to create a patterned braid around the maypole, or in the case of Pinnal Kollattam, in three-dimensional space. The more involved the dance, the more complex the patterns of the resulting braid become, potentially creating surprising interference patterns where opposing ribbon colors interlace according to the shifting clockwise/counter-clockwise movements of a dance, creating a particular braided structure.

Over the past few decades, such systems-based approaches have developed in stage dance too, where they have become collectively known as *algorithmic choreography*. This term is often associated with Merce Cunningham and his use of the *Dance Forms* software in the 1990s (Schiphorst 1993), but this is a later case that focuses on automation of the choreographic process through the use of computers, rather than working with computation as a material in its own right. Choreographer Jeanne Beaman's earlier work from the 1960s also aimed to automate the process of the choreographer, creating dances from chance procedures by using a computer to select movement, duration, and space from a list. The computer created "70 dances in 4 minutes" (Eacho 2021). However, again, these examples tend to view the computer as a sequencer and randomizer, automatically selecting movements, and putting them together in new combinations, but not really *computing* anything. By contrast, a maypole or Pinnal Kollattam dance does compute something tangible—the interference patterns emerging from human movements being captured as potentially complex braids.

Computation in dance can be human or technology-based, bringing together processes that follow rule-based systems to create choreography. These algorithms may be simple or complex, designing movement through gestures, spatial patterns, or timing structures. Automation, as in the case of machine weaving, may use computation to create choreography that is then executed by movers, but we believe that automation needs computation, more than computation needs automation.

Determinism vs Predictability

A system can be both deterministic and unpredictable. The results of running such a process is practically impossible to predict in advance, yet when run again, always produces the exact same results. Theoretically, we know this from the halting problem in computer science; that is, there is no general method for telling whether a given deterministic procedure will complete, so we can't always predict whether such a procedure will return an answer at all. We also know deterministic,

[1] A Jacquard device automates the control of an industrial loom for weaving using punched cards to select threads.

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

unpredictable systems from chaos theory: if we make a very slight change to a deterministic, chaotic system, it will give completely different results.[2]

This difference between determinism and predictability can be seen in the way that random number generators are often used in media arts to create variable results. These are known as *pseudo*-random number generators, because they are deterministic, only giving variation by including time and state in their calculation. They are designed to be unpredictable through the use of chaotic interference patterns, creating deterministic sequences that exhibit the properties of noise. So, they are a good example of deterministic unpredictability. However, pseudo-random number generators are not what we are interested in here. Rather we are addressing the creative possibilities of composing unpredictable, deterministic patterns by hand, through nonautomated computation. This comes not from work with "black box" random number generators, which make arbitrary choices. Instead, we see creative possibilities in hands-on experimentation with algorithmic pattern operations—for example repetitions, symmetries, and interference patterns at multiple scales, to create unpredictable, complex results from simple parts.

Sequencing vs Patterning

The third and final dichotomy is between linear sequences and patterns. The word "pattern" is applied across many fields including mathematics, arts, crafts, and design, but carries a different weight and meaning in each of them; in some cases it is even pejorative. For instance, in music, *pattern* tends to stand for any repeating, discrete sequence, and therefore patterns tend to be loved by electronic dance musicians, but derided by "serious" classical composers (Hugill 2020). On the other hand, in textiles, the word *pattern* may refer to a technical procedure for a traditional craftsperson to follow, such as numerical braiding patterns indicating the relational exchange of threads, or procedural knitting patterns where the craftsperson follows instructions to loop, jump, and switch between parts of the pattern in order to produce a garment of the correct design and size. Here then, we refer to *sequences* as linear or step-by-step, and *patterns* as more nuanced, branching, and recursive procedures with symbolic elements that may be repeated, but are also transformed at different scales, and aligned and composed together to create complex results.

This dichotomy of sequencing vs patterning follows from the previous two dichotomies, computation vs automation and determinism vs predictability. Automatism tends towards simple, predictable step-by-step sequences, not only to make the automation easy to accomplish, but also to make it possible for a human operator to step in when there is a procedural breakdown (Bainbridge 1983). Computation on the other hand embraces unpredictability in combining potentially simple elements in order to produce surprising, complex forms.

Complexity from Simplicity

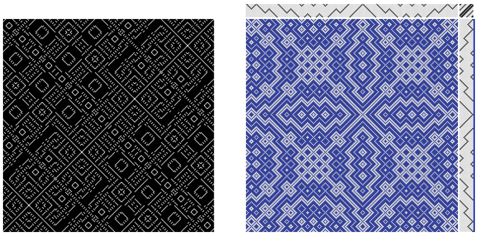
If we embrace computation and determinism, and reject automation and predictability, what is left to work with in terms of creative material? Our answer is an approach that focuses on perceiving the complex and unpredictable results of simple pattern-making processes. We use the word *complex* with care, however. Where artists engage with computation, there may be an assumption that they are doing complex, unfathomable things. We argue that this is a failure to grasp the affordance of computation as a means to generate and experience complex forms from simple structures. Two examples support this: a contemporary bitfield pattern and a heritage weaving pattern, shown side-by-side in figures 1 and 2. Figure 1 shows a "bitfield" rendering of the function `(x^y)%9==0` for each (x, y) point in two-dimensional space.[3] The ^ operator stands for a bitwise exclusive-or,[4]

[2] A classic example of a chaotic system would be a game of pinball, where no two games are alike.

[3] For community discussion of bitfields and related practices of bytebeat and sizecoding, see https://forum.algorithmicpattern.org/t/bytebeat-tixyland-and-other-functions-of-time/396.

[4] For details on bitwise operations see https://en.wikipedia.org/wiki/Bitwise_operation.

and % for modulo or remainder of a division. By aligning the bits of two “x” and “y” numbers, combining them with this Boolean operation, and painting the respective pixel white where the result divides equally by 9, an interference pattern is revealed. Figure 2 shows a similar interference pattern, but from handweaving.[5] Interference is again set between x and y directions (which weavers refer to as weft and warp, respectively). This pattern arises from a matrix multiplication of the binary values from the shaft loom configuration shown at the top, the feet movements (treadling) shown on the right, and the connection between them (tie up) in the top right. The resulting weaving pattern (seen in figure 2 as the darker elements of the pattern) shares visual similarities with the bitfield pattern on the left, despite the manner of construction being very different. However, we should give proper respect to the far older and more developed craft of weaving. There is much more that we could vary in the weaving block design, for example by introducing a pattern to the selection of color across the warp and weft threads. Also bear in mind that this two-dimensional design represents an entanglement of threads into three dimensions, a process that creates visual and physical properties far more complex than what is represented in two dimensions on the screen. But still, the family resemblance between these computational crafts, one contemporary and one ancient, is striking.



Figures 1 & 2. Bitfield pattern (image by Martin Kleppe) and a traditional “Earl’s Canvas” weaving pattern (image courtesy of Denise Kovnat), both two-dimensional algorithmic patterns.

Although perhaps simple, these coded patterns provide very rich ground to explore. This is the affordance of language—instructions as elements that can be mixed together and rearranged in a multitude of ways, as interoperable fragments that may be abstracted or recursively embedded, creating a generative, combinatorial explosion of possibility. Its unpredictability means that every change might feel like a step into the dark; but where the results are immediately generated, as in the practice of live coding,[6] each change can instead feel like switching on a light.

Focusing on computation but *not* automation, and deterministic procedures but *not* predictability, clears the way to seek notations for algorithms that represent rules for humans to follow through tacit, embodied engagement with movement.

Histories of Notating Movement

Reviewing related historical and contemporary approaches to movement notation allowed us to reflect upon and bring together the different histories of code and choreography. A key reference point in the representation and notation of human exertion as *effort* can be found in *Effort: Economy in Body Movement* written by choreographer Rudolf Laban in collaboration with management consultant F.C. Lawrence in 1947. The book proposes a systematic, seemingly elegant way to categorize human efforts in movement, through a combination of elements such as time, weight, and space. *Effort* is as applicable to industrial time-and-motion studies as it is to dance choreography. We should also note that Laban’s earlier work was as choreographer for Hitler and Goebbels, until his work on the 1936 Olympic games was rejected (the Nazis rejecting Laban, rather than vice versa). While this theory arose from white supremacy and Fordism, it continues to be influential in choreographic research and practice.

[5] For discussion of this pattern and its translation to echo threading see https://www.denisekovnat.com/2020/11/gebrochene-echo-and-jin-with-fiberworks.html.

[6] Live coding is a practice largely situated within the performing arts, where a performer creates a live work by writing code while it is being executed.

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

In South Indian Carnatic dance, spoken syllables are used to represent different movements, strung together to form movement phrases or *jatis*. These syllables may be written as notation to aid memory, but the emphasis is very much on orality, with different, overlapping sets of syllables associated with different dance styles (Seth 2017). This approach extends also to Carnatic music, for which syllables are central to the concept of rhythm, applied both to represent drumming articulations (e.g., on the mridangam) or directly in vocal performance, a practice known as *konnakol*. These ancient, yet still actively developing dance and music traditions pose a challenge for our otherwise Western-centric view of notation; a set of nonlexical syllables are used as symbols that represent gestures, but in spoken rather than written form. Because our interest is in *live* notation, where a written notation is transient, changing along with the piece it represents, this challenge is central to our ongoing work.

One example of a live notation system in stage choreography is Michael Klien and Nick Rothwell's late 1990s choreo/graph system. Choreo/graph provides a live score that is visible to dancers on screens at the sides of the stage. The system allows the dance to be changed in real time while being performed, most notably in Michael Klien's 2002 duet, *Duplex* (deLahunta 2002) at Theater am Turm (TAT) in Frankfurt with dancers from William Forsythe's company. Here a cueing system gave dancers either sequences of choreographic material or tasks to manipulate the movement. The liveness of this notational score demonstrates a digital system for dance that changes during the performance rather than a fully set piece created via algorithmic processes.

Live notation was also the name of a research network bringing together live coders and performers of live art.[7] McLean convened the live notation network in collaboration with live artist Hester Reeve (McLean and Reeve 2012), with Sicchio as a core member able to bridge coding and dancing through her performance practice. A far-reaching outcome of this exchange came from the recognition of a shared approach that project member Emma Cocker identified as *kairotic coding* (2014). She provided a philosophical reflection on the role of notation in live improvisation, using the mythological figure of Penelope's *un*weaving as the metaphor for a notation that can be unraveled and rewritten. This prefigured collaborative work with weaver and mathematician Ellen Harlizius-Klück, work that took the connection between live coding and weaving further, beyond metaphor into direct correspondence (Cocker 2017).

With these examples in mind, drawing on concepts from dance notation around documenting gesture, representing the ephemeral as in konnakol, and live notation that changes during performance (as with choreo/graph), we began to explore how to create a programming language for live coding the movement of humans and robot performers. Through the approach of live coding, which allows for continuous decision-making in both generating and responding to algorithmic patterns, we can generate choreography, through computational but not automated processes. By bringing robots into the work as performers, we seek connections between humans and machines through language and movement, bound together in liveness.

Live Coding Robotic Performance

Our work in algorithmic choreography as collaborators working at a distance between Richmond, Virginia, USA (Sicchio) and Sheffield, UK (McLean), builds upon our in-person collaborations while previously living in the same city (Sicchio and McLean 2017). At such a distance, it was perhaps natural to explore choreographic notation of robotic movement, and we did so using off-the-shelf ROBOTIS components (AX=12A servo motors, controlled via an Arduino interface).[8] We worked with two identical robots, one in each of our studios, with each robot having three servo motors, thereby providing three "degrees of freedom" (see fig. 2). Despite Sicchio's prior and

[7] Live art is a fine art/performance art tradition, using live bodily action as medium.

[8] For technical information about working with ROBOTIS servo motors, see the following blog post: https://slab.org/2022/02/28/making-robots-with-ax-12a/.

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

ongoing work with choreographing robots (Sicchio et al. 2022) and our earlier collaborations, we decided to come to this work afresh, reapproaching algorithmic choreography by thinking through practice.

We began by triggering movements of our robots using the Strudel live coding environment for algorithmic pattern (Roos and McLean 2023),[9] which is a port of the TidalCycles open source live coding system.[10] Despite being on different continents, we were able to work in the same editor via the Flok live coding environment,[11] which is designed for network music. Flok allows multiple people to code simultaneously within the same editor, where each participant has their own "cursor." When someone triggers a live edit, the running code is automatically updated on all participating computers. Despite being designed for making music, the combination of the Strudel javascript environment and Flok editor works for patterning servo motor activity as it does for sound.

This initial configuration therefore created the following chain of action, each link having a different bearing on the result:

> Sicchio and McLean → Flok editor → Strudel language → Arduino microcontroller → Servo motors

Being a live coding editor, the Flok system allows the creation and manipulation of code while it runs (Blackwell et al. 2022). Because the development of that code forms the structure of a piece, live coding performances tend towards a progressive build up and reduction of structural complexity, without sudden changes or shifts. Although largely designed for making music, the Strudel language is essentially a system for combining sequences and transformations, at multiple scales, in order to generate algorithmic patterns. Its use therefore tends towards generating complex patterns from simple parts, with broken symmetries that at first may seem arbitrary but through repetition reveal an underlying structure. So already, the live coding editor and language environment provide affordances for us to explore.

Ceding Control

We could stop here in framing our piece at the notational level of the live coding environment. To do so would be to focus on the notation of movement, rather than the movement itself and its influence on the human dancers and choreographers. Indeed, this is how we initially approached control of our robots, looking for ways to pinpoint a position in space using the notation, instructing each robot to move to that position, through a technique called *inverse kinematics*. This extreme level of control is made possible with robots, in that what you write on the computer is what you get.

On closer examination though, such an approach to notation as a technology of control[12] is a losing battle. Firstly, in our case, it is not possible for the end point of a robot (i.e., the end of its "nose") to move to any given point in space; each robot only has three motors, configured as a mechanical waist (bottom motor), back (middle motor), and head (top motor), respectively. Because the base of each robot is fixed to a table, it can only contort to certain positions. Even if the robot was free to reach any point that we might want it to, we would still have to carefully consider its physical characteristics. It takes time to move; so to reach a particular point at a particular time, you have to start early. If you move too fast, the robot will overshoot, and resonate back and forward until it settles

[9] The Strudel live coding environment runs in a web browser, and is accessible at https://strudel.cc/.

[10] As described at tidalcycles.org, "Tidal Cycles (or 'Tidal' for short) is a free/open source live coding environment for algorithmic patterns, written in Haskell. Tidal is using SuperCollider, another open-source software, for synthesis and I/O."

[11] The free/open source Flok editor for networked multiuser live coding is available at https://flok.clic.cf/.

[12] For an exploration of technologies of control versus technologies of work, see Ursula Franklin's transcribed lectures, "Real World of Technology" (Franklin 1999).

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

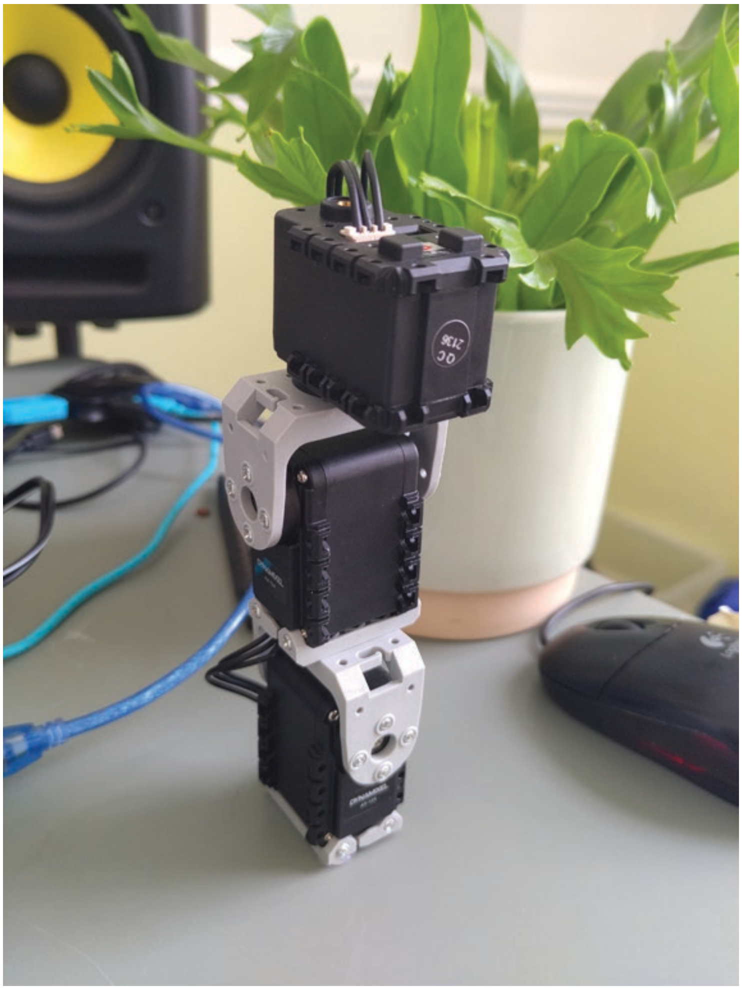


Figure 3. One of our robots, created from three ROBOTIS servo motors coupled with standard fixings, creating three degrees of freedom. (Photo by Alex McLean)

into the intended position. With servo motors, fast movements are also extremely noisy. We found the more we attempted to exert control over our robot through precise instructions, the more such uncontrollable, unwanted elements intruded on the work. So we instead worked with the robotic quirks as creative material, exploring issues of vibration and timing as resonances and rhythms that are intrinsic to the robot, offering up material affordances that we chose to embrace rather than work against.

Thus, in order to work with any robot, we first need to understand its physical constraints, using these as creative material to work with. Indeed, this is true also when working with humans. For example, percussionist Jaki Liebezeit, the founder of the experimental band Can, developed an approach to drumming with a notation based on the binary dot-dash familiar to morse code (Podmore 2020). The constraints of this system were based on the observation that in order to perform a louder "accented" strike of the drum, the hand needs to move the drumstick further, up and then down, which takes additional time. Liebezeit's system therefore worked on the interplay between left and right hands, where due to the laws of physics, an accent with one hand requires time provided by a preceding double movement with the other hand. From these simple rules, complex rhythms and paradiddles are generated. Accordingly, rather than seeing the physical constraints of a human or robotic agent as getting in the way of an ideal notated piece, we take physical constraints as a starting point, therefore providing a rich creative space in which to work. So what did our robots do? How did they dance?

Robots Dancing

As mentioned earlier, we each have one robot, sitting on our respective desks in Richmond and Sheffield, made from off-the-shelf servo motors. These motors not only allow external control over where they travel, how fast, and with how much power, but also allow the computer to sense its current position, and the pressure applied to it. These physical properties present a number of constraints. For example, it isn't possible to increase the speed of the robots without increasing the noise that it makes. The robot also isn't able to move around or across the surface to which it is mounted. More subtly, care should be taken to avoid overusing certain movements that wear out the components by overheating them or overstressing interconnecting wires.

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

Such physical properties offer us a number of choices in deciding how we interact with the robot, for example:

1. If the robot is already moving to a position when you tell it to move to a new one, should it complete its current move, or immediately adjust its course?
2. Should the robot simply move as quickly as possible to the new position, or should its speed follow a particular curve of acceleration and deceleration as it reaches its goal?
3. Should a movement be expressed in terms of the average speed of movement, or in terms of the time the movement should take?
4. Relatedly, should a movement be timed in terms of when the robot arrives at its target position, or in terms of when the movement begins?
5. How about relative movements, like "switch sides," or "move halfway to the ground"—should these movements be relative to the current target position of the moving robot, or to the current position?

To a large extent, all of these constraints and choices apply just as well to humans as to robots, with the exception that human muscles tend to move more quietly than servo motors—but human movements have their noises too, such as gasps, squeaks, and clicks. The primary difference then is that with robot performers, answers to such questions tend towards control, and with human ones, the answers tend towards respecting human agency. In considering them together, we are therefore able to make these choices explicit and open to interrogation.

As live coding language designers, we work by defining ways of making things, via the creation of a notation. This means that once we identify choices such as the five above, we are able to defer those choices by turning them into an option, codified as a parameter to a computer language function. This allows us to not only make these choices during a live performance, but change our minds, or even write code to vary the choice over time, following its own pattern.

In our case, we made the choices that most respected the characteristics of the robots we were working with. Our goal was to find movements that were interesting choreographic materials, rather than direct instructions. This relates to what Tim Ingold (2011) criticizes as *hylomorphism*, where a maker forces their ideas on material, rather than working *with* material through creative feedback. For example, considering choice 5 above, if we instruct a robot to "switch sides," it should move a servo until it mirrors its starting position; but what if we quickly tell it to switch sides again, while it is still moving? If we respect the imaginary world of the notation, the robot should go back to where it started, but if we respect the physicality of the robot, the robot should switch sides relative to its current position. In the latter case, if we keep interrupting its movements with the same "switch sides" instruction, each movement will be of a shorter distance until the robot settles at its center, orthogonal to the ground. By respecting and perhaps embodying the robot in this way, we have discovered a new gesture afforded by it, a movement that resonates and settles.

From Sequence to Pattern: Introducing the Robot to the Audience

The syntax of our language consisted of parts of the robot or body, predetermined gestures, and spatial-temporal relations. Our robots consisted of three motors, which we earlier loosely referred to as "head," "waist," and "back." Through initial explorations we created a small collection of simple gestures including "move," "lean," "switch," "toggle," and "reset." Other higher order movements such as "sway," "wiggle," and "diagonaltwist" were added later, described in the programming language in terms of the composition of basic electronic waveforms (e.g., sine, sawtooth, and square waves; white or perlin noise) directed to the different motors. The distance by which the motor would move and the duration of the movement were also programmable for each gesture. A simple

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

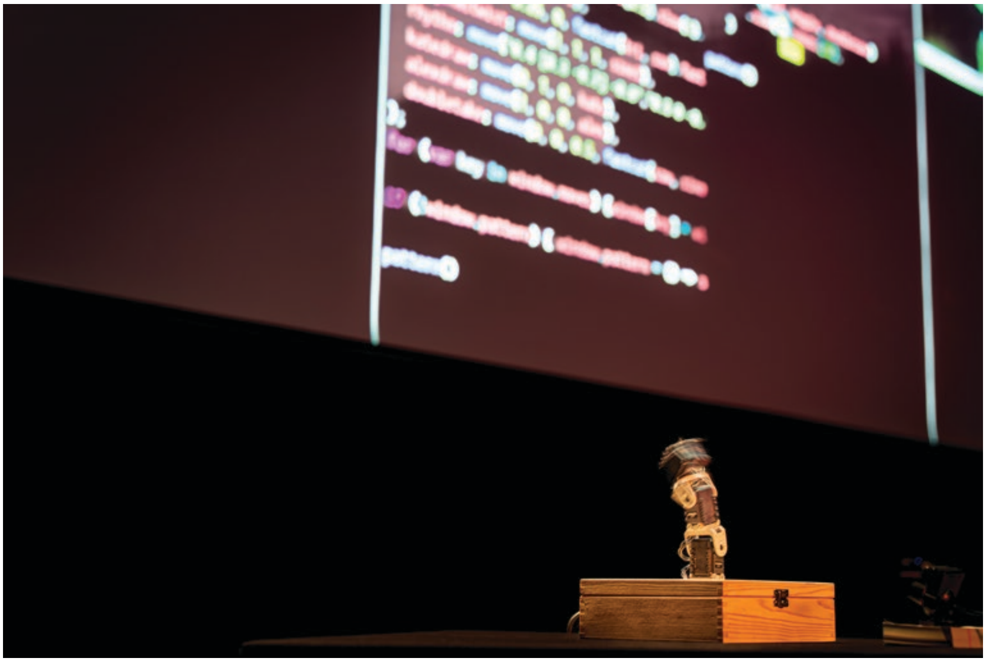


Figure 4. Robot shown performing in front of its code at the International Conference on Live Coding, Utrecht 2023. (Photo courtesy of Paulus van Dorsten)

program to move the top motor of the robot to 25% of its range over a quarter of a second would look something like this:

```
part(“head”)
.action(“move”)
.to(0.25)
.dur(250)
.robot()
```

This code could be looped, and new gestures, timings, and spatial commands could be added, creating compound movements with unique and often surprising characteristics. We could then quickly develop sequences of evolving gestures by combining movements. By harnessing the expressivity of Strudel we were able to not only sequence the movements of the robot, but start to create more complex patterns through transformation and combination of the robot’s movement. This allowed a series of gestures to be performed and combined at different scales, allowing patterning of choice or probability.

As part of this work we presented a live coding choreographic performance at The International Conference on Live Interfaces in June 2022, organized by conference chair Adriana Sá and CICANT at Universidad Lusófona in Lisbon, Portugal.[13] During the performance McLean was in Sheffield and Sicchio in Chicago, both using the networked Flok editor to program movement and sound, while using video conferencing to engage with the Lisbon audience.

[13] For video documentation of the performance at the conference see www.youtube.com/watch?v=Lh8yAOT3WOM-&t=2810s. Figure 4 shows a robot at work during a similar performance at a 2023 conference.

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

In this telematic performance we introduced our live notation techniques first as a score for audience participation, introducing our robots only later in the piece. The piece began with us sharing our screen with the audience in Lisbon, projected onto the back wall of the largely empty stage of the concert hall. A camera was also set on the stage for us to view how the audience was moving, allowing us to live code their gestures. Different actions were highlighted, with the instructions patterned by our code read out loud by a text-to-speech synthesizer. These instructions were simple to begin with, for example "move your head 35% over 5 seconds." To accompany the choreography and audio directions, minimal, patterned musical rhythms were live coded by McLean.

We asked the audience to participate in performing these patterned movements, instructed via the code. To facilitate this, the performance began with simple instructions, building up into a structured sequence. But as the piece progressed, more complex movements were created, by composing the simple sequences together and transforming them with multiple patterning functions. The audience attempted to learn and perform the movements in real time as they grew in complexity. As the audience began to struggle to keep up with the live coded choreographic instructions, we revealed an additional window on the screen. In this square the robot was shown to the audience for the first time, following the same live coding instructions that had been given to the audience. In its particular way, the robot was able to continue to perform the actions as instructed even as the movements grew more complex.[14]

The interpretation of such code from human to robot feels very different when compared to from human to human. Some people were sitting and some were standing. Some did a gesture once and paused, while others looped the movement over and over. Some became single-minded in completing the instructions even when they became more and more complex and difficult to perform because there was no time to break instructions down into smaller pieces. Most humans gave up, sat back and watched as the piece progressed. However, the robot was able to take on each code change without difficulty. Its dance was not limited to its ability to keep up with the algorithmic changes in the system as the work progressed from sequences to patterns. While the notation remained relatively simple, it generated a complexity that only machine movement could execute live in real time.

But still, we humans were "in the loop" as choreographers of the robotic movements, responding to them by live coding. Within the narrative structure of the piece, in particular the building up of patterns, and the reveal of the robot joining the audience in executing the code, all the instructions for patterning movement were improvised by us as live choreographer-programmers. By placing ourselves as humans improvising with computational movement, rather than automating movement as a prepared script, we created and manipulated patterns throughout the piece, responding to each subjective moment in time. While the robot may highlight differences between human and machine performers, the live coding process demonstrates the human labor of notation, and the ways in which we can author and guide computation through a close, tacit understanding of movement.

The future of this work includes workshopping a notational system that allows for further computational real-time notations focusing on movement for the robot as well as movement for human dancers. We aim to continue jumping between fleshy and robotic movements to understand the relationship between them, and are already working to compare and contrast efforts between human and machine expression.

Developing Notation through Use

Our design process for the next iteration of our notation is now focused on the movements of each of the individual servo motors as curves over time, combined into whole gestures. Rather than a goal-directed approach, such as our initial experiment with inverse kinematics, we begin with

[14] We should note that the robot was better able to follow the choreography because the notation was designed specifically for it. It would have been impossible for the robot to execute instructions designed for a human body, such as balancing or jumping.

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

movement qualities of each individual motor, and then progress to combine those qualities to discover whole gestures that emerge when the three motors move together to form a gestalt, greater than the sum of its parts. These gestures then form the basis of our choreographic vocabulary.

The aim however isn't a fixed set of gestures that are then sequenced into a piece. Performing one gesture followed by another is one key way of combining gestures, but there are many more ways to combine them, and then to combine those combinations. For example, because a gesture is described as numbers changing over time in the code, it makes sense to combine two gestures by simply adding (or multiplying) successive numbers from each together. Or, the gestures could be segmented, and then interlaced, one with another. These may seem like abstract operations, but they are well grounded in the algorithmic underpinnings of the long history of craft, as illustrated in our examples comparing weaving and bitfield patterns. Combining otherwise simple elements opens up creative ground for choreographic exploration.

We also are interested in interactions between the human choreographer/performers and the robots within a performance. One way to facilitate this back and forth of movement is through the use of sensors to detect movement from the humans and map this onto the robots. We are developing the use of accelerometers to shape both the movement of the robot during the performance and the underlying tempo. This might affect the overall impulse of the robot's motion or just one of the three motors at a time, again allowing for combinatorial actions within the robot's choreography. This approach provides an opportunity to explore patterned movement not only on a single robot, but as live feedback between humans and robots.

Conclusion/Exhaustion

Ironically, automation is exhausting. As we already noted with reference to Bainbridge (1983), automated processes must be supervised by an expert, but the usual hands-off nature of automation means that an expert's vigilance wanes, and through disengagement, eventually their expertise wanes too. We have explored an alternative to automation, where live code as higher-order notation is placed in the same cybernetic loop as human and/or robotic movement. Here the code does not control the movement, but rather induces it by describing unpredictable (yet deterministic) patterned movements. The live coder responds to those movements by manipulating the code, to the point that the code follows from the movement, as much as the movement follows from the code.

What code does in this configuration is make an explosion of possibilities explicit. Language is combinational in that given a fairly small number of elements and an expressive syntax, there are a very large number of possibilities in which those elements can be arranged. When these language elements represent transformations of pattern, by combining those transformations together we find a practically inexhaustible supply of new patterns to explore.

When we share our live code manipulations then, we are sharing a kind of collective wealth that Mark Fisher argues capitalism is set up to block: "Real wealth is the collective capacity to produce, care and enjoy. [...] This is Red Plenty [...] Everything for everyone. All of us first" (Fisher 2018). We share code, and changes to it, as we explore the pattern that results. Anyone can take a snapshot of that code, and make a minor adjustment to make something totally new. Scarcity vanishes.

This generation of collective wealth from patterns is not new—it is the basis of much traditional craft. For example, a textile fragment such as the iron age card-woven fragment found in the salt mines of Hallstatt, Austria, may have a complex structure, but that structure can be "read" to reveal the movement-code used to construct it (Griffiths et al. 2022). So here we join a wider movement to reconnect contemporary technology with its basis in ancient craft, making both old and new culturally expressive technology more open to change.

Discussion of technology, craft, and exhaustion brings the industrial Luddite movement to mind. Live coders have been compared to Luddites for valuing human craft over automation in their practice (McLean 2017). Indeed, live coding has been characterized by Colombian psychologist Camilo Andrés Hoyos Lozano (2022) as a technology of *post-work* because of its particular relationship with

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press

automation, its free/open source ethos, and its capability to restructure rules as they are followed. Live coder and researcher Alejandro Franco Briones (2023) sees similar possibilities for live coding in resisting marketization and supporting emancipation.

But still, our project to date is limited by the conventional, off-the-shelf robotic technology that we use. This technology is expensive, reliant on both exploited labor and on supply chains made fragile by ongoing environmental collapse and resulting health emergencies. However, there is no shortage of electronics to be found in e-waste stockpiles, including motors. For our next steps then, we will look for ways to apply our work to repurposed servo motors, adjusting to an open hardware approach. Or if all else fails, we can still return to apply our notations solely to human movement.

References

Bainbridge, Lisanne. 1983. "Ironies of Automation." *Automatica* 19, 6:775–79. doi.org/10/fhdj8w

Blackwell, Alan F., Emma Cocker, Geoff Cox, Alex McLean, and Thor Magnusson. 2022. *Live Coding: A User's Manual*. The MIT Press. doi.org/10.7551/mitpress/13770.001.0001

Cocker, Emma. 2014. "Live Notation: Reflections on a Kairotic Practice." *Performance Research Journal* 18, 5:69–76. doi.org/10.1080/13528165.2013.828930

Cocker, Emma. 2017. "Weaving Codes/Coding Weaves: Penelopean Mêtis and the Weaver-Coder's Kairos." *TEXTILE* 15, 2:124–41. doi.org/10.1080/14759756.2017.1298233

deLahunta, Scott. 2002. "Duplex/ ChoreoGraph: in conversation with Barriedale Operahouse." www.sdela.dds.nl/sfd/frankfin.html

Eacho, Douglas. 2021. "Scripting Control: Computer Choreography and Neoliberal Performance." *Theatre Journal* 73, 3:339–57. doi.org/10.1353/tj.2021.0071

Fisher, Mark. 2018. *k-punk: The Collected and Unpublished Writings of Mark Fisher (2004–2016)*, ed. Darren Ambrose. Repeater.

Franklin, Ursula. 1999. *The Real World of Technology*. 2nd ed. House of Anansi Press.

Franco Briones, Alejandro. 2023. "Towards Another Transdiscipline: Art, Science and Emancipation as a Promise and Provocation for Live Coding." Zenodo, 18 April. doi.org/10.5281/zenodo.7842097

Graham, Amanda Jane. 2016. "Space Travel: Trisha Brown's Locus." *Art Journal* 75, 2 (Summer). doi.org/10.1080/00043249.2016.1202621

Griffiths, David, Alex McLean, and Amber Griffiths. 2022. "Digital is physical & a remote tablet weaving exploration." Then Try This, 7 July. thentrythis.org/notes/2022/07/09/digital-is-physical-remote-tablet-weaving-explorations/

Harlizius-Klück, Ellen. 2017. "Weaving as Binary Art and the Algebra of Patterns." *TEXTILE* 15, 2:176–97. doi.org/10.1080/14759756.2017.1298239

Hugill, Andrew. 2020. "Shifting Meanings: The Fate of Words in Transdisciplinary Academia." January. andrewhugill.com/writings/Shifting%20meanings.html

Ingold, Tim. 2011. "The Textility of Making." In *Being Alive: Essays on Movement, Knowledge and Description*, 210–19. Routledge.

Laban, Rudolf von, and F.C. Lawrence. 1947. *Effort*. Macdonald & Evans.

Lozano, Camilo Andrés Hoyos. 2022. "Critical Subjectivity in Algorave's Post-Work Practices." *New Sociology: Journal of Critical Praxis* 3 (June). doi.org/10.25071/2563-3694.40

McLean, Alex. 2017. "Lessons from the Luddites." *Furtherfield* (blog), 2 June. www.furtherfield.org/lessons-from-the-luddites/

McLean, Alex, and Hester Reeve. 2012. "Live Notation: Acoustic Resonance?" In *Proceedings of International Computer Music Conference*, 70–75. Ljubljana, Slovenia. Zenodo. doi.org/10.5281/zenodo.7895652

Podmore, Jono. 2020. *Jaki Liebezeit: The Life, Theory and Practice of a Master Drummer*. Unbound.

Roos, Felix, and Alex McLean. 2023. "Strudel: Live Coding Patterns on the Web." In *Proceedings of the 7th International Conference on Live Coding*. Utrecht, Netherlands. Zenodo. doi.org/10.5281/zenodo.7842142

Schiphorst, Thecla. 1993. “A Case Study of Merce Cunningham’s Use of the LifeForms Computer Choreographic System in the Making of Trackers.” MA thesis, Simon Fraser University.

Seth, RajyaLakshmi. 2017. “Oral and Written Traditions in Documentation of Dance Notation in Indian Classical Dances.” Nehru Memorial Museum & Library. www.indianculture.gov.in/research-papers/oral-and-written-traditions-documentation-dance-notation-indian-classical-dances

Sicchio, Kate, and Alex McLean. 2017. “Sound Choreography <> Body Code: Software Deployment and Notational Engagement without Trace.” *Contemporary Theatre Review* 27, 3:405–10. doi.org/10.1080/10486801.2017.1343244

Sicchio, Kate, Patrick Martin, Charles Dietzel, and Alicia Olivio. 2022. “Towards A Framework for Dancing Beyond Demonstration.” In *MOCO ’22: Proceedings of the 8th International Conference on Movement and Computing*. ACM Digital Library. dl.acm.org/doi/10.1145/3537972.3537981

Sulzman, Mona. 1978. “Choice/Form in Trisha Brown’s ‘Locus’: A View from inside the Cube.” *Dance Chronicle* 2, 2:117–30. doi.org/10.1080/01472527808568723

TDReadings

Daly, Ann. 1988. “Movement Analysis: Piecing Together the Puzzle.” *TDR* 32, 4 (T120):40–52. doi.org/10.2307/1145888

van Hensbergen, Rosa. 2019. “*Dance X Fase X Quad*: Choreographic Seeing in Lucinda Childs, Anne Teresa de Keersmaeker, and Samuel Beckett.” *TDR* 63, 3 (T243):108–27. doi.org/10.1162/dram_a_00859

Whatley, Sarah. 2017. “Transmitting, Transforming, and Documenting Dance in the Digital Environment: What Dance Does Now that It Didn’t Do Before.” *TDR* 61, 4 (T236):78–95. doi.org/10.1162/DRAM_a_00693

https://doi.org/10.1017/S1054204323000576 Published online by Cambridge University Press