ECHO

a journal of music, thought and technology

ORPHEUS INSTITUUT

Title:	*Work in Progress*
Subtitle:	The Background and Development of the !trumpet
Author(s):	Nicolas Collins
Issue:	#8
Publication date:	
Review status:	
License:	CC BY-NC-ND 4.0
Article DOI:	

Abstract

The !trumpet ("not-trumpet") is a software synthesis system controlled from, and playing back through, a trumpet. The player produces no acoustic sounds via the mouthpiece. Instead, breath pressure and valve movement are read by an embedded microcontroller and sent to a laptop, where the data is mapped onto various parameters in software; the resulting electronic sound plays through a loudspeaker inside the bell, and is further processed acoustically by valving (changes in the length of tubing filter the speaker output), movement of a plunger mute (wah-wah style filtering), and orientation of the instrument in space. This essay describes the backround and evolution of the !trumpet from conception, through construction, to adaptations in response to performance experiences.

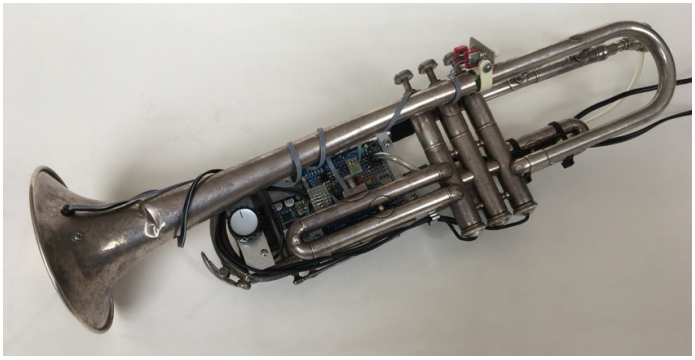

Figure 1: The *!trumpet*.

Work in Progress

I began my involvement in electronic music in the early 1970s, when I was seventeen, at a time when a Moog synthesizer literally cost as much as a car. The emergence of integrated circuits opened the door for cash-strapped, non-engineer, would-be instrument builders like myself — especially if we were willing to accept a Cage-tinged aesthetic of unexpected outcomes from poorly designed circuitry, soon to be legitimized under the rubric of "the circuit as score".[1]

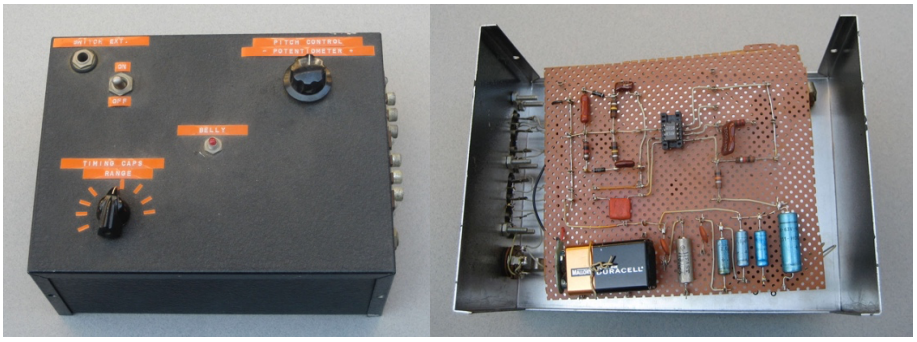

Figure 2: My First Circuit, Signetics 566 oscillator, 1972.

The next critical breakthrough in experimental self-made instruments came with the introduction of the microcomputer in the late 1970s. The microcomputer gave us a portable, pretty cheap package that combined the essential characteristics of an **instrument** (sound generation), a **score** (its memory allowed sequential programming) and a **performer** (decision making, branching). Very powerful, if somewhat non-tactile, the early computers did some of those things better than others.

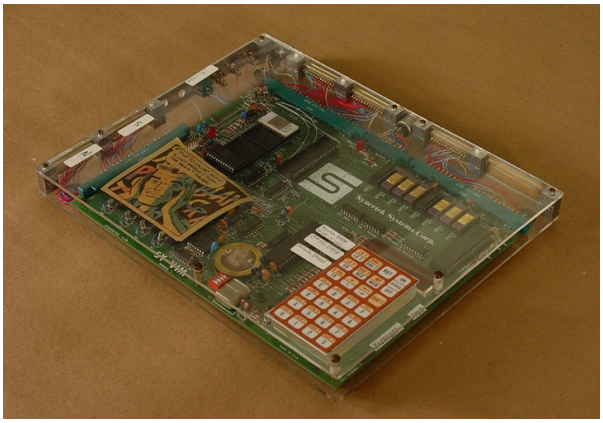

Figure 3: My First Computer, Synertek VIM, 1978. Photo:Simon Lonergan

From this point forward I divided my musical resources into three distinct categories, according to what I perceived as their "intrinsic" strengths:

- Hardware (circuits and physical instruments) was great for nuanced control, interesting sounds, and instability.
- Software was best for making logical decisions and generating compositional structure.
- People made choices (not always predictable) based on musical assessment and personal preference.

Over the years since then I've mixed hardware, software and humans in different ratios as appropriate for specific projects. The !trumpet is rooted in two threads that emerged from this resource mashup.[2]

Thread #1: trombone-propelled electronics

I built my trombone-propelled electronics in 1987 to transform recorded brass band music from the Peruvian altiplano, in a composed piece for live performance. A handmade DSP system was controlled from a keypad on a trombone slide that was coupled to a shaft encoder via a retractable dog leash; the keypad and encoder together acted like a mouse to click and drag program parameters. The electronic sounds played back through a speaker affixed to the mouthpiece, for further acoustic filtering by movement of the slide and mute. I included a breath control for volume articulation.

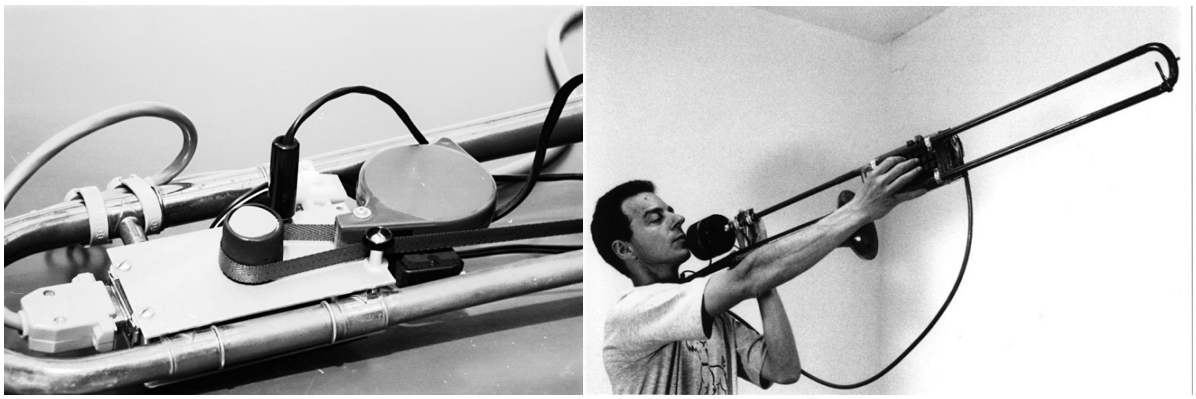

Figure 4: Trombone-propelled electronics, showing retractable dog leash attached to optical shaft encoder (left) and keypad on slide (right). Photos: André Hoeksema.

Audio processing was accomplished by a seriously hacked early digital reverb, controlled from an embedded Commodore 64 computer. The program auto-booted from an EPROM, so no keyboard or monitor was needed after programming (pre-laptop days). An amplifier for the trombone speaker sat on top.[3]

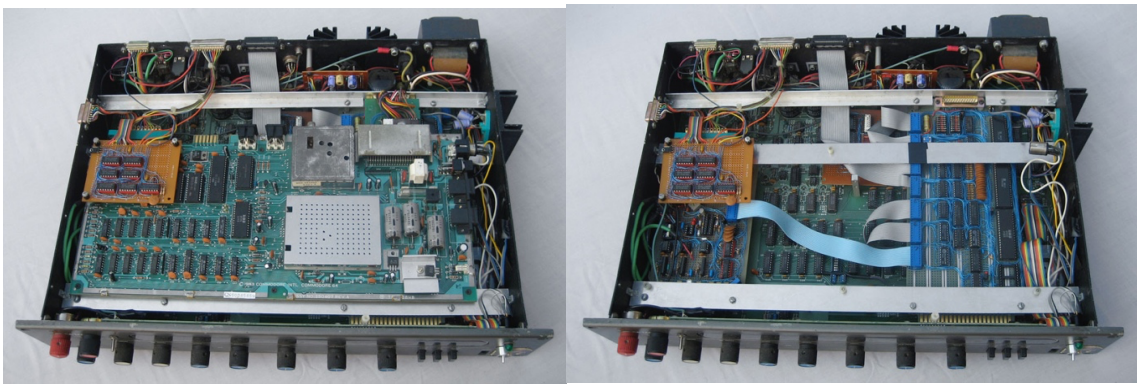

Figure 5: Hacked Ursa Major Stargate digital reverb, with Commodore 64 motherboard in place (left) and removed to show interface circuitry (right).

Figure 6: Stargate under Bryston amplifier for driving speaker in trombone.

The instrument's first application was in the composed work Tobabo Fonio (1987).[4] But, as an early live sampler with an "acoustic" output through its bell, the chameleon-like trombone-propelled electronics proved very adaptable to live improvisation with acoustic instruments, and it opened my access that world.[5]

Thread #2: Hardware Hacking

When I started teaching at an art school in 2000, my digitally-saturated students encouraged me to design a course in "making electronic music without computers": an old idea current once again, as witnessed by the rise of Circuit Bending, the revival of interest in vintage analog synthesis, and the introduction of euro-rack modular systems. Despite my minimalist roots (I had been a student of Alvin Lucier in the 1970s), twenty novice hackers pursuing similar designs -- each hacker playing through their own speaker -- immersed me in a form of semi-controlled audio chaos that I found engrossing, liberating, and evocative of my earliest circuit-based works.[6]

Figure 7: Hacking workshop, Crest, France, September 2023.

I began composing pieces based on the workshop ambience. In Salvage (2008), six players resurrect a dead circuit board pulled from the garbage. I built six simple oscillators; the pitch of each is determined by whatever lies between the two probes connected to it. In a "normal" oscillator design

this would take the form of some kind of resistor, but here the probes link unpredictable arrays of unknown components on the unseen underside side of a discarded circuit board (resistors, but also capacitors, diodes, chips, etc.).[7]

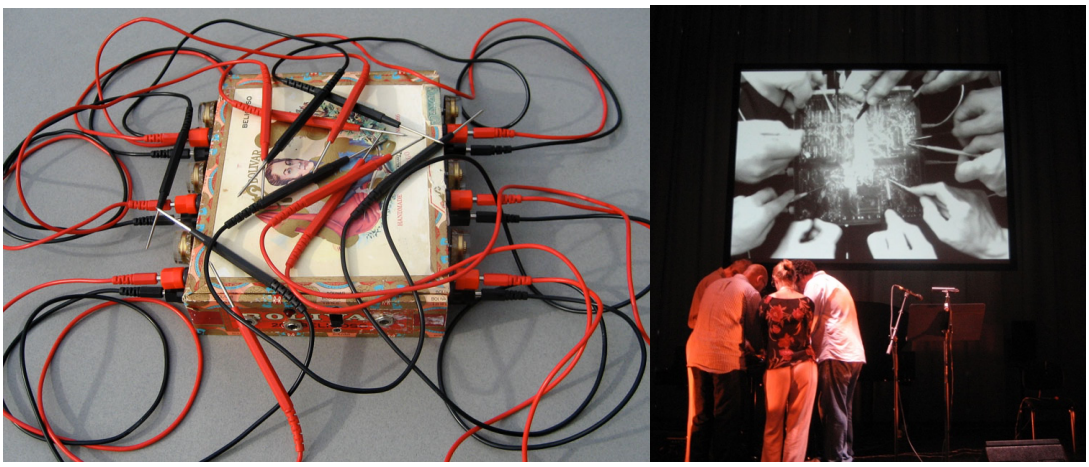

Figure 8: *Salvage* (2008), circuitry (left) and performance in Porto, Portugal, 2009 (right).

In The Royal Touch (2014) fishing weights under the fingers of a solo performer make nudgeable contacts between a similar bank of oscillators and a found circuit board.[8]

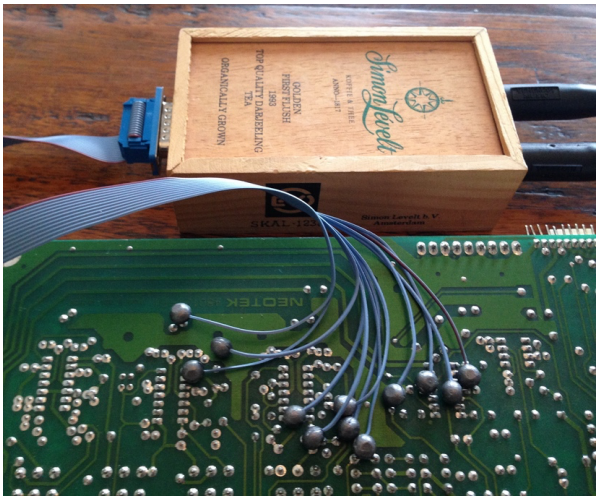

Figure 9: *The Royal* Touch (2014). Oscillators inside the wooden tea box at the top of the frame.

By 2006 looping and live sampling had become pedalaciously ubiquitous. I abandoned my trombone-propelled electronics out of saturation and boredom. Having no other system suitable for collaborative improvisation, I took a break from working in that domain. But by ten years later I missed the activity enough that I began plotting an instrument that would get me back on that stage, one specifically designed for improvisation rather than the execution of pre-composed music.

With the chaos of the workshops in my ears I decided to write a program that would behave less like software as I knew it and more like unstable circuitry in the hands of hackers. I started by emulating the behavior of the circuits in Salvage and The Royal Touch:

- Disjunctive transitions, rather than linear ones, in imitation of the jumps between values in response to random connections.
- Glissandi at different rates between values -- a function of capacitor charge and discharge times in analog circuitry.
- Drift, rather than stability, when a value is held.

I opted to use no sampling whatsoever, only synthesis (my first foray into this technique in thirty years).

I incorporated my circuit-emulation software into a hybrid instrument optimized for improvisation. Building on my experience with the trombone-propelled electronics, I began with an armature that provided acoustic presence and familiar physical gestures: a trumpet with a built-in speaker and a breath control. (Easier to fit into an overhead bin than a trombone.)

Figure 10: !trumpet.

Playing the valves filters the speaker's sound acoustically, as did the slide on the earlier trombone-propelled electronics. A tube leads from the mouthpiece to a pressure sensor, for breath articulation of the sounds.

Each of the three valves has a magnet at the bottom of its piston; its position is detected by a Hall-effect magnetic field sensor at the base of its cylinder, effectively creating a slide fader, for a total of three independent continuous controllers. (The trombone slide provided only one.)

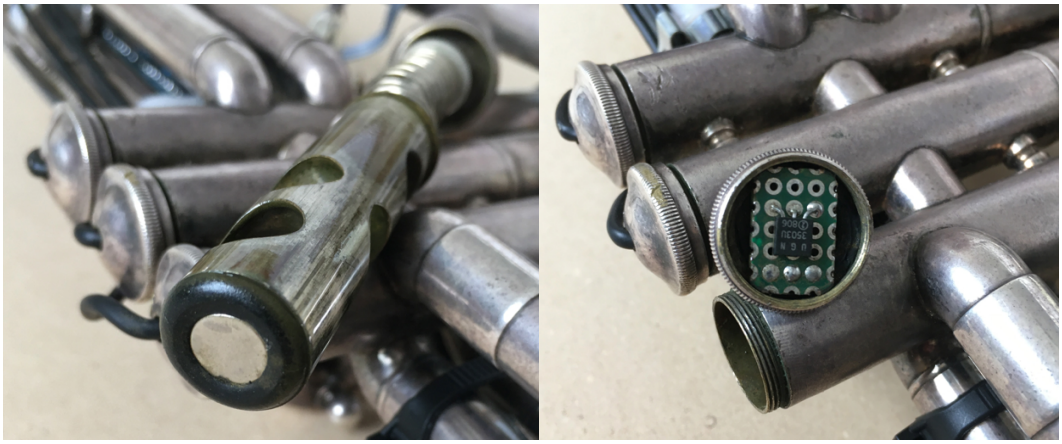

Figure 11: valves, showing magnet at base of piston (left) and Hall-effect sensor in base (right).

The three valve sensors are read by an Arduino on the trumpet and sent via a USB cable to a Macbook running software programmed in Max/MSP. To emulate disjunctive circuit transitions (described above) the linear output of each valve-fader is mapped to a look-up table that is randomly generated when the instrument is turned on – as if a piano's linking of keys to strings is scrambled every time the keyboard lid is opened, rather than conforming to the expected low-to-high linear sequence, but remains fixed, discoverable and learnable, until it is closed again.

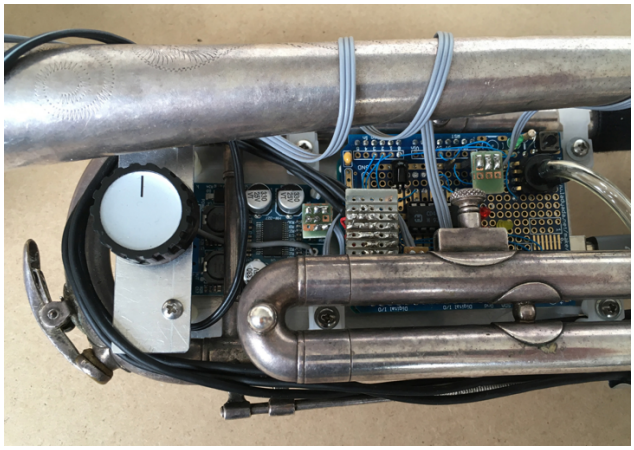

Figure 12: Arduino under interface board, pressure sensor for breath control visible at the upper right.

Each valve controls pitch or a similar primary parameter of one of three voices – oscillators or filtered noise sources – that make up a preset modeling a different circuit array. Presets are selected via switches installed on a toilet-plunger mute (also used for acoustic filtering of the speaker), whose closures are sent the short distance to the Arduino via an infrared link (from a TV remote control), and then on to the Macintosh. The player cannot select a specific preset directly: a "next" switch calls up a random preset; an "undo" switch steps back to the previous setting (CMD-Z, one of the great gifts of modern computing).

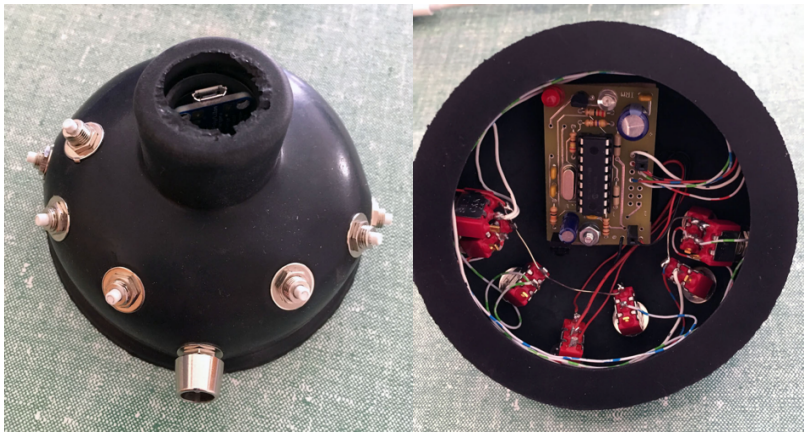

Figure 13: mute, showing embedded infrared transmitter circuit (right).

Other functions controlled by the mute switches include toggling between the internal speaker and a line output to the PA when a louder, bassier sound is desired; a "circular breathing" switch to freeze volume; and other software parameters.[9]

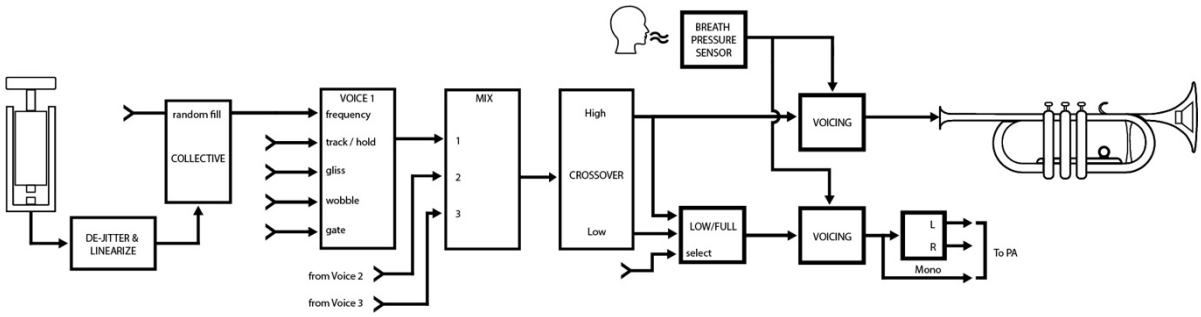

Figure 14: !trumpet software flow chart.

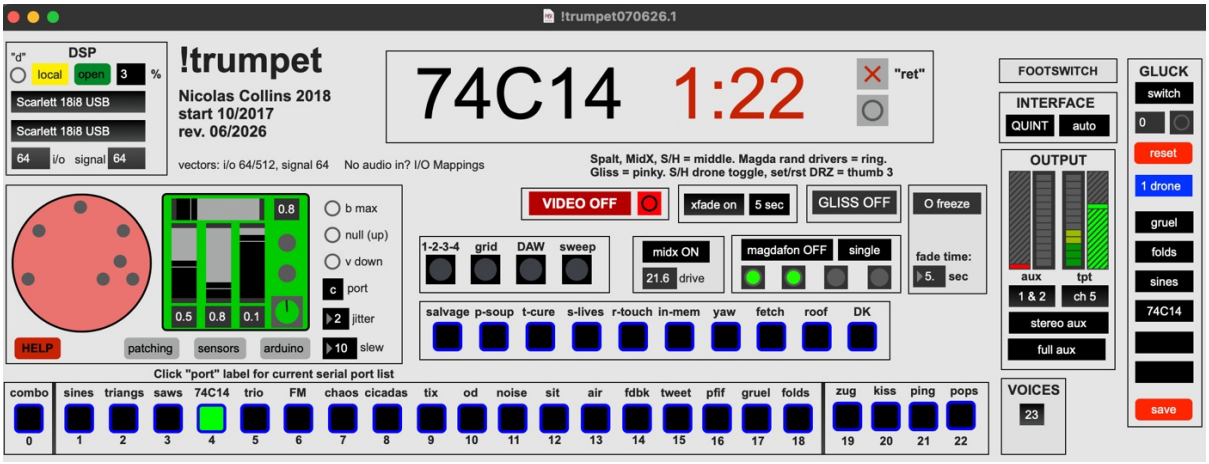

Figure 15: Screenshot of !trumpet Max/MSP program.

From its first appearance on stage in 2018, the !trumpet proved a very effective instrument for both solo and small-group improvisation. It surprised me with its unpredictability, while rewarding me with it performability. And the !trumpet has developed further through the process of playing. My initial design strategy -- limiting the Arduino's function to collecting sensor data and sending it on to a laptop running a higher-level software shell -- simplified the process of adding or subtracting features (the introduction of new voices and retirement of old ones) while preserving a consistent relationship between actions performed on the physical instrument (movement of valves, response to breath, mapping of switches) and their sonic effect. This control structure was optimized in the early months of transitioning from design to performance, and has undergone only minor tweaks subsequently — probably because it links to only a small number of actuators, and most are modeled on mechanical and kinesthetic features of a time-tested instrument, the trumpet.

Thanks largely to this two-part design strategy, the !trumpet has expanded sonically while maintaining enough instrumental consistency that practice makes, if not perfect, better.

That said, playing alongside performers of more-or-less conventional instruments (those whose histories extend back further than ten years) reveals the Achilles heel of many newly-invented ones: on multiple occasions I've felt that my hours of clever programming were shut down by a throwaway gesture on the part of a fellow performer. Humiliated, I would curl up and whimper in the corner of my studio until, after few miserable days, I'd mutter, "I think I can approximate that one sound that Birgit/Axel/Ute made just before the end of our first set", and I'd get to work programming a simulacrum in homage.

From its first appearance on stage in 2018, the !trumpet proved a very effective instrument for both solo and small-group improvisation. It surprised me with its unpredictability, while rewarding me with it performability.[10] And the !trumpet has developed further through the process of playing. My initial design strategy -- limiting the Arduino's function to collecting sensor data and sending it on to a laptop running a higher-level software shell -- simplified the process of adding or subtracting features (the introduction of new voices and retirement of old ones) while preserving a consistent relationship between actions performed on the physical instrument (movement of valves, response to breath, mapping of switches) and their sonic effect. This control structure was optimized in the early months of transitioning from design to performance, and has undergone only minor tweaks subsequently — probably because it links to only a small number of actuators, and most are modeled on mechanical and kinesthetic features of a time-tested instrument, the trumpet.

Thanks largely to this two-part design strategy, the !trumpet has expanded sonically while maintaining enough instrumental consistency that practice makes, if not perfect, better.

That said, playing alongside performers of more-or-less conventional instruments (those whose histories extend back further than ten years) reveals the Achilles heel of many newly-invented ones: on multiple occasions I've felt that my hours of clever programming were shut down by a throwaway gesture on the part of a fellow performer. Humiliated, I would curl up and whimper in the corner of my studio until, after few miserable days, I'd mutter, "I think I can approximate that one sound that Birgit/Axel/Ute made just before the end of our first set", and I'd get to work programming a simulacrum in homage.[11] As a result, the two dozen presets currently in use divide between modeling

unstable circuit networks – my starting point -- and emulating glimpses of my favorite musicians. Voices are weeded out when I get bored, new ones are added when inspired, the instrument evolves and remains fresh. There are sonic similarities between these two groups of voices: they all share a certain "noisiness", for want of a better word — more Axel Dörner than Bobby Hackett. While trumpet player Birgit Ulher, a frequent duo partner, has observed that electronic sounds have inspired her sonic palate (as they have other contemporary instrumentalists), her acoustic playing has in turn found its way into my programmed sounds.[12] Give and take.

Figure 16: Nicolas Collins and Birgit Ulher performing at the Blurred Edges festival, Hamburg, 2022. Photo:Gunnar Lettow.

[1] See E. Tomás, "Composing the instrument: Revisiting the concept of the instrument's score," Organised Sound, pp. 1–10, 2026. doi:10.1017/S1355771825101015. And Y. Nakai, *Reminded by the Instruments – David Tudor's Music*, Oxford University Press, 2021. ISBN: 9780190686765.
[2] See N. Collins, *Semi-Conducting -- Rambles Through The Post-Cagean Ticket*, Bloomsbury Academic, 2025.
[3] N. Collins, "Low Brass: The Evolution of Trombone-Propelled Electronics", *Leonardo Music Journal* Vol. 1, 1991.
[4] Video link: https://www.youtube.com/watch?v=89jbl0ZuaH4
[5] *100 of the World's Most Beautiful Melodies*, Trace Elements Records CD, 1989. 42 improvised duets with fifteen musicians.
[6] N. Collins, *Handmade Electronic Music – The Art of Hardware Hacking*, third edition, Routledge, 2020. First edition published by Routledge in 2006. Japanese edition by O'Reilly Japan, 2013. Korean edition by Habit Media, 2016.
[7] Performance in Tokyo, Japan, 2009: https://www.youtube.com/watch?v=NKC6OioEqp0. Studio recording at EMS, Stockholm, Sweden, 2015: https://www.youtube.com/watch?v=XV50-Cwy1RI.
[8] Studio recording at EMS, Stockholm, Sweden, 2015: https://www.youtube.com/watch?
[9] For a more detailed technical description see N. Collins, "The Development of !trumpet", *Musica/Tecnologia (Music/Technology)* 15, 2021. Much of the early stages of the development of this instrument took place during my residencies at the Orpheus Instituut, whose generous support is gratefully acknowledged.
[10] Lucky Dip (2020) JavaScript web app shuffling videos of fifty solo improvisations with the !trumpet: http://www.nicolascollins.com/LuckyDip.htm
[11] The notion of expanding one's performance options by modifying the instrument itself, rather than one's playing technique, is a noteworthy feature of digital luthiery. This concept has a precedent in re-patching a

synthesizer, or configuring a new arrangement of individual percussion instruments in a drum set, for example. But the relative ease, zero cost, and "undo" features of software editing add to its appeal.

[12] See https://relativepitchrecords.bandcamp/album/spark-gap