THE MATTER STACK

Complete Design & Build Compendium

A layered physical–acoustic–photonic display — from a $332 desktop build to a hardened, collaborative museum instrument

---

Original build guide · illustrated 4×4 → 8×8 → MATD build manual
art & expressive-medium addendum · choreography-engine software spec
V3 museum hardening · V4 collaborative knob/slider exhibit · full parts list with pricing

Compiled July 2026

Contents

How to use this compendium. Parts I–II get a machine working; Part III is the creative direction; Part IV is what you code; Part V hardens it for the public; Part VI is the collaborative endpoint; Part VII is what you buy. Page numbers refer to this combined document.

The Matter Stack

A Layered Physical–Acoustic–Photonic Display: Concept, Build Guide, and Holodeck Roadmap

Design document and step-by-step construction plan — July 2026

1. Executive Summary

This document describes a three-layer tangible display — the Matter Stack — that combines a magnetically sculpted ferrofluid surface, an acoustic levitation layer, and a photonic detail overlay into one registered workspace. Each layer contributes exactly what its physics does best: matter holds graspable shape, sound moves fast and delivers mid-air haptics, and light carries unlimited visual detail. Because ultrasound is a pressure phenomenon and ferrofluid responds only to magnetic fields, the two force media coexist in the same volume without interfering, which is the enabling observation behind the whole design.

The document contains a complete beginner-appropriate build plan: a TinyLev-class acoustic levitator (the proven on-ramp to MATD-style displays), a 16-channel electromagnet grid driving a sealed ferrofluid cell, an integration stage, full parts and cost tables (core build approximately $235–330), safety guidance, and a candid analysis of what does and does not scale on the road to a holodeck.

2. The Matter Stack Concept

The stack is ordered by mass and timescale. The magnetic layer is the slowest and most substantial: a pool of ferrofluid over an addressable electromagnet grid forms persistent, genuinely touchable terrain — the body of a displayed object, changing over seconds and staying put when grasped. The acoustic layer is the dynamic middle: a phased ultrasonic field flies a millimetre-scale bead through the air above the fluid at metres per second, drawing volumetric annotations, cursors, and particles by persistence of vision, while the same field projects focused pressure points onto the user's skin for mid-air haptic feedback. The photonic layer is pure information: an eye-worn display, phone AR view, or projector paints texture, color, labels, and text onto both physical layers at effectively unlimited resolution.

Interaction follows the same division. The user grasps the fluid to move and mold the object; touches the air to operate acoustic controls, feeling a tactile tick at the moment of contact; and reads the photonic layer for data. Registration ties it together: the levitated bead is a bright, trajectory-known optical fiducial, so it can anchor the coordinate frame for the entire stack. A later depth camera closes the loop on the fluid layer, whose shape — unlike the bead's — is not pre-planned and must be sensed.

A note on scope: this build starts with a TinyLev-class levitator, which holds and slowly moves a bead in a standing wave, rather than a full MATD, which requires per-transducer phase control to fly the bead at up to 8.75 m/s. TinyLev is the recognized first step: it teaches transducer handling, drive electronics, and trap physics on the exact same components, and Section 7 describes the upgrade path to a true phased array. Attempting the phased array first is the most common way this project fails for a first-time builder.

3. Build Part A — The Acoustic Layer (TinyLev-class levitator)

This build follows the open-source TinyLev design by Marzo, Barnes and Drinkwater (Review of Scientific Instruments, 2017), with the authoritative step-by-step guide published on Instructables under 'Acoustic Levitator.' An authorized complete kit is sold by Makerfabs and is the recommended route for a first-time builder, since component sourcing and transducer quality are the main failure points. Budget one to three weekends.

A.1 Print the frame

1. Download the TinyLev STL files from the Instructables guide and print the two-bowl frame in PLA. Use a 0.4 mm or 0.6 mm nozzle with a brim and no supports; the frame prints in one piece.
2. Clean the transducer sockets with a small file so each transducer press-fits snugly. Optionally drill the small center hole in each bowl for a camera or needle access later.

A.2 Mark transducer polarity — the critical step

Transducers must all push and pull in phase. Manufacturer polarity markings are unreliable — builders consistently report that a large fraction arrive with inverted markings — so every transducer must be tested individually. One inverted transducer creates a dead spot; several make levitation impossible.

1. Load the polarity-test sketch from the guide onto the Arduino Nano. Connect one wire to pin A0 and one to GND.
2. Open the Arduino Serial Plotter. Touch the two wires to the transducer's legs and gently press the transducer face.
3. If the plotted signal deflects upward (or sits high), mark the leg connected to A0 as positive. If it deflects down, the legs are reversed — mark accordingly.
4. Mark the positive leg of every transducer with a paint pen before any soldering. Do this in one patient sitting; it is tedious and it is also the single highest-leverage hour of the build.

A.3 Populate and solder the arrays

1. Press transducers into both bowls with all marked (positive) legs oriented the same way per bowl, following the guide's orientation diagram. Fix each with a small drop of epoxy or superglue — hot glue is reported to fail.
2. Solder bus wires connecting all positive legs together and all negative legs together within each bowl. Keep joints low-profile so they do not shadow neighboring transducers.

3. Check with the multimeter in continuity mode: every positive leg to the positive bus, every negative leg to the negative bus, and no short between buses.

A.4 Drive electronics

1. Flash the TinyLev sketch to the Arduino Nano. It generates complementary 40 kHz square waves.
2. Wire the Nano's signal pins to the L298N inputs per the guide's schematic, and the L298N outputs to the two arrays — the marked leg of each array to the positive output of its channel.
3. Connect the 12 V supply through the power switch to the L298N. Power the Nano from USB or the L298N's 5 V regulator, but never both simultaneously.
4. Optional but valuable: with a two-channel oscilloscope, confirm both arrays are driven in phase; the guide describes the expected waveforms. If running above 12 V or for very long sessions, upgrade the L298N heatsink.

A.5 First levitation

1. Assemble the two bowls facing each other at the frame's designed separation and power on at 6 V for testing.
2. Place a 1–3 mm polystyrene bead into the central region using a scoop of teabag paper or fine metal mesh on a toothpick — acoustically transparent materials that do not collapse the standing wave the way fingers or tweezers do.
3. The bead should snap into a pressure node and hover. Multiple beads can be stacked in adjacent nodes, evenly spaced. If beads are ejected or oscillate wildly, reduce the supply voltage slightly; if they refuse to trap, try smaller or irregularly shaped beads and re-verify polarity of any suspect transducers.
4. Milestone demo: stable levitation for several minutes, plus slow vertical movement of the bead using the phase-shift feature of the driver code.

4. Build Part B — The Magnetic Layer (ferrofluid over an electromagnet grid)

Budget two weekends. Everything about ferrofluid rewards preparation: it is not dangerous, but it stains skin, clothing, plastics, and porous surfaces permanently. Work gloved, over plastic sheeting, and keep the fluid sealed except when filling the cell.

B.1 Build the sealed fluid cell

1. Prepare a saturated salt solution: dissolve table salt into distilled water until no more dissolves, then let it settle. The dense saline suspends the ferrofluid and stops it from wetting and staining the glass.

2. Fill the flat glass container about two-thirds with the saline, then inject 10–20 ml of ferrofluid with a syringe below the surface. It will pool as a glossy black blob.
3. Seal the lid. From here on, all interaction with the fluid is magnetic, through the glass — which is exactly the point.
4. Hand-test with a strong neodymium magnet under the cell: observe spiking, how sharply features form, and how fast the blob follows a moving magnet. This calibrates your intuition for the grid's achievable 'resolution.'

B.2 Assemble the electromagnet grid

1. Mount the 16 electromagnets face-up in a 4×4 grid at roughly 30 mm spacing on a rigid non-magnetic base (plywood or a 3D-printed jig), directly beneath where the cell will sit. The cell should rest 3–8 mm above the coil faces.
2. Label the coils 1–16 and route each coil's two leads to a screw-terminal strip. Neatness here saves hours of fault-finding later.

B.3 Drive electronics — protect the microcontroller

1. Solder or fit a flyback diode directly across every coil, reverse-biased (cathode/stripe to the coil's positive lead). Coils store energy; when a MOSFET switches off, the collapsing field produces a high-voltage spike that the diode safely recirculates. Skipping even one diode will eventually destroy the drive electronics.
2. Wire each coil through one MOSFET module: coil positive to +12 V, coil negative to the module's drain/load terminal, module ground common with the Arduino Mega ground.
3. Connect each module's signal input to a PWM-capable Mega pin. PWM is what turns 16 on/off magnets into a smooth, shapeable field — cross-fading duty cycles between neighboring coils moves the fluid bump continuously rather than in jumps.
4. Power the coils from the 12 V 10 A supply on a separate rail from the Arduino's supply, grounds tied together at one point.
5. In firmware, cap the maximum duty cycle at roughly 60–70 percent and consider a software thermal budget (for example, limiting how long any coil may exceed 50 percent). Holding electromagnets run hot at full duty; heat is this subsystem's main reliability enemy.

B.4 First terrain

1. Place the sealed cell on the grid. Energize a single coil at 50 percent duty: a smooth bump of fluid should rise above it.
2. Write the traveling-bump demo: cross-fade PWM between adjacent coils so the bump orbits the grid in a circle. This is the subsystem's milestone — it proves smooth spatial control, not just on/off spikes.

3. Characterize the response: how fast can the bump travel before it tears or lags? That measured speed becomes the magnetic layer's specification in the integrated system.

5. Build Part C — Integration and the Photonic Layer

1. Print or build a gantry that suspends the TinyLev horizontally 10–15 cm above the fluid cell, so the levitation zone sits over the terrain. Piezoelectric transducers are non-magnetic and the ferrofluid is deaf to ultrasound at these intensities, so the layers do not fight — but keep the Arduinos and driver boards ~15 cm from the strongest magnets as cheap insurance.
2. Two-layer milestone: a bead levitating above while a fluid bump tracks beneath it, both driven from one host computer sending commands to the two Arduinos over USB serial.
3. Add the photonic layer the inexpensive way first: a mini projector aimed at the cell projection-maps color and labels onto the fluid — the glossy black surface takes projected light strikingly well — or a phone AR app overlays graphics using a printed marker on the frame.
4. Full-stack milestone: one script animates all three layers in sync — for example, terrain rises, the bead circles the new peak, and the projector paints elevation colors on the fluid.
5. Later upgrade: add the depth camera above the cell so the system senses the fluid's actual shape and registers the photonic layer to measured, not commanded, geometry — the first step toward the closed-loop registration this architecture ultimately depends on.

6. Parts and Cost Lists

6.1 Acoustic layer

Item	Qty	Est. cost	Notes
10 mm 40 kHz ultrasonic transducers	76	$45–60	Buy ~76 to have spares; the type used in car parking sensors. Bulk packs are cheapest.
Arduino Nano (or clone)	1	$5–10	Generates the 40 kHz drive signal.
L298N dual motor driver board	1	$5–8	Acts as the amplifier for both transducer arrays.
12 V DC power supply (2 A) + switch	1	$10–15	Run at 6–9 V for testing; 12 V maximum.
3D-printed TinyLev frame (two bowls)	1	$5	Free STL files from the Instructables guide; ~0.4 mm nozzle, brim, no supports.
Hook-up wire, solder, heat-shrink, glue	—	$10	Hot glue is reported unreliable for transducer mounting; use epoxy or superglue.
Expanded polystyrene beads, 1–3 mm	1 bag	$5	The levitated 'pixels.' Must be under ~3 mm (half a wavelength).
Teabag paper or fine metal mesh + toothpick	—	$2	Acoustically transparent scoop for placing beads; fingers disturb the field.

Subtotal (scratch build)		**$85–115**	**Authorized Makerfabs complete kit: roughly $100–130 and removes all sourcing risk.**

6.2 Magnetic layer

Item	Qty	Est. cost	Notes
12 V electromagnets, 25 mm barrel	16	$35–50	Holding-type electromagnets; 4×4 grid at ~30 mm spacing.
Logic-level MOSFET driver modules (IRLZ44N or similar)	16	$15–25	One channel per coil; modules are far more beginner-friendly than bare MOSFETs.
Flyback diodes (1N4007 or 1N5819)	20	$5	One across every coil, reverse-biased. Non-negotiable — omitting them destroys the microcontroller.
Arduino Mega 2560 (or clone)	1	$15–25	Enough PWM-capable pins for 16 channels.
12 V 10 A power supply	1	$20–25	Coils draw ~0.3–0.5 A each; budget headroom and never run all 16 at full duty.
Ferrofluid (e.g., Ferrotec EFH-1), 50 ml	1	$25–35	Quality fluid resists clumping. Cheap fluid stains worse and performs worse.
Flat glass container / large petri dish + lid	1	$10	The fluid cell. Glass, not plastic — ferrofluid stains plastic permanently.
Distilled water + table salt	—	$3	Saturated salt solution suspends the ferrofluid and prevents glass staining.
Perfboard, screw terminals, 18 AWG wire	—	$10–15	Power distribution for 16 coil channels.
Nitrile gloves, plastic sheeting, paper towels	—	$8	Ferrofluid stains skin, clothes, and furniture. Treat it like ink that never dries.
Subtotal		**$145–200**	

6.3 Integration and tools

Item	Qty	Est. cost	Notes
3D-printed gantry / frame	1	$5–10	Holds the acoustic levitator 10–15 cm above the fluid cell.
Mini LED projector (optional)	1	$50–80	Projection-maps color and labels onto the glossy black fluid; a phone AR app is the free alternative.
USB depth camera (optional, later)	1	$60–150	Lets the system sense the fluid's actual shape for closed-loop overlay registration.
Tools if not owned: soldering iron, multimeter	—	$30–45	A multimeter is mandatory for transducer polarity testing.

Project total (core build, no optionals)		**$235–330**	**Assumes scratch acoustic build and no projector or depth camera.**

7. Upgrade Path: From TinyLev to a True MATD

TinyLev holds the bead in a standing wave and can move it slowly along one axis by phase shifting. A Multimodal Acoustic Trap Display requires individual phase control of every transducer so the trap can be steered anywhere in the volume thousands of times per second, flying the bead at up to 8.75 m/s while synchronized RGB LEDs color it. The established open route is the Ultraino platform and its successors (such as OpenMPD) from the same research group: a flat array of 64 or more transducers driven by an FPGA or a microcontroller with dedicated driver boards providing per-channel phase resolution. Expect $300–600 in parts, surface-mount soldering or pre-fabricated driver boards, and real firmware work. Everything learned in Build Part A — polarity discipline, transducer handling, drive electronics, trap behavior — transfers directly. Amplitude modulation on the same array then adds the MATD's remaining modalities: mid-air tactile pressure points and audible sound from the display itself.

8. Safety

Ultrasound: 40 kHz is inaudible but intense near the arrays. Do not place your ear close to a running levitator, limit continuous close-range hand exposure, and keep pets — whose hearing extends far higher than ours — out of the room during long sessions. Electrically the acoustic build is low-voltage and benign.

Magnetics: keep strong magnets and energized coils away from pacemakers and implanted devices, credit cards, and mechanical watches. Electromagnets run hot; mount them on a non-flammable base, enforce duty-cycle limits in software, and power down when unattended.

Ferrofluid: not acutely toxic, but treat it as a permanent stain hazard — gloves, sheeting, sealed cell. Never let it contact bare neodymium magnets directly; it is nearly impossible to remove and the suspension degrades.

Power: the 12 V 10 A supply can deliver enough current to melt thin wiring. Fuse the coil rail, use 18 AWG or heavier for power distribution, and double-check polarity before first power-up.

9. Scaling Analysis and Holodeck Roadmap

9.1 What does not scale — the physics walls

Force does not project. Both force media in the stack decay viciously with distance. Magnetic field strength falls off roughly with the cube of distance from a source, so a coil that sculpts fluid across a few centimetres would need MRI-class power to act across metres — at which point the field itself becomes the hazard. Acoustic traps degrade similarly: trap stiffness collapses with range, and the focal intensities already required at desktop scale would, scaled to room volume, raise genuine heating and hearing-safety concerns despite being ultrasonic.

The speed of sound is a latency floor. Sound crosses a 3.5-metre room in roughly ten milliseconds. Wall-mounted arrays therefore cannot maintain phase-precise, fast-reacting traps deep in the room; responsiveness degrades with distance and no engineering removes this limit.

The fast particle becomes a projectile. A millimetre bead at 8.75 m/s in a small guarded volume is harmless; room-scale imagery implies faster or heavier particles sharing space with human eyes. Safety architecture, not resolution, becomes the binding constraint.

Ferrofluid does not tile. A room-scale fluid surface means large volumes of expensive, staining liquid plus containment. The magnetic layer scales as furniture — tables, panels, consoles — not as floors and walls.

9.2 What scales beautifully

The photonic layer. Optics, displays, and compute ride industrial learning curves, while force media obey inverse-cube laws. This asymmetry is the central architectural fact of any holodeck: force will always be local and expensive; light will always be global and cheap. Room-scale systems must therefore concentrate force where hands actually are and let photons carry everything else.

9.3 The holodeck roadmap

Stage 1 — Zones, not volumes. Embed matter-stack workstations (exactly this build) at the loci of interaction: tabletops, wall niches, doorframe consoles. The photonic layer stitches the zones into one continuous scene. The room is an illusion of light with islands of genuine matter precisely where you touch it.

Stage 2 — Replace the bead with a swarm. At room scale, the flying bead's successor is a swarm of small LED microdrones: the same persistence-of-vision principle, but self-powered, so no force must be projected from the walls at all. This sidesteps the force-projection wall entirely and inherits the trajectory-broadcast registration trick — a drone's planned path, like the bead's, costs zero latency. Multi-particle acoustic research (electrostatic anti-collapse, learned trajectory stabilization) meanwhile grows the desktop bead count.

Stage 3 — Force that travels with the user. Wearable haptics — ultrasonic wristbands, gloves, ring transducers — keep the force source permanently centimetres from skin, the only place force is cheap. The room tells the wearable what to render; the wearable delivers it locally.

Stage 4 — Encounter surfaces. Mobile robots quietly pre-position physical props and actuated surfaces where the user is about to touch the virtual world, generalizing the magnetic layer's role: persistent matter, delivered just-in-time instead of everywhere-at-once.

9.4 Why this desktop build is the right first step

Not because it scales — it does not — but because its architecture is the one that scales: a persistent matter layer, a fast dynamic layer, and a photonic detail layer, mutually registered in real time around a shared fiducial. Every holodeck stage above is that same three-layer pattern with substitutions. The registration, synchronization, and content-splitting problems this build forces you to solve at a bench are the holodeck's actual research problems, purchasable today for a few hundred dollars.

10. References

Hirayama, R., Martinez Plasencia, D., Masuda, N., & Subramanian, S. (2019). A volumetric display for visual, tactile and audio presentation using acoustic trapping. Nature, 575, 320–323.

Marzo, A., Barnes, A., & Drinkwater, B. W. (2017). TinyLev: A multi-emitter single-axis acoustic levitator. Review of Scientific Instruments, 88, 085105. Build guide: instructables.com/Acoustic-Levitator (UpnaLab).

Marzo, A., et al. Ultraino: an open phased-array platform for airborne ultrasound; successor software platform OpenMPD.

Shi, S., Hübl, M. C., Grosjean, G., Goodrich, C. P., & Waitukaitis, S. (2025). Electrostatics overcome acoustic collapse to assemble, adapt, and activate levitated matter. PNAS, 122(50).

StableLev: Data-Driven Stability Enhancement for Multi-Particle Acoustic Levitation. CHI 2024.

The Matter Stack — Illustrated Build Manual

Part A: the 4×4 build · Part B: upgrading to 8×8 · Part C: the MATD phased-array path

Step-by-step assembly with diagrams · companion to the Matter Stack design documents — July 2026

This manual builds the layered physical–acoustic–photonic display in three stages. **Part A** gets a working single-workstation display: a TinyLev-class acoustic levitator over a 4×4 ferrofluid terrain grid, with a projected photonic overlay. **Part B** upgrades the terrain to a 64-coil 8×8 grid for smoother, higher-count shapes. **Part C** replaces the levitator's fixed standing-wave trap with a steerable phased array (a Multimodal Acoustic Trap Display). Do the parts in order — each reuses the skills and most of the hardware of the one before it.

> ⚠ Read the safety notes in each part before powering anything. The three recurring hazards: flyback diodes are mandatory on every coil; ferrofluid stains permanently (keep it sealed); and 40 kHz ultrasound is intense at close range.

Figure 1 — The three-layer stack (side view)

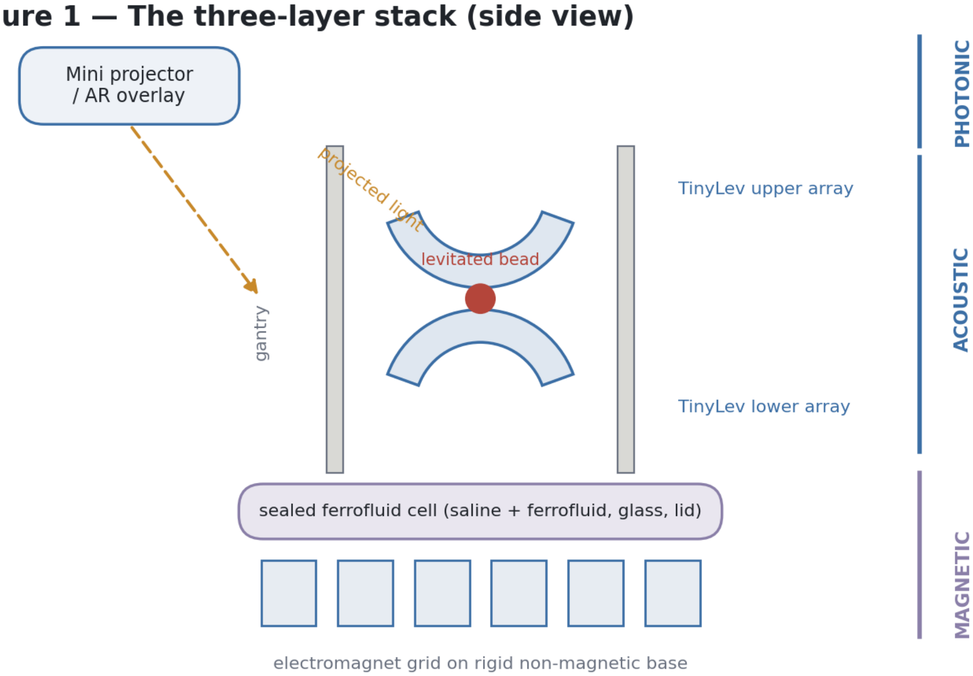


Figure 1 — How the three layers stack. Sound and magnetism share the same volume without interfering, which is what makes the display possible.

PART A Build the 4×4 Matter Stack

Budget three to five weekends and about $235–330. Build the acoustic layer first, then the magnetic layer, then integrate. Each subsystem has a milestone you should hit before moving on.

A1 · Acoustic layer — the TinyLev levitator

Follow the open-source TinyLev design (Marzo, Barnes & Drinkwater, 2017; step-by-step on Instructables as “Acoustic Levitator”). A first-time builder should consider the authorized Makerfabs kit to remove sourcing risk.

A1.1. Print the frame. Print the two-bowl TinyLev frame in PLA with a 0.4–0.6 mm nozzle, brim, no supports. Clean each transducer socket with a small file so transducers press-fit snugly.

A1.2. Test transducer polarity — the highest-leverage hour of the build. Manufacturer polarity marks are unreliable. Load the polarity-test sketch on the Arduino Nano, open the Serial Plotter, and test every transducer as in Figure 2. Mark the positive leg of each with a paint pen before any soldering.

Figure 2 — Transducer polarity test (do this for EVERY transducer)

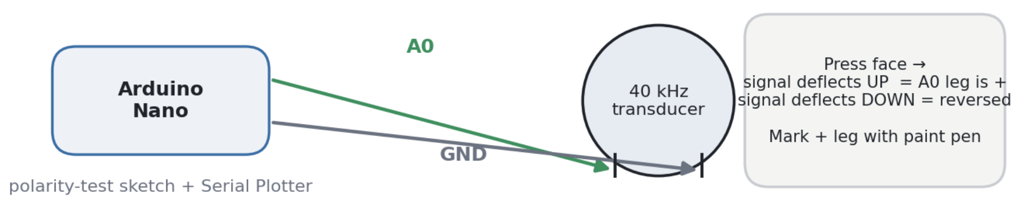


Figure 2 — Polarity test. One inverted transducer makes a dead spot; several make levitation impossible. Test all ~72.

A1.3. Populate and solder the arrays. Press transducers into both bowls with all marked (+) legs oriented the same way per bowl. Fix each with a small drop of epoxy or superglue (not hot glue). Bus all + legs together and all − legs together within each bowl; keep joints low-profile.

A1.4. Verify wiring. With a multimeter in continuity mode, confirm every + leg reaches the + bus, every − leg the − bus, and that there is no short between buses.

A1.5. Wire the drive electronics. Flash the TinyLev sketch (complementary 40 kHz square waves) to the Nano and wire it through the L298N to the two arrays, as in Figure 3. Power the L298N from the 12 V supply through the switch.

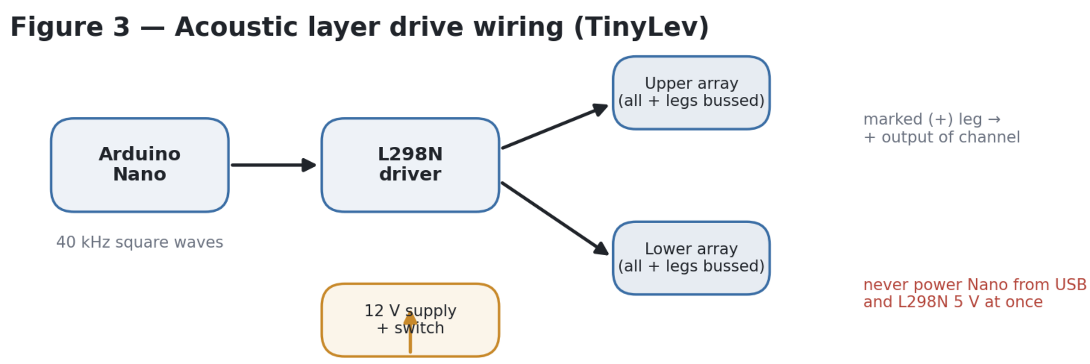


Figure 3 — Acoustic drive chain. Marked (+) leg of each array goes to the + output of its channel.

⚠ Power the Nano from USB **or** the L298N's 5 V regulator — never both at once.

A1.6. First levitation (milestone). Assemble the bowls facing each other at the designed separation and power on at 6 V. Drop a 1–3 mm polystyrene bead into the central region using a teabag-paper or fine-mesh scoop. It should snap into a pressure node and hover. Milestone: stable levitation for minutes, plus slow vertical movement via the driver's phase-shift feature.

A2 · Magnetic layer — ferrofluid over a 4×4 coil grid

Budget two weekends. Preparation is everything: ferrofluid stains permanently. Work gloved, over plastic sheeting.

A2.1. Build the sealed fluid cell. Make a saturated salt solution (dissolve table salt into distilled water until no more dissolves; let settle). Fill a flat glass container two-thirds with the saline, then inject 10–20 ml of ferrofluid below the surface with a syringe. Seal the lid. All interaction from here is magnetic, through the glass.

A2.2. Assemble the 4×4 grid. Mount 16 electromagnets face-up in a 4×4 grid at ~30 mm spacing on a rigid non-magnetic base (Figure 4). The cell rests 3–8 mm above the coil faces. Label coils 1–16 and route each coil's two leads to a screw-terminal strip.

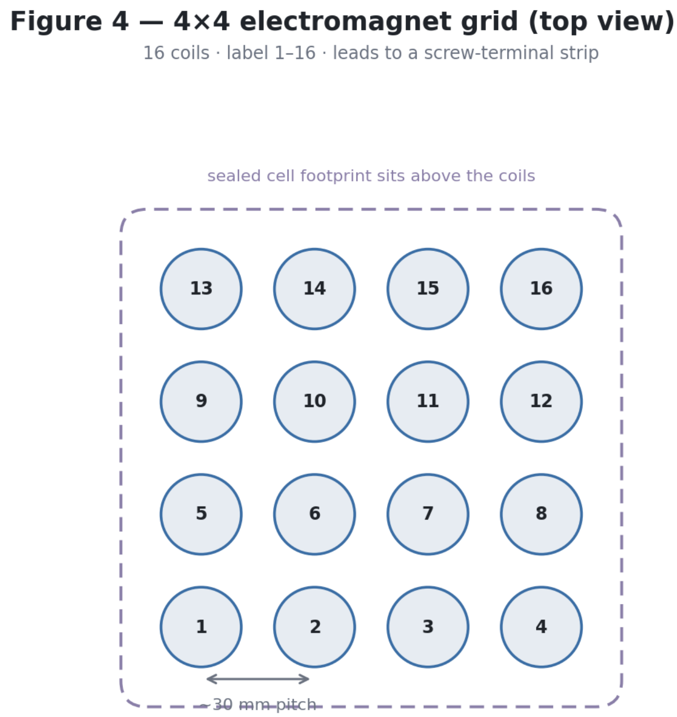
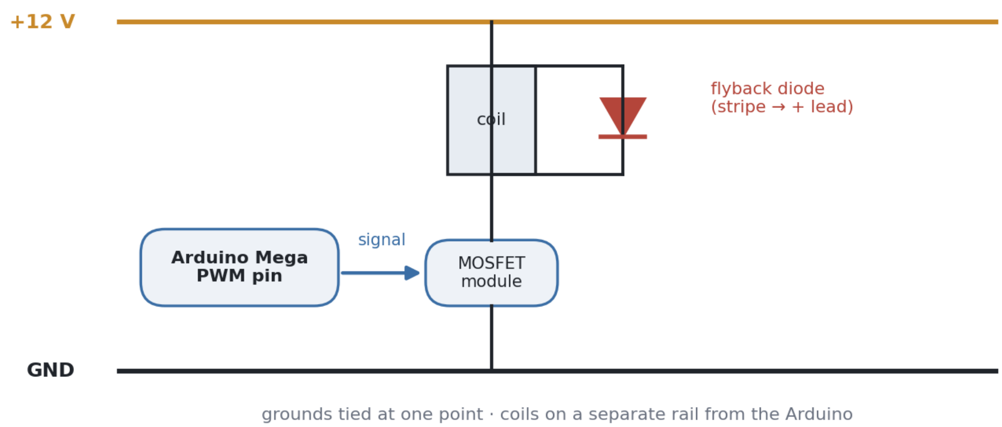


Figure 4 — 4×4 grid layout (top view). Neat labeling and lead routing saves hours of fault-finding.

A2.3. Fit a flyback diode across every coil. Solder a diode reverse-biased across each coil (stripe/cathode to the coil's + lead). This recirculates the voltage spike when a MOSFET switches off (Figure 5).

A2.4. Wire one MOSFET per coil. Coil + to +12 V; coil − to the MOSFET module's drain/load terminal; module ground common with the Arduino Mega ground. Connect each module's signal input to a PWM-capable Mega pin. Power coils from the 12 V 10 A supply on a separate rail from the Arduino, grounds tied at one point.

Figure 5 — One coil channel: MOSFET + flyback diode (repeat ×16)



Omitting even one flyback diode eventually destroys the drive electronics.

Figure 5 — One coil channel, repeated 16×. PWM cross-fading between neighbors moves a fluid bump smoothly instead of in jumps.

> ⚠ In firmware, cap duty cycle at ~60–70% and add a thermal budget (limit how long any coil exceeds 50%). Holding coils run hot — heat is this subsystem's main reliability enemy.

A2.5. First terrain (milestone). Place the sealed cell on the grid and energize one coil at 50% duty — a smooth bump should rise. Then write the traveling-bump demo: cross-fade PWM between adjacent coils so the bump orbits the grid. Measure how fast the bump can travel before it tears; that speed is the layer's spec.

A3 · Integration and the photonic layer

A3.1. Build the gantry. Suspend the TinyLev horizontally 10–15 cm above the fluid cell so the levitation zone sits over the terrain (Figure 6). Keep the Arduinos and driver boards ~15 cm from the strongest magnets.

A3.2. Two-layer milestone. From one host computer, drive both Arduinos over USB serial so a bead levitates above while a fluid bump tracks beneath it.

A3.3. Add the photonic layer. Aim a mini projector at the cell to projection-map color and labels onto the glossy fluid (or use a phone-AR app with a printed marker). Full-stack milestone: one script animates all three layers in sync — terrain rises, the bead circles the peak, the projector paints elevation color.

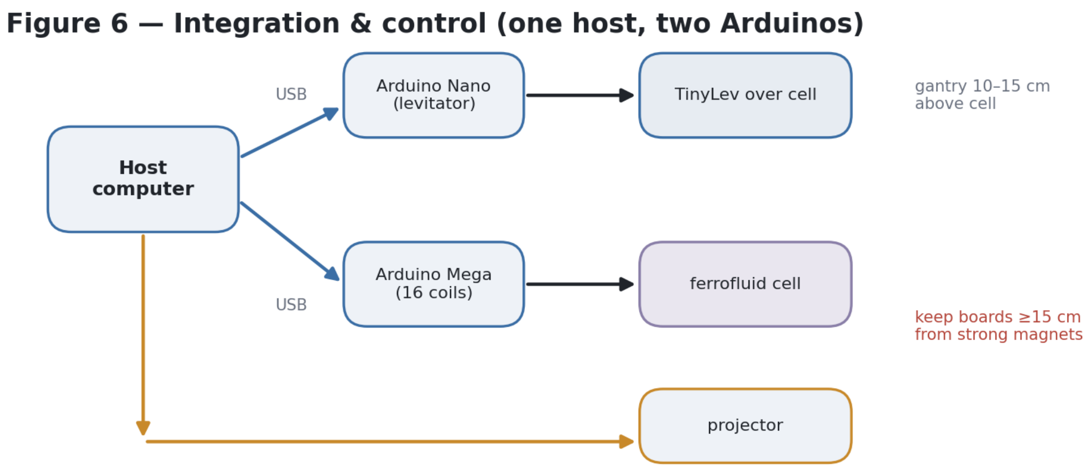


Figure 6 — Integration and control. One host drives two Arduinos; the projector maps light onto the cell.

A3.4. Optional closed loop. Add a depth camera above the cell to sense the fluid's actual shape and register the photonic overlay to measured, not commanded, geometry — the first step toward closed-loop registration.

PART B Upgrade the terrain from 4×4 to 8×8

This stage swaps the 16-coil grid for 64 coils at ~15 mm pitch, roughly doubling resolution per axis. The acoustic and photonic layers are unchanged. Three things change: the coil array, how 64 channels are addressed, and power/cooling. Budget roughly $180–320 in parts over the 4×4.

B1 · Build the 8×8 coil array

B1.1. Lay out 64 coils. Mount 64 electromagnets in an 8×8 grid at ~15 mm pitch on a new rigid non-magnetic base (Figure 7). Keep the cell standoff at 3–8 mm. The original 4×4 positions fall on every other row and column, which helps when reusing a jig.

B1.2. Fit shaped pole pieces (optional). A small shaped pole piece on each coil sharpens its field and buys a little extra separation between neighboring features. Going denser than 8×8 mostly wastes coils — below the fluid's natural spike spacing, peaks merge regardless of pitch.

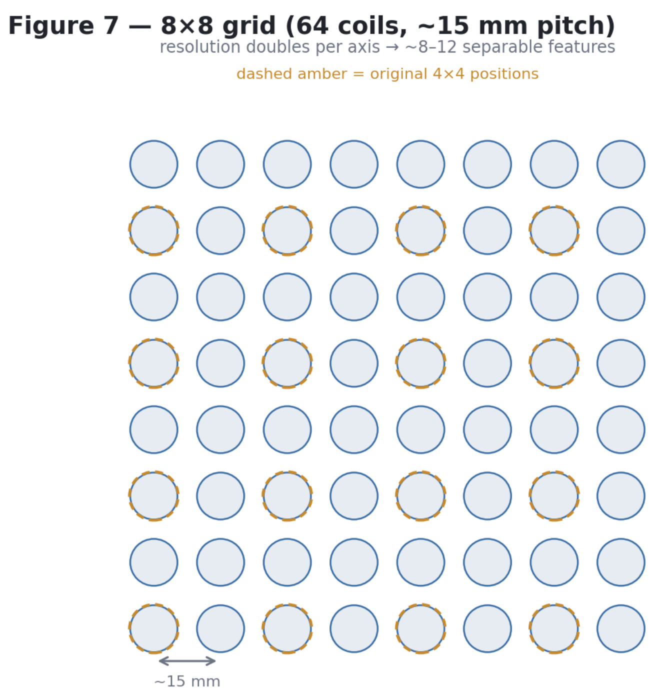


Figure 7 — 8×8 grid. Expect ~8–12 cleanly separable features; the ferrofluid's spike spacing, not coil pitch, sets the ceiling.

B2 · Re-address 64 channels with PCA9685 boards

An Arduino Mega has only ~15 hardware-PWM pins, so the drive path moves to dedicated PWM driver ICs.

B2.1. Chain four PCA9685 boards. Set their I^2C addresses to 0x40, 0x41, 0x42, 0x43 (solder the address jumpers) and connect all four to the Mega's single I^2C bus, SDA/SCL shared (Figure 8). 4 × 16 = 64 channels.

B2.2. Keep the per-coil MOSFET + flyback. Each PCA9685 channel still drives one logic-level MOSFET, and every coil still gets its own flyback diode. The board replaces the PWM *source*, not the power stage.

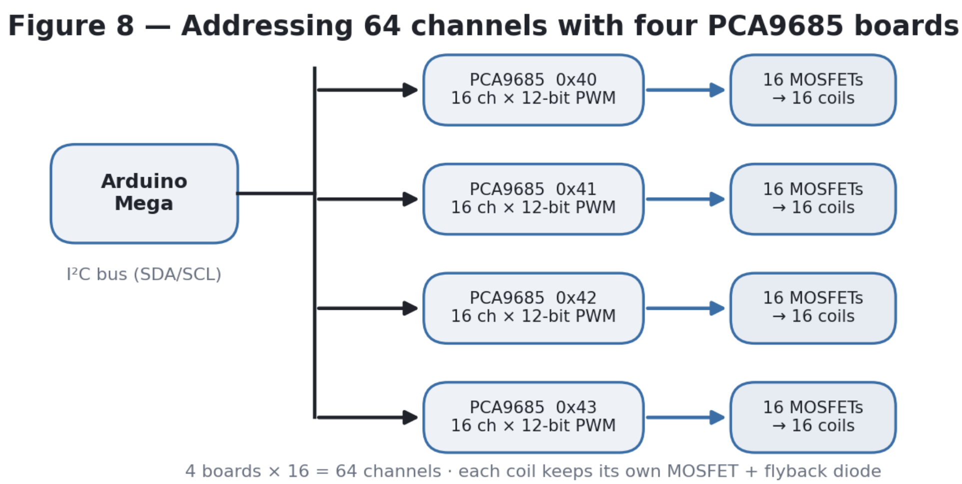


Figure 8 — Addressing architecture. Each coil keeps the MOSFET-plus-flyback channel from Figure 5.

> ⚠ Do not omit a single flyback diode when scaling up — with 64 coils there are 64 chances to destroy the drive electronics.

B3 · Power and cooling

B3.1. Size the supply. Sixty-four coils at 0.3–0.5 A each cannot all run at once. Fit a 12 V, 20–30 A supply, fuse the coil rail, and distribute power on 16 AWG or heavier bus bars (Figure 9).

B3.2. Add active cooling. Mount heatsinks on the coil bank and forced-air fans underneath; run the fans whenever coils are energized.

B3.3. Enforce the thermal budget in firmware. Cap duty at 60–70%, cap how many coils may run hot simultaneously, and add a per-coil dwell limit. At 4× the coil count, heat and power are the dominant constraints.

Figure 9 — Power distribution & cooling for 64 coils

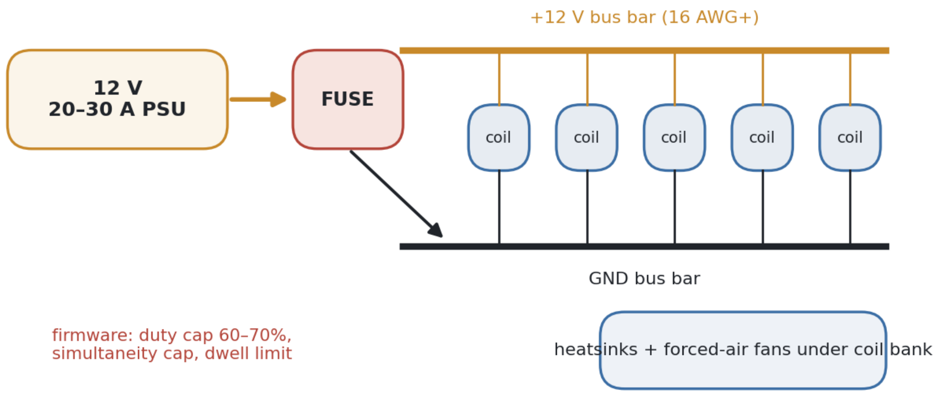


Figure 9 — Power distribution and cooling for the 64-coil bank.

B4 · Host and firmware changes

B4.1. Update the terrain output stage. The host now sends 64 duty values instead of 16; firmware maps each to its PCA9685 channel and scales to 12-bit PWM. Because authoring uses a normalized coordinate frame, existing scenes and demos run unchanged on the denser grid.

B4.2. Re-characterize. Repeat the traveling-bump test and re-measure tear-speed; the finer grid changes the achievable feature size and travel speed you can safely command.

PART C MATD upgrade — the phased-array path

The TinyLev holds a bead in a fixed standing wave and can only creep it along one axis. A Multimodal Acoustic Trap Display (MATD) gives every transducer its own phase, so the trap can be steered anywhere in the volume thousands of times per second — flying the bead at up to 8.75 m/s, coloring it with synchronized RGB LEDs, and (via amplitude modulation) adding mid-air haptics and audible sound. Expect $300–600 in parts, surface-mount or pre-fabricated driver boards, and real firmware work. Everything learned in Part A transfers directly.

> ⚠ This is an upgrade to attempt only after Parts A and B work reliably. Attempting a phased array first is the most common way the project fails for a first-time builder.

C1 · Replace the two bowls with a flat phased array

C1.1. Build or buy a flat array. Replace the facing-bowl geometry with a single flat array of 64–256 transducers on a rigid PCB or 3D-printed plate, mounted above the cell (Figure 10). The established open route is the Ultraino platform and its successor OpenMPD from the same research group.

C1.2. Reuse your polarity discipline. Every transducer still must be polarity-tested and oriented consistently — exactly the Part A procedure, at larger scale.

Figure 10 — MATD upgrade: single flat phased array replaces the two bowls

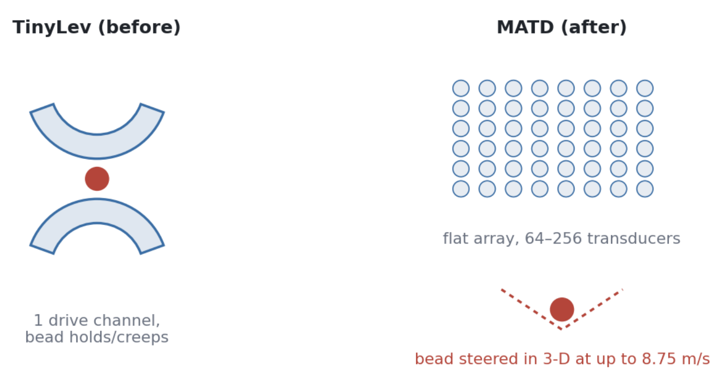


Figure 10 — From a two-bowl standing-wave trap to a flat steerable array over the cell.

C2 · Per-transducer phase control

C2.1. Add a phase controller. Drive the array from an FPGA or a microcontroller with dedicated driver boards that provide per-channel phase resolution (Figure 11). This is the core architectural change: TinyLev drove every transducer from one channel; MATD drives each independently.

C2.2. Bring up a static trap first. Before steering, reproduce a stationary trap with the array to confirm phase calibration, then command slow moves, then fast trajectories.

Figure 11 — Per-transducer phase control (Ultraino / OpenMPD class)

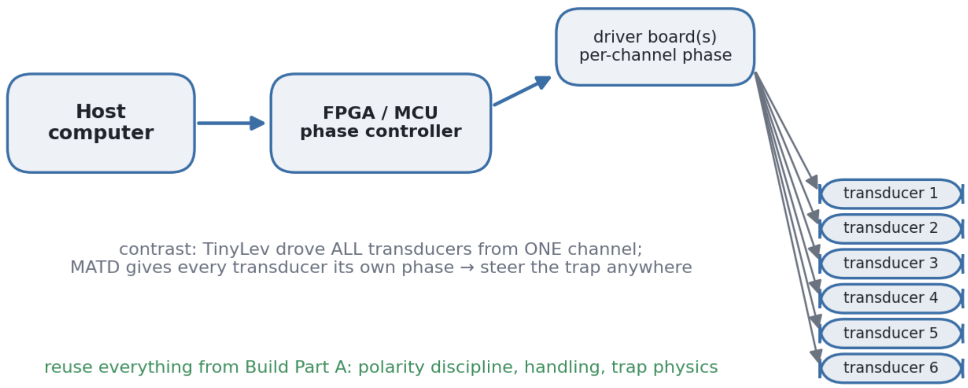


Figure 11 — Per-transducer phase control. Individual phase per transducer is what lets the trap move anywhere.

C3 · Unlock the MATD modalities

C3.1. Volumetric imagery. Fly the single bead fast enough that persistence of vision draws lines and shapes; synchronize an RGB LED to color it along the path.

C3.2. Mid-air haptics. Amplitude-modulate the same array to project focused pressure points onto a user's skin for tactile feedback.

C3.3. Audible sound. Amplitude-modulate in the audible band so the array itself emits sound — the display can voice its own soundtrack, no separate speakers.

Figure 12 — MATD modalities from one array

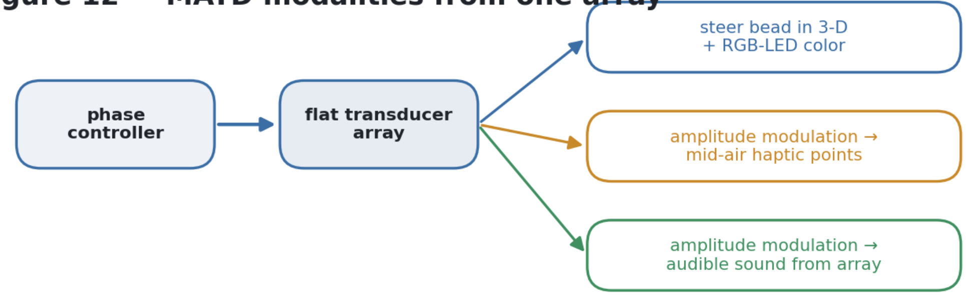


Figure 12 — One array, three added modalities: steered/colored imagery, mid-air haptics, and audible sound.

With Part C complete the stack is a full three-layer display: a persistent 8×8 ferrofluid matter layer, a steerable acoustic layer with haptics and sound, and a photonic overlay — mutually registered around the bead fiducial. That is the architecture the holodeck roadmap scales, purchasable at a bench today.

The Matter Stack — Living Instrument

Art & Expressive-Medium Addendum: an 8×8 sound-reactive matter display

Companion to The Matter Stack: A Layered Physical–Acoustic–Photonic Display · Design & build addendum — July 2026

1. Purpose and Scope

This addendum adapts the base Matter Stack build into an expressive art instrument rather than a data display or a science exhibit. It specifies three changes to the core guide: an upgrade of the magnetic layer from a 4×4 to an **8×8 electromagnet grid** for smoother, higher-count terrain; a **sound-reactive engine** that turns live room audio into synchronized motion of matter, sound, and light; and a **timeline/choreography system** that supports scripted “micro-theater” pieces. It closes with a staged roadmap that ends at a hardened, collaborative, museum-grade installation — the direction to build toward, not to build first.

Two creative modes share one machine. In **data-as-art** mode the installation listens to the room and renders the ambient soundscape as living terrain, a darting light, and shifting color — the crowd itself becomes the composer. In **micro-theater** mode a scripted score drives the three layers through a short narrative: terrain as landscape, the levitated bead as a character or star, light as sky and emotion. The two modes are not exclusive; the strongest pieces use a scripted scaffold that the room's sound modulates in real time.

> **The governing idea: temporal composition, not spatial resolution.** Even at 8×8 the terrain holds only a handful of independent features. Treat the system not as a low-resolution screen but as an instrument with a few slow “keys,” one fast moving voice, and a full palette of light — expressive through *sequence and timing*, the way most music is expressive with few notes. This reframing is what keeps a low feature count from reading as a limitation.

2. The Instrument: A Two-Speed Voice

The three layers divide naturally by timescale, and that division is the instrument's design. The **ferrofluid terrain is the slow voice** — features change over roughly 0.2–1 second, so it behaves like a pad or drone: evolving harmonies, swelling landscapes, textures that breathe. The **levitated bead is the fast voice** — it moves at metres per second, so it is the lead: quick melodic runs, accents, a point of light that darts and hovers. The **photonic layer is the palette** — projected color, intensity, and label, painting mood across both physical layers and carrying any detail the matter cannot.

Composing across these three at once, locked to a shared clock and a shared coordinate frame, is the actual art form here. No single-medium instrument can do it, and the registration problem it forces — keeping light aligned to matter that is moving — is the same problem the base guide identifies as the holodeck's real research question. Building this piece is therefore both an artwork and a genuine step down that road.

Why the bead is also the anchor

The bead's trajectory is known in advance because the drive code commands it, so it doubles as a bright, moving optical fiducial that anchors the whole coordinate frame at zero latency. The terrain's shape, by contrast, is an emergent physical result and must be *sensed* — which is why the depth camera (Section 4) matters for any piece where light must register precisely to the fluid rather than to commanded values.

3. Hardware Delta — From 4×4 to 8×8

The magnetic layer is the only subsystem that changes substantially. Moving from 16 coils at 30 mm pitch to 64 coils at roughly 15 mm pitch roughly doubles spatial resolution per axis, letting terrain features be smaller, closer, and morph more smoothly. Three engineering realities come with that jump.

3.1 Addressing 64 channels

An Arduino Mega has only about fifteen hardware-PWM pins — nowhere near 64. Bit-banging that many soft-PWM channels is unreliable, so the drive path changes to dedicated PWM driver ICs. The recommended route is four **PCA9685** boards (16 channels each, 12-bit PWM, driven over a single I²C bus): beginner-friendly, cheap, and more than fast enough, since coil switching needs only hundreds of hertz. Each of the 64 channels still drives its own logic-level MOSFET and — non-negotiably, exactly as in the base guide — its own flyback diode across the coil. Omitting even one diode still eventually destroys the drive electronics.

3.2 Power and heat become the real ceiling

Sixty-four holding coils drawing 0.3–0.5 A each cannot all run at once. Budget a **12 V, 20–30 A** supply with the coil rail fused and 16 AWG or heavier distribution, and treat the thermal budget as firmware, not a suggestion: cap duty cycle around 60–70 percent, cap how many coils may exceed 50 percent simultaneously, and add active cooling — heatsinks on the coil bank and forced-air fans underneath. As the base guide states, heat is this subsystem's main reliability enemy; at 4× the coil count it is the dominant design constraint. The practical consequence for composition: you have a fixed “power budget” to spend each frame — a few tall features, or many gentle ones, never everything at full height.

3.3 The fluid, not the grid, sets the new limit

Below a certain spacing, neighboring peaks merge no matter how fine the coil pitch, because the ferrofluid's own spike spacing (the Rosensweig instability, set by fluid density and surface tension) takes over as the binding constraint. In practice an 8×8 grid yields roughly **8–12 cleanly separable features** rather than a literal 64. Shaped pole pieces on each coil sharpen the field and buy a little more separation; going denser than 8×8 mostly wastes coils. Keep everything else from the base build unchanged: the sealed saline-suspended ferrofluid cell, the horizontally-mounted TinyLev gantry 10–15 cm above the fluid, and a ≥15 cm standoff between driver boards and the strongest magnets.

4. Software Architecture — The Choreography Engine

This is where most of the new work lives. A host computer runs a real-time engine that keeps all three layers locked to one clock and one coordinate frame, and drives the two Arduinos over USB serial. The engine is organized as five cooperating modules feeding a shared timeline.

Module	Job	Output
Master clock & scene graph	Single timeline all layers subscribe to; holds the current scene, cues, and blend state.	Frame tick + scene parameters
Terrain engine	Turns a target “heightfield” into per-coil PWM with easing/cross-fade so bumps travel smoothly; enforces the thermal budget.	64-channel duty values → Mega
Bead choreographer	Turns a target path into the levitator's phase-shift commands; keeps the bead as a known fiducial.	Trajectory + phase → Nano
Photonic renderer	Projection-maps color, intensity, and labels onto terrain and bead; warped to the cell.	Projector / AR frame
Room-sound analyzer	Captures a microphone feed and extracts musical features in real time (Section 5).	Feature vector per frame

A sixth, optional **registration** service closes the loop: a depth camera above the cell measures the fluid's actual surface and corrects the photonic renderer so light lands on real, not commanded, geometry. It is not required for a first gallery piece but is the pivot toward the museum build.

The data flow is simple to state: microphone → analyzer → (blended with the scripted scene) → the three layer engines → hardware. The scene graph decides, per piece, how much control comes from the live sound versus the script.

5. Data-as-Art: Mapping Room Sound to the Stack

The chosen data source is the sound of the room itself — voices, footsteps, music, the hush of a quiet gallery. A single microphone feeds a real-time analyzer (an FFT plus a few feature extractors) that produces a compact feature vector every frame. Each feature is routed to the layer whose timescale matches it. The core insight is to **match signal speed to actuator speed**: slow envelopes drive the slow terrain, fast transients drive the fast bead.

Audio feature	Speed	Drives	Effect
Low-band energy (bass, 20–250 Hz)	Slow envelope	Central terrain coils	A broad swell rises and falls with the low end

Mid-band energy (250 Hz–4 kHz)	Medium	Edge / ring terrain coils	Ripples and secondary peaks around the rim
Onsets / transients (beats, claps)	Fast	Bead choreographer	The bead darts or jumps on each hit — the lead voice
Overall loudness (RMS)	Slow	Photonic intensity	Brighter, more saturated light as the room gets louder
Spectral centroid (“brightness”)	Medium	Photonic hue	Warm hues for dark/bassy sound, cool hues for bright sound
Silence / low activity	—	All layers	Falls back to a slow generative idle so the piece never dies

Smoothing, latency, and safety of the mapping

Because the terrain lags by up to a second, its features are driven through envelope followers (attack/release smoothing) so the fluid never chases detail it cannot render — it reads the *mood* of the sound, not its waveform. The bead, being fast, is driven by discrete onset events rather than continuous level, so it feels responsive and rhythmic. Note that the base guide's “speed of sound is a latency floor” caution concerns projecting acoustic *force* across a room; here sound is only an *input* captured at one microphone, so that limit does not apply. Keep the fluid within its stable regime — hold duty within the thermal budget and avoid commanding spikes faster than the measured tear-speed from Build Part B, so features shear smoothly instead of splattering.

6. Micro-Theater: Scripted Choreography

Micro-theater treats the stack as a tiny stage. A scripted score sequences the three layers through a short narrative arc; the same scene graph that carries live-sound parameters also carries authored keyframes, so a piece can be fully scripted, fully reactive, or — best — a scripted skeleton whose details the room's sound fills in.

The scene model

A piece is a list of **scenes**, each a set of keyframes across three tracks — a *terrain* track (target heightfields over time), a *bead* track (waypoints and dwell), and a *light* track (color, intensity, labels). Scenes blend with cross-fades; a cue system advances them on a timer, an operator key, or an audio trigger (for example, “hold until the room falls quiet, then begin”). Authoring is the creative instrument, and building this sequencer is the bulk of the software effort.

An example three-beat piece: “Landfall”

- **Beat 1 — Calm sea.** Terrain flat and glassy; the bead drifts low and slow like a distant boat light; cool blue wash. Room sound barely stirs the surface.
- **Beat 2 — The storm.** Low-band energy from the room (or a scripted swell) raises a central peak; mid-band ripples spread to the rim; the bead climbs and circles faster; light shifts to warm amber, intensity tracking loudness. This is where live sound and script interlock.

- **Beat 3 — Landfall and stillness.** The peak subsides into a low ridge that stays put; the bead settles at its crest; light cools and dims to a single held point. A slow generative idle takes over until the next cycle.

7. Parts and Cost Delta

These items are *in addition to or replacing* the base build's magnetic-layer parts; the acoustic layer, sealed fluid cell, and integration items from the core guide are unchanged. Figures are indicative.

Item	Qty	Est. cost	Notes
12 V electromagnets, 25 mm barrel	64	$140–200	Replaces the 16-coil grid; 8×8 at ~15 mm pitch.
Logic-level MOSFET modules (IRLZ44N)	64	$60–100	One channel per coil.
PCA9685 16-ch PWM driver boards	4	$12–20	I²C-chained; solves the 64-channel addressing problem.
Flyback diodes (1N5819)	70	$8	One across every coil. Still non-negotiable.
12 V 20–30 A power supply	1	$35–55	Headroom for many simultaneous coils; fuse the rail.
Heatsinks + 12 V fans (active cooling)	—	$20–35	Coil-bank cooling for sustained gallery runtime.
USB microphone / audio interface	1	$15–40	Room-sound capture for the analyzer.
Mini LED projector	1	$50–90	The photonic layer; glossy ferrofluid takes projection well.
USB depth camera (for closed loop)	1	$60–150	Optional in v1; required for the museum build.
Perfboard, screw terminals, 16 AWG wire	—	$20–30	Power distribution for 64 channels.
Delta subtotal (art rig, no depth cam)		**≈ $360–580**	**over base build**

The 8×8 magnetic-layer parts alone add roughly $180–320 versus the base 4×4; the projector, microphone, and cooling account for the rest. A depth camera adds $60–150 when you move toward the closed-loop and museum stages.

8. Roadmap: From Gallery Piece to Collaborative Exhibit

Build in stages. Each stage is a working, showable artifact, and each de-risks the next — so the museum-grade collaborative instrument is the destination, not the starting point.

Stage	What you build	What it proves / adds
V1 — Living instrument (this doc)	8×8 rig, room-sound engine, micro-theater sequencer, projector.	The two-speed instrument works; sound → matter/light is expressive and stable. A gallery-ready reactive piece.

V2 — Closed loop	Add the depth camera; register light to the fluid's measured shape.	Light lands on real geometry; the registration problem is solved on the bench. Visual polish jumps.
V3 — Hardened	Ruggedized sealed cell, full active cooling, conservative duty caps, attract/idle loop, graceful bead-loss recovery, tamper-proof enclosure, acoustic guarding.	Survives all-day unattended runtime and a public that touches everything. Turns an artwork into an exhibit.
V4 — Collaborative	Multiple public inputs (knobs/pads or several microphones), one input per terrain feature; ensemble mixing in the engine.	Two or three visitors play together — the low feature count becomes a feature: one person per channel. The queue-forming museum piece.

9. Safety Notes (Deltas)

Heat, scaled up. Sixty-four coils multiply the base build's thermal load. Mount on a non-flammable base, enforce duty and simultaneity caps in firmware, run the fans whenever coils are energized, and power down when unattended until the enclosure and interlocks of V3 exist.

Power. A 20–30 A supply can melt thin wire and start fires. Fuse the coil rail, use heavy gauge for distribution, and verify polarity before first power-up.

Ultrasound. The 40 kHz field is inaudible but intense near the arrays; keep ears and pets away during long runs. Public proximity is a V3 concern — the levitation chamber must be guarded before any exhibit deployment.

Ferrofluid and magnets. Keep the cell sealed; ferrofluid stains permanently and must never touch bare neodymium. Keep strong fields away from pacemakers, cards, and mechanical watches.

10. References

Hirayama, R., Martinez Plasencia, D., Masuda, N., & Subramanian, S. (2019). A volumetric display for visual, tactile and audio presentation using acoustic trapping. Nature, 575, 320–323.

Marzo, A., Barnes, A., & Drinkwater, B. W. (2017). TinyLev: A multi-emitter single-axis acoustic levitator. Review of Scientific Instruments, 88, 085105. Build guide: instructables.com/Acoustic-Levitator (UpnaLab).

Marzo, A., et al. Ultraino: an open phased-array platform for airborne ultrasound; successor software platform OpenMPD. (Per-transducer phase control — the route to a steerable bead and to the array emitting audible sound via amplitude modulation.)

Jordà, S., Geiger, G., Alonso, M., & Kaltenbrunner, M. (2007). The reacTable: exploring the synergy between live music performance and tabletop tangible interfaces. Proc. TEI '07. (Reference for a tangible, collaborative, real-time musical instrument.)

Kodama, S. Protrude, Flow (2001) and Morpho Tower (2006–2007). Ferrofluid sculpture — the artistic precedent for a magnetically animated matter display.

Shi, S., Hübl, M. C., Grosjean, G., Goodrich, C. P., & Waitukaitis, S. (2025). Electrostatics overcome acoustic collapse to assemble, adapt, and activate levitated matter. PNAS, 122(50). (Path to higher bead counts.)

StableLev: Data-Driven Stability Enhancement for Multi-Particle Acoustic Levitation. CHI 2024.

Matter Stack — Choreography Engine

Software Specification: data formats, scene-file schema, and Arduino serial protocol

Companion to The Matter Stack — Living Instrument (Art & Expressive-Medium Addendum) · Engineering spec v0.1 — July 2026

1. Overview and Design Principles

The choreography engine is the real-time host program that turns a choreography (scripted keyframes plus live room-sound) into synchronized setpoints for the two microcontrollers driving the physical layers. It runs on a host computer — reference implementation in Python — and speaks to an **Arduino Mega** (the 8×8 terrain grid via four PCA9685 boards) and an **Arduino Nano** (the TinyLev levitator) over two USB-serial links.

Four principles govern the whole design:

- **Host computes, controllers actuate.** All choreography, audio analysis, mapping, and safety arithmetic happen on the host. Firmware is deliberately dumb: parse a frame, validate it, apply setpoints, run PWM, and fail safe. No creative logic lives on an Arduino.
- **One clock, one frame.** A single monotonic engine clock timestamps everything; the scene player, the modulation layer, and both serial writers all sample the same time `t`. A single normalized coordinate frame (Section 3) ties matter, sound, and light together.
- **Match signal speed to actuator speed.** The terrain is slow (0.2–1 s), the bead is fast (tens of ms). The engine drives each from the part of the signal it can actually follow — slow envelopes to terrain, discrete onsets to the bead.
- **Fail safe by default.** Every link carries a heartbeat; loss of it forces the controller to a safe state (coils off, levitator to a defined idle). Thermal and current limits are enforced on the host and re-checked in firmware as a last resort.

2. Architecture and Data Flow

The engine is a set of modules feeding a fixed-rate control loop. The loop samples inputs, evaluates the active scene blended with live modulation into a single `FrameState`, then hands that state to per-layer compute stages and the serial writers.

Pipeline: `microphone → sound analyzer → (scene player ⊕ modulation) → FrameState → {terrain engine → Mega} · {bead choreographer → Nano} · {photonic renderer → projector}`. An optional depth-camera registration service corrects the photonic stage in v2.

Module rates and roles:

Module	Rate	Role
Engine control loop	`60 Hz`	Fixed-timestep scheduler; produces one FrameState per tick.
Sound analyzer	`~86 Hz`	1024-sample FFT, 512 hop @ 44.1 kHz; emits the feature vector (§4.2).

Scene player	`60 Hz`	Evaluates keyframe tracks of the active scene at time t; handles cues/transitions.
Modulation layer	`60 Hz`	Adds smoothed audio-feature contributions onto scripted base values (§6).
Terrain engine	`30 Hz`	Feature list → heightfield → per-coil duty; thermal budget + slew limit (§7).
Bead choreographer	`50 Hz`	Bead target → levitator phase/drive; onset-driven darts (§8).
Photonic renderer	`30–60 Hz`	Projection-maps color/intensity to cell coords; uses commanded bead pos as fiducial.
Serial writers (×2)	`30 / 50 Hz`	Encode and transmit frames; read ACK/STATUS; run link watchdogs (§9).
Safety supervisor	`60 Hz`	Global thermal/current budget, heartbeats, ESTOP, clean shutdown (§11).

3. Coordinate Frames and Units

All spatial authoring is in a normalized **cell frame**: `x, y ∈ [0,1]` across the fluid cell, origin at the front-left corner, x to the right, y away from the viewer. The engine maps normalized coordinates to physical coil positions using the geometry in the config file, so scenes are resolution-independent and survive a 4×4 → 8×8 change unmodified.

Quantity	Symbol / field	Type & range	Notes
Cell position	`x, y`	`float [0,1]`	Normalized over the cell; origin front-left.
Terrain height	`h(x,y)`	`float [0,1]`	0 = flat, 1 = max stable displacement. Not duty; mapped per-coil in §7.
Coil duty	`duty`	`uint8 [0,255]`	Wire value to Mega; firmware scales to 12-bit PWM.
Bead height	`bead.z`	`float [0,1]`	Position along the levitator's vertical node axis (TinyLev v1).
Bead x,y (v2+)	`bead.x, bead.y`	`float [0,1]`	Reserved for a phased MATD array; ignored by TinyLev firmware.
Color	`rgb`	`uint8[3] 0–255`	sRGB. Authoring may use HSV; engine converts.
Intensity	`level`	`float [0,1]`	Photonic brightness multiplier.
Time	`t`	`float seconds`	Monotonic from engine start; double precision.

4. Runtime Data Formats

4.1 FrameState — the per-tick payload

Every control tick produces one immutable FrameState. It is the single source of truth handed to the compute stages. Terrain is authored as a compact *feature list* (a few parametric bumps) rather than a dense grid — this matches the physics (§3.3 of the addendum: only ~8–12 separable features) and keeps scenes small. The terrain engine rasterizes it to the 8×8 grid at runtime.

```
FrameState = {
  "t": 12.480,                      # engine time, seconds
  "terrain": {
    "features": [                   # 0..~12 parametric bumps
      {"x":0.50,"y":0.50,"amp":0.82,"sigma":0.14},
      {"x":0.20,"y":0.75,"amp":0.35,"sigma":0.10}
    ],
    "floor": 0.00                   # baseline height added everywhere
  },
  "bead":  {"z":0.66,"drive":0.90,"slew":0.40},   # slew: 0 slow .. 1 instant
  "light": {"mode":"wash","rgb":[220,120,40],"level":0.7,
            "targets":"all"},       # or per-feature color list
  "flags": {"estop":false}
}
```

4.2 Audio feature vector

The analyzer emits one feature vector per audio hop, each field normalized to [0,1] by a running auto-gain (slow AGC) so a quiet gallery and a loud crowd both use the full range. Definitions:

Field	Definition	Speed / use
`rms`	Windowed RMS loudness, AGC-normalized.	Slow → light intensity
`band_low`	Energy 20–250 Hz (bass).	Slow → central terrain feature
`band_mid`	Energy 250 Hz–4 kHz.	Medium → rim terrain features
`band_high`	Energy 4–16 kHz.	Medium → sparkle / light hue
`centroid`	Spectral centroid, mapped log to [0,1] ("brightness").	Medium → light hue
`flux`	Spectral flux (positive change between frames).	Fast → texture / agitation
`onset`	Boolean + strength[0,1] from a peak-picked flux detector.	Fast → bead dart

5. Scene-File Schema

A choreography is authored as a JSON **piece**: metadata, a settings block, an optional global `modulation` section (live-sound bindings), and an ordered list of **scenes**. Each scene has a duration, a transition, three keyframed tracks (`terrain`, `bead`, `light`), and a `cue` describing how it advances. Keyframes carry a time (seconds, relative to scene start), a value, and an easing.

5.1 Field reference

Path	Type	Meaning
`piece.name / author / bpm`	`str / str / num`	Metadata. bpm optional, enables beat-based cue timing.
`settings.loop`	`bool`	Restart the piece when the last scene ends.
`settings.idle`	`obj`	Generative fallback used on silence or between pieces.
`modulation[]`	`array`	Global live-sound bindings (§6). Applied on top of every scene.

`scenes[].id`	`str`	Unique scene name; used by cue targets.
`scenes[].duration`	`num (s)`	Nominal length; a cue may extend or cut it.
`scenes[].transition`	`obj`	{"type":"crossfade","dur":1.5} blend from previous scene.
`...terrain[]`	`keyframe[]`	Each value is a terrain feature-list snapshot (see §4.1).
`...bead[]`	`keyframe[]`	Each value = {z, drive, slew}.
`...light[]`	`keyframe[]`	Each value = {rgb|hsv, level, mode, targets}.
`keyframe.t / .v / .ease`	`num / obj / str`	Time in scene, value, easing: linear|ease|hold|cubic.
`scenes[].cue`	`obj`	Advance rule: by timer, operator key, or audio trigger (§5.3).

5.2 Worked example — the “Landfall” piece

A complete, valid three-scene piece. Terrain features are abbreviated for space; a real file lists each keyframe's full feature array. The listing continues across the two blocks below.

```
{
  "piece": {"name":"Landfall","author":"CB","bpm":72},
  "settings": {
    "loop": true,
    "idle": {"type":"perlin","amp":0.15,"speed":0.05}
  },
  "modulation": [
    {"src":"band_low","dst":"terrain.features[0].amp",
     "gain":0.6,"offset":0.0,"attack":0.25,"release":0.8,"clamp":[0,1]},
    {"src":"rms","dst":"light.level","gain":0.7,"offset":0.2,
     "attack":0.15,"release":0.6,"clamp":[0,1]},
    {"src":"onset","dst":"bead.impulse","gain":1.0,"mode":"event"}
  ],
```

```
  "scenes": [
    {
      "id":"calm_sea","duration":18.0,
      "transition":{"type":"crossfade","dur":2.0},
      "terrain":[
        {"t":0.0,"v":{"features":[{"x":0.5,"y":0.5,"amp":0.05,"sigma":0.30}],
                      "floor":0.0},"ease":"linear"}],
      "bead":[{"t":0.0,"v":{"z":0.25,"drive":0.8,"slew":0.2},"ease":"ease"}],
      "light":[{"t":0.0,"v":{"hsv":[210,0.5,0.6],"level":0.4,"mode":"wash",
                             "targets":"all"},"ease":"linear"}],
      "cue":{"type":"timer"}
    },
    {
      "id":"storm","duration":20.0,
      "transition":{"type":"crossfade","dur":1.0},
      "terrain":[
        {"t":0.0,"v":{"features":[{"x":0.5,"y":0.5,"amp":0.30,"sigma":0.16}],
                      "floor":0.05},"ease":"cubic"},
        {"t":8.0,"v":{"features":[{"x":0.5,"y":0.5,"amp":0.90,"sigma":0.12},
                                  {"x":0.25,"y":0.7,"amp":0.4,"sigma":0.10},
                                  {"x":0.78,"y":0.3,"amp":0.4,"sigma":0.10}],
                      "floor":0.1},"ease":"cubic"}],
      "bead":[{"t":0.0,"v":{"z":0.4,"drive":0.9,"slew":0.5},"ease":"ease"},
              {"t":8.0,"v":{"z":0.85,"drive":1.0,"slew":0.7},"ease":"ease"}],
      "light":[{"t":0.0,"v":{"hsv":[210,0.5,0.6],"level":0.5},"ease":"linear"},
               {"t":10.0,"v":{"hsv":[30,0.9,1.0],"level":0.9},"ease":"linear"}],
      "cue":{"type":"audio","when":"rms<0.15","hold":true,"timeout":30.0}
    },
```

```
    {
      "id":"stillness","duration":14.0,
      "transition":{"type":"crossfade","dur":3.0},
      "terrain":[
        {"t":0.0,"v":{"features":[{"x":0.5,"y":0.45,"amp":0.30,"sigma":0.22}],
                      "floor":0.05},"ease":"cubic"}],
      "bead":[{"t":0.0,"v":{"z":0.35,"drive":0.85,"slew":0.15},"ease":"ease"}],
      "light":[{"t":0.0,"v":{"hsv":[205,0.4,0.5],"level":0.3,"mode":"point"},
                "ease":"linear"}],
      "cue":{"type":"timer"}
    }
  ]
}
```

5.3 Cue types

cue.type	Advances when	Fields
`timer`	Scene duration elapses.	–
`operator`	A key/OSC message is received (live performance).	`key`
`audio`	An expression over the feature vector becomes true.	`when, hold, timeout`

6. Reactive Modulation Model

Live sound never replaces the scripted choreography — it *modulates* it. For each bound parameter the engine computes a final value as the scripted base (from the keyframe tracks) plus the sum of its bindings' contributions:

```
value(param, t) = base(param, t) + Σ  clamp( gain_i * curve_i(feat_i) + offset_i )
                                   i
smoothed feature (envelope follower), evaluated each tick dt:
    tau   = (x > y) ? attack : release          # asymmetric
    alpha = 1 - exp(-dt / tau)
    y    += (x - y) * alpha                      # y is the smoothed feature
```

Continuous bindings (e.g. `band_low → terrain.features[0].amp`) use the envelope follower so slow actuators track the *mood* of the sound, not its waveform; long release keeps the terrain from twitching. **Event** bindings (`mode:"event"`, e.g. `onset → bead.impulse`) fire on a discrete trigger and inject a fast, decaying impulse — this is what makes the bead feel rhythmic. Bindings are additive and order-independent; a per-parameter clamp keeps the total in range and protects the fluid's stable regime.

7. Terrain Engine — Features to Coil Duty

The terrain engine converts the feature list into 64 duty bytes in four stages, every 30 Hz frame:

1. Rasterize. Sample the height field on the 8×8 coil lattice as a sum of Gaussian bumps: $h(cx,cy) = floor + \Sigma\ amp_k \cdot \exp(-((cx-x_k)^2+(cy-y_k)^2) / (2 \cdot sigma_k^2))$, clamped to [0,1].

2. Map to duty. Apply each coil's calibrated height→duty curve (measured once in Build Part B and stored in config as a small lookup per coil). The relationship is non-linear; a per-coil curve corrects for coil-to-coil variation.

3. Thermal budget. If `Σ duty` exceeds the global power cap, scale all duties by a single factor so the tallest features survive and the whole field dims together rather than clipping arbitrarily. A per-coil dwell integrator (an I^2t-style accumulator) forces any coil that has run hot to cool, overriding the setpoint if needed.

4. Slew limit. Clamp per-coil change per frame to a max Δduty so bumps travel smoothly and the fluid shears instead of splattering (the tear-speed measured in Build Part B sets the ceiling).

Output is a 64-byte array in row-major coil order, sent as a `TERRAIN_FRAME` (§9.2).

8. Bead Choreographer — Target to Levitator

For the TinyLev v1 the bead has one controllable axis: the standing-wave node it sits in, moved by the driver's phase-shift feature. The choreographer maps `bead.z ∈ [0,1]` to a phase setpoint `0..PHASE_MAX`, applies the requested slew (rate limit), and layers onset impulses on top: an `onset → bead.impulse` event adds a brief, fast offset that decays back to the scripted z, producing the darting “lead voice.” Drive level maps to the levitator amplitude (kept within the safe range from Build Part A). Bead x,y fields are carried through the pipeline but ignored by TinyLev firmware; they become active with the phased-array (MATD) upgrade, at which point only the choreographer's output stage and firmware change.

9. Serial Protocol

Two independent USB-serial links, each a framed binary protocol. Defaults: **Mega 250000 baud**, **Nano 115200 baud**, 8N1. Framing is **COBS** (Consistent Overhead Byte Stuffing) with `0x00` as the packet delimiter, so a payload can contain any byte value without collision.

9.1 Frame layout

Before COBS encoding, every packet is:

```
[ TYPE (1) | SEQ (1) | LEN (1) | PAYLOAD (LEN) | CRC16 (2, little-endian) ]
  CRC16 = CRC-16/CCITT-FALSE  (poly 0x1021, init 0xFFFF) over TYPE..PAYLOAD
  whole packet is then COBS-encoded and terminated with a 0x00 byte
  SEQ increments per packet; the receiver echoes it in ACK/STATUS
```

9.2 Host → Mega (terrain)

Type	Name	Payload	Notes
`0x01`	`TERRAIN_FRAME`	`64 × uint8`	Row-major coil duty. Sent at 30 Hz.
`0x02`	`SET_CONFIG`	`pwm_hz u16, duty_cap u8, max_hot u8`	PWM freq, global duty cap, max simultaneous hot coils.
`0x03`	`ALL_OFF`	—	Immediate: all coils to 0. ESTOP path.
`0x04`	`PING`	`t_ms u32`	Heartbeat; expected every ≤100 ms.
`0x05`	`HELLO`	`proto_ver u8`	Startup handshake; expects HELLO_ACK.

9.3 Mega → Host

Type	Name	Payload	Notes
`0x81`	`STATUS`	`seq u8, flags u8, i_est u16(mA), t_c u8`	Echoes last SEQ; flags = ok/over-temp/over-current/underrun.
`0x82`	`ERROR`	`code u8`	CRC fail, bad LEN, budget trip, etc.
`0x8F`	`HELLO_ACK`	`proto_ver u8, n_coils u8`	Capabilities reported at startup.

9.4 Host → Nano (levitator)

Type	Name	Payload	Notes
`0x10`	`BEAD_TARGET`	`phase u16, drive u8, slew u8`	Node position, amplitude, rate limit. Sent at 50 Hz.
`0x11`	`LEVITATOR_EN`	`on u8`	Enable/disable drive. Disable = safe idle.
`0x12`	`SET_CONFIG`	`drive_max u8`	Cap amplitude to the safe range.
`0x04`	`PING`	`t_ms u32`	Heartbeat; expected every ≤100 ms.

9.5 Watchdog and example

Each firmware runs a receive watchdog: if no valid packet (any type) arrives within **150 ms**, it forces the safe state — Mega drops all coils to 0, Nano disables drive. The host therefore sends PING as a keepalive whenever it has no data frame to send. On host exit, the engine sends `ALL_OFF` and `LEVITATOR_EN 0` before closing the ports.

Example — a TERRAIN_FRAME with SEQ 42 and all-zero duty (pre-COBS):

```
TYPE=01 SEQ=2A LEN=40 PAYLOAD=00*64 CRC=<lo><hi>
 -> CRC-16/CCITT-FALSE over [01 2A 40 00...00]  (67 bytes)
 -> COBS-encode the whole packet, append 0x00 terminator, write to port
```

10. Timing, Threading, and Synchronization

The control loop uses a fixed-timestep accumulator (60 Hz) so behavior is deterministic regardless of host jitter. Audio capture runs in its own callback thread and publishes the latest feature vector through a lock-free single-slot buffer. Each serial link has its own writer thread consuming the newest Frame-derived packet (drop-oldest, never queue-and-lag) and its own reader thread parsing STATUS. Because both writers sample the same FrameState, the layers stay phase-locked to the shared clock.

Stage	Typical latency	Consequence
Audio in → feature vector	`~12–15 ms`	Onset detection is tight enough to feel rhythmic.
Engine tick	`16.7 ms`	One-frame quantization of all outputs.
Serial + bead actuation	`~20–50 ms`	Bead reads as responsive (the “lead voice”).
Serial + fluid response	`~0.2–1 s`	Terrain reads the mood, by design (the “pad”).

The bead doubles as the optical fiducial: because its commanded trajectory is known, the photonic renderer aligns to it with zero sensing latency. The terrain's true shape is emergent and, in v2, is measured by the depth camera to register light to real geometry.

11. Software Safety Supervisor

A dedicated supervisor runs at loop rate and owns the machine's safe state:

- **Global budget.** Maintains the running current/thermal estimate and enforces the duty cap and per-coil dwell integrator before frames leave the host (firmware re-checks as backstop).
- **Heartbeats.** Guarantees each link sees traffic within its watchdog window; escalates to ESTOP on repeated STATUS errors or link loss.
- **ESTOP path.** A single call drives `flags.estop`, which forces terrain to zero and the levitator to idle within one tick and latches until cleared.
- **Clean shutdown.** On exit or exception, sends ALL_OFF + LEVITATOR_EN 0, flushes, and closes ports — never leaves coils energized.

12. Config, Firmware Responsibilities, and Implementation Notes

12.1 Engine config (excerpt)

```
ports:   {mega: "/dev/ttyACM0", mega_baud: 250000,
          nano: "/dev/ttyUSB0", nano_baud: 115200}
grid:    {cols: 8, rows: 8, pitch_mm: 15, origin: "front_left"}
coil_cal:  "coil_curves.csv"       # per-coil height->duty lookup
thermal: {duty_cap: 190, max_hot: 20, dwell_ms: 4000}
rates:   {loop_hz: 60, terrain_hz: 30, bead_hz: 50}
audio:   {device: "default", sr: 44100, fft: 1024, hop: 512,
          bands: [250, 4000, 16000]}
projector: {homography: "cell_homography.json"}
```

12.2 Firmware does only this

Both Arduinos: read bytes, COBS-decode, verify CRC and LEN, apply setpoints (Mega → PCA9685 duty; Nano → phase/drive), service the receive watchdog, enforce a local last-resort duty/amplitude cap, and emit STATUS. No easing, no mapping, no scene logic — all of that is the host's job, which keeps firmware small, testable, and safe.

12.3 Reference stack and layout

Python reference: `pyserial` (links), `sounddevice + numpy` (capture/FFT), `cobs` (framing), `opencv` (projector homography, depth cam in v2). Suggested modules: `clock.py`, `scene.py`, `modulation.py`, `audio.py`, `terrain.py`, `bead.py`, `photonic.py`, `transport.py`, `safety.py`, `engine.py`, plus `pieces/*.json` and `config.yaml`.

12.4 v2 / V4 hooks

The schema and pipeline already carry what later stages need: `bead.x/y` for the MATD phased array; a registration service slot for the depth-camera closed loop; and, for the collaborative museum build (V4), the modulation layer generalizes from one microphone to multiple input sources by adding bindings whose `src` is a per-visitor control channel — one participant per terrain feature.

PART V V3 — Museum Hardening

V3 is the stage that turns a working art rig into something the public can be left alone with. It sits between the closed-loop V2 (depth-camera registration) and the collaborative V4 (knob/slider control): no new display capability, but the enclosure, thermal, electrical, and software robustness that let the machine run all day, unattended, in front of strangers who will touch everything. Do this before any public deployment — and before V4, which builds directly on it.

> ⚠ Hardening is a safety stage, not a polish stage. Until the enclosure, guarding, thermal cutoff, and watchdogs below are in place, the machine should only run attended.

Figure 15 — V3 museum hardening (enclosure cutaway)

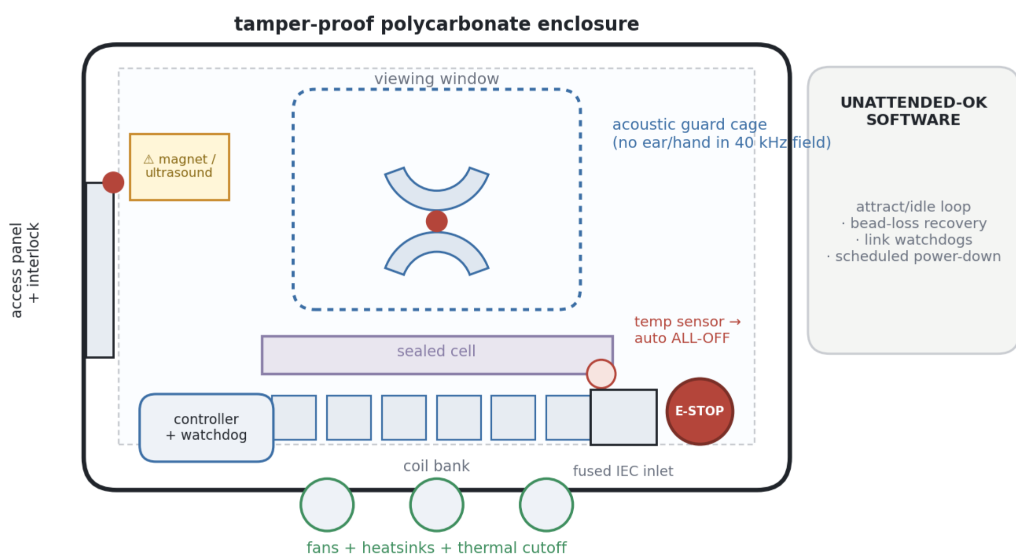


Figure 15 — What V3 adds around the working stack: enclosure, acoustic guard, cooling with a thermal cutoff, an interlocked access panel, an emergency stop, and unattended-safe software.

V3.1 · Enclosure and tamper-proofing

1. **Enclose everything.** Build a rigid polycarbonate or acrylic enclosure around the whole stack with a clear viewing window. Visitors see the terrain, bead, and light but cannot reach the ferrofluid cell, coils, levitator, or wiring.
2. **Use tamper-proof fasteners.** Close the enclosure with security screws (pin-hex or similar) so panels need a special driver to open. Nothing hand-removable.
3. **Interlock the access panel.** Fit the service door with an interlock switch that cuts coil power and disables the levitator whenever it opens, so the machine can never run with the enclosure open.

V3.2 · Guard the acoustic layer

4. **Cage the levitator.** Surround the levitation zone with a physical guard that keeps ears and hands out of the intense near-field of the 40 kHz array while preserving the required standoff and the sightline to the bead.
5. **Label the field.** Apply clear signage for the two invisible hazards — strong magnetic field (pacemakers, cards, watches) and ultrasound (keep close exposure brief).

V3.3 · Thermal system and interlocks

6. **Cool the coil bank properly.** Move from the bench fans to sustained cooling: heatsinks on the coil bank plus filtered forced-air fans sized for continuous all-day duty inside a closed enclosure.
7. **Add a hardware thermal cutoff.** Mount a temperature sensor or thermostat on the coil bank that drops coil power (independently of the firmware) if it exceeds a safe threshold. This is the backstop if a software limit ever fails.
8. **Tighten the firmware budget.** Set conservative duty caps, a simultaneity cap, and per-coil dwell limits for unattended running, and have an over-temp STATUS flag trigger an automatic ALL-OFF.

V3.4 · Electrical safety

9. **Contain and fuse the power.** Feed the enclosure from a fused, switched IEC inlet; fuse the coil rail; bond exposed conductive parts to protective earth; keep all mains wiring inside the enclosure with no user access.
10. **Fit an emergency stop.** Add a clearly marked E-STOP that removes coil power and disables the levitator in one action, latching until reset by an operator.
11. **Strain-relieve everything.** Anchor and strain-relieve every cable; a machine that runs for months cannot rely on friction-fit connectors.

V3.5 · Unattended-safe software

12. **Never let the show die.** Run an attract/idle loop so the display is always doing something inviting when no one is interacting.
13. **Recover from bead loss.** Detect when the levitated bead is lost and either re-trap automatically or fall back gracefully to a terrain-and-light-only mode until it can be re-seeded — without operator intervention.
14. **Watchdog both links.** Keep the heartbeat watchdogs on the Mega and Nano (150 ms) so any lost link forces the safe state, and add a scheduled daily power-down/restart to clear faults.

V3.6 · Accessibility, signage, and sign-off

Mount the viewing window and any controls at an accessible height, use the photonic layer for captions so the piece explains itself, and post the magnet/ultrasound warnings and an operator quick-reset instruction. Before opening to the public, run an unattended soak test (several hours minimum) and confirm each safe-state path — open the access panel, trip the thermal cutoff, hit the E-STOP, and pull each USB link — verifying the machine reaches a safe state every time.

With V3 complete the installation is exhibit-grade: sealed, guarded, thermally protected, electrically safe, and able to run and recover on its own — the foundation the collaborative V4 controls plug straight into.

PART VI V4 — The Collaborative Exhibit (knob / slider control)

V4 is the destination of the roadmap: the hardened machine becomes an instrument several visitors play together. The move is simple and powerful — replace (or augment) the single room microphone with a small set of tangible control stations, and give **each participant one terrain feature**. Because the 8×8 grid only supports about 8–12 separable features anyway, the low count stops being a limitation and becomes the social design: one person, one bump. Two to four people sculpt a shared landscape at once, each owning their own peak, the projector coloring each person's feature in their own hue.

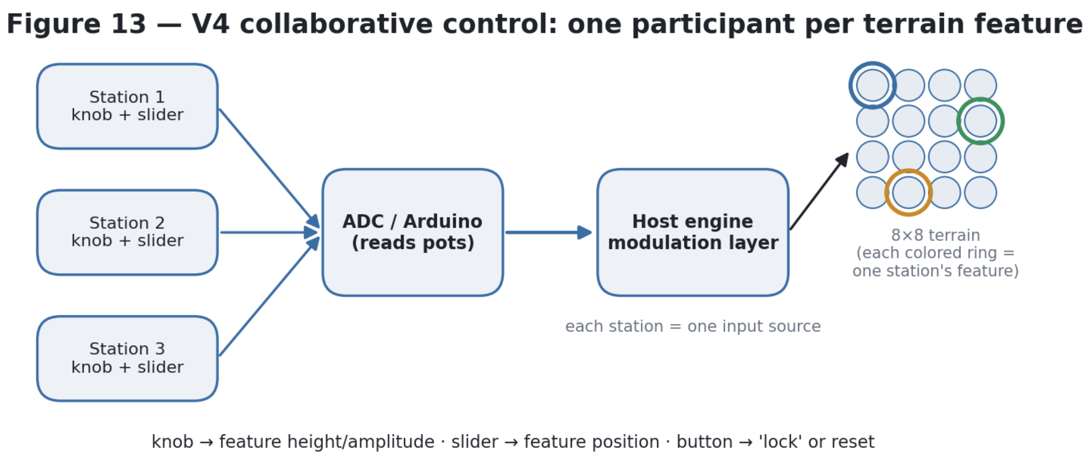
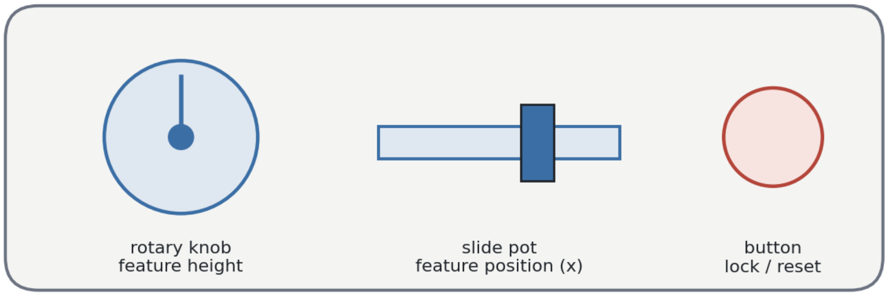


Figure 13 — Control stations feed the engine's modulation layer as new input sources; each maps to one feature on the terrain.

V4.1 · Hardware — the control stations

Each station is a small panel with a rotary knob, a slide potentiometer, and a button (Figure 14). The knob and slider are ordinary potentiometers read as analog voltages; the button is a debounced digital input. A single ADS1115 16-bit ADC (or the spare analog pins of a dedicated Arduino) reads all stations and streams their values to the host over USB — exactly like the two existing controllers.

Figure 14 — A single control station (accessible panel)



industrial-grade pots, debounced button, panel at wheelchair height · 2–4 stations around the cell

Figure 14 — One station. Use industrial-grade pots and mount the panel at an accessible height for museum use.

Control	Reads as	Default mapping
Rotary knob	analog 0–1	Height / amplitude of the participant's feature
Slide potentiometer	analog 0–1	Position of the feature along one axis (x)
Button	digital, debounced	Lock the current shape, or reset the feature to flat

V4.2 · Software — stations as modulation sources

No new engine architecture is needed. The choreography engine already treats live inputs as *modulation sources* bound to parameters; V4 simply adds a control-surface source per station. Each station's knob and slider become bindings whose `src` is that participant's channel, targeting that participant's feature — so the same additive, smoothed modulation model (with per-pot calibration and dead-zones) drives the terrain.

```
"modulation": [
  {"src":"ctrl.station1.knob",   "dst":"terrain.features[0].amp",
   "gain":1.0, "attack":0.15, "release":0.4, "clamp":[0,1]},
  {"src":"ctrl.station1.slider", "dst":"terrain.features[0].x",
   "gain":1.0, "attack":0.10, "release":0.3, "clamp":[0.05,0.95]},
  {"src":"ctrl.station1.button", "dst":"terrain.features[0].lock", "mode":"event"}
  /* ... repeat block per station, each to its own feature index ... */
]
```

Three behaviours are worth building beyond the raw mapping: **calibration** (map each pot's real travel to 0–1 with a small dead-zone so it reaches both ends cleanly); **collision handling** (when two participants push features together, let them merge gracefully or apply a gentle repulsion so neither is erased — the fluid's spike-spacing limit still applies); and **fair idle** (a station with no recent input drifts back toward a calm baseline, and with nobody present the whole field falls to the generative idle from the sound-reactive mode).

A shared microphone can stay in the mix as a global 'conductor' — the room's overall loudness lightly modulating every feature and the photonic intensity — so the ensemble still responds to the collective energy on top of individual control.

V4.3 · Bring-up sequence

1. **Wire one station.** Connect one knob and one slider to the ADS1115 (or Arduino analog pins) and the button to a digital pin with a pull-up; stream the three values to the host.
2. **Calibrate.** Sweep each pot end to end, record min/max, and map to 0–1 with a small dead-zone at both extremes.
3. **Map to one feature.** Add the three bindings above for feature 0 and confirm the knob raises a bump, the slider moves it, and the button locks/resets it.
4. **Add stations.** Duplicate the wiring and the binding block for each additional station, each targeting its own feature index and its own projected color.
5. **Tune the ensemble.** Set collision handling and idle behaviour, add the optional conductor microphone, and rehearse with two or three people to balance responsiveness against calm.

V4.4 · Museum-hardening ties

V4 inherits the V3 hardening and adds control-surface robustness: industrial-grade potentiometers rated for tens of thousands of cycles, a debounced and sealed button, panels mounted at wheelchair-accessible height, and strain-relieved wiring. Combined with the sealed cell, guarded levitator, active cooling, attract/idle loop, and graceful recovery from V3, the result is the queue-forming collaborative exhibit the roadmap set out to reach — a persistent matter layer, a steerable acoustic layer, and a photonic overlay, played together by the public, one hand per feature.

PART VII Parts, Pricing & Final Build Price

This section lists a complete bill of materials for every stage, with a representative price and a place to order each item. Build cost accrues in five stages — the 4×4 base, the 8×8 upgrade, the MATD phased-array upgrade, V3 museum hardening, and the V4 collaborative controls — and the final summary gives the running total.

> **Pricing basis.** Prices are representative in USD as of July 2026 for hobby-grade sourcing, and will drift — confirm each at checkout. Where a branded part costs much more than a clone (Arduino, PCA9685), both are noted; the totals use the economical option. "Amazon" denotes typical marketplace listings for a commodity part rather than one specific seller.

Stage 1 — the 4×4 base build

Item	Qty	Price	Where to order
Acoustic Levitator kit — 76 transducers, Nano, L298N, switch, 9 V PSU, TinyLev frame	1	$70	Makerfabs
Expanded polystyrene beads, 1–3 mm (bag)	1	$6	Amazon
Teabag paper / fine mesh + toothpicks (bead scoop)	1	$2	Amazon
12 V electromagnets, 25 mm (5 kg holding)	16	$64	Amazon (uxcell)
Logic-level MOSFET modules (IRLZ44N)	16	$18	Amazon
Flyback diodes 1N5819 (100-pack)	1	$7	Amazon
Arduino Mega 2560 (clone; official ≈ $50)	1	$18	Arduino / Amazon
12 V 10 A power supply	1	$16	Amazon
Ferrofluid, Ferrotec EFH-1, 50 ml	1	$30	Ferrotec / Amazing Magnets
Flat glass container + lid / large petri dish	1	$10	Amazon
Distilled water + table salt	1	$3	Grocery
Perfboard, screw terminals, 18 AWG wire	—	$14	Amazon
Nitrile gloves, plastic sheeting, paper towels	—	$8	Amazon
3D-printed gantry / frame	1	$6	Self-print / Makerfabs
Mini LED projector (photonic layer)	1	$60	Amazon
Stage 1 subtotal		**$332**	

Optional: USB depth camera for closed-loop registration, +$90 (Amazon). Tools if not owned — soldering iron + multimeter, +$38.

Stage 2 — upgrade to 8×8 (delta over Stage 1)

Item	Qty	Price	Where to order
12 V electromagnets, 25 mm (fresh 64; reuse the 16 to save ≈ $46)	64	$200	Amazon (uxcell)
Logic-level MOSFET modules (IRLZ44N)	64	$60	Amazon

PCA9685 16-ch PWM boards (clone; Adafruit ≈ $60 for 4)	4	$20	Adafruit / Amazon
Flyback diodes 1N5819 (extra 100-pack)	1	$7	Amazon
12 V 30 A power supply	1	$32	Amazon
Heatsinks + 12 V fans (active cooling)	—	$28	Amazon
USB microphone (sound-reactive input)	1	$25	Amazon
Perfboard, terminals, 16 AWG wire (power distribution)	—	$20	Amazon
Stage 2 subtotal		**$392**	

Stage 3 — MATD phased-array upgrade (delta)

Item	Qty	Price	Where to order
40 kHz transducers, bulk (some reusable from the kit)	256	$120	Amazon
Phased-array driver boards + FPGA/MCU controller (Ultraino / OpenMPD class)	1 set	$250	Ultraino (open HW)
Custom array PCB / plate fabrication	1	$40	Makerfabs PCB
RGB LED + driver (bead coloring)	1	$15	Adafruit
Stage 3 subtotal		**$425**	

MATD parts vary widely with the array size and whether you fabricate boards or buy them; budget $300–600. The figure above is a mid-range 256-transducer build.

Stage 4 — V3 museum hardening (delta)

Item	Qty	Price	Where to order
Polycarbonate / acrylic enclosure panels + security fasteners	1 set	$60	Amazon
Acoustic guard / cage for the levitator	1	$15	Amazon
Access-panel interlock switch(es)	1–2	$8	Amazon
Thermostat / thermal cutoff + temperature sensor	1	$12	Amazon
Emergency-stop button + fused, switched IEC inlet	1	$18	Amazon
Cable management, strain relief, tamper-proof fasteners	—	$15	Amazon
Extra filtered fan + printed warning signage	—	$16	Amazon
Stage 4 subtotal		**$144**	

Applied before public deployment; see Part V for the full hardening procedure. Costs assume a modest desktop-scale enclosure — a larger free-standing cabinet costs more.

Stage 5 — V4 collaborative controls (delta)

Item	Qty	Price	Where to order
Rotary potentiometers + knob caps	4	$12	SparkFun / Amazon

Slide potentiometers	4	$14	Adafruit / SparkFun
ADS1115 16-bit ADC breakout (or spare Arduino analog pins)	1	$12	Adafruit
Control-surface Arduino (Nano / Uno)	1	$12	Amazon
Panel / enclosure + wiring (2–4 stations)	—	$20	Amazon
Stage 5 subtotal		**$70**	

Final build price

Building the stages in sequence — the recommended path, since each is a working artifact — the running total is:

Stage	Adds	Stage cost	Cumulative
1 · 4×4 base build	TinyLev + ferrofluid terrain + projector	$332	$332
2 · 8×8 upgrade	64-coil grid, PCA9685, power & cooling, mic	$392	$724
3 · MATD upgrade	steerable phased array, RGB bead	$425	$1,149
4 · V3 museum hardening	enclosure, guard, thermal cutoff, E-stop, interlock	$144	$1,293
5 · V4 controls	3–4 knob/slider stations	$70	$1,363

Complete build ≈ $1,365 — the full hardened V4 installation: 8×8 sound-reactive terrain, MATD phased-array levitation with haptics and sound, photonic overlay, museum-grade hardening (enclosure, guarding, thermal cutoff, E-stop), and a collaborative knob/slider panel. Excludes the optional depth camera (+$90) and tools (+$38).

Ways the number moves: reusing the sixteen 4×4 coils and their MOSFETs inside the 64-coil grid saves ≈ $60; genuine Adafruit PCA9685 boards add ≈ $40; a larger or bought-not-built MATD array can add several hundred dollars; a free-standing museum cabinet costs more than the desktop enclosure priced here. A first, gallery-ready machine that stops after the 8×8 stage (sound-reactive, no MATD or hardening) lands around **$724**.