ABSTRACT

Title of Dissertation: MEDIATING COMPLEXITY: REASONING LAYERS FROM EMBODIED INTERACTION TO AGENTIC SYSTEMS

Yuran Ding
Doctor of Philosophy, 2025

Dissertation Directed by: Professor Ashok Agrawala
Department of Computer Science
University of Maryland

Dr. Paul Strohmeier
Max Planck Institute for Informatics

This dissertation explores how intelligent systems can act as reasoning layers that mediate between human intent and low-level system signals in complex workflows. Across projects spanning embodied interaction, multimodal presentation systems, and machine learning system optimization, I derive empirical findings about how systems can encode interaction through form, couple feedback to user action at the pre-reflective level, fuse heterogeneous signals for intent inference, and expose autonomous reasoning to support human decision-making.

These findings are synthesized into DEXAS, a framework for human–agent collaborative reasoning, and instantiated in PROMPTS, an agentic system that turns system traces into diagnostic insights and prunes large configuration search spaces through explainable proposals for optimizing large-scale LLM workloads. Together, this work establishes reasoning layers as a new systems abstraction that reduces the translation burden on experts and reconfigures their role toward high-level judgment.

MEDIATING COMPLEXITY: REASONING LAYERS FROM EMBODIED INTERACTION TO AGENTIC SYSTEMS

by

Yuran Ding

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2025

Advisory Committee:
Professor Ashok Agrawala, Chair/Advisor
Dr. Paul Strohmeier, Co-Advisor
Professor Dinesh Manocha
Professor Huaishu Peng
Professor Vanessa Frias-Martinez
Professor Ming Lin

© Copyright by
Yuran Ding
2026

Acknowledgments

I owe my deepest gratitude to my advisors, Professor Ashok Agrawala and Dr. Paul Strohmeier, whose guidance carried me through not only the many challenges of research, but also moments of uncertainty and existential doubt. Their wisdom, patience, generosity and kindness have shaped me far beyond academia, and made this PhD journey truly unforgettable. I also owe my thanks to Professor Bobby Bhattacharjee, Professor Dinesh Manocha, as well as Migo Gui for their kinds words and support, and gestures to help during my difficult times.

I am sincerely grateful to my committee members—Professor Dinesh Manocha, Professor Huaishu Peng, Professor Vanessa Frias-Martinez, and Professor Ming Lin—for their time, support, and invaluable feedback throughout this process.

I want to thank my family—my mom, dad, and grandma—whose unwavering love and support grounded me during this journey and gave me the courage to keep going. There were times of deep uncertainty, when I questioned everything and felt close to giving up, but your encouragement reminded me that I always had something to return to. No matter how far away we were, or how alone I felt in those moments, you remained my anchor and my source of buoyancy—always there, quietly holding me up. Because of you, I was able to find my way back, again and again.

My path into research would not have been possible without the early support and mentorship of Dr. Paul Strohmeier, Professor Chris Harrison, Professor Craig Shultz, Professor Roberta Klatzky, and Professor Jun Nishida. Their encouragement opened the door for me to pursue this path, and played a key role in helping me obtain the NSF GRFP fellowship.

I am deeply grateful for the support of the NSF Graduate Research Fellowship Program. Beyond financial support, it provided me with a sense of security and the freedom to explore, take risks, and grow as an independent researcher. That freedom shaped not only the direction of my work, but also

the way I learned to think.

I would also like to thank the incredible mentors and collaborators I met during my time in industry research—Dr. Xinwei Chen, Dr. Xiaofan Zhang, Dr. Ruobing Han, and Dr. Ruofei Du. Your guidance not only helped me grow as a researcher and engineer, but also gave me the confidence and opportunity to take the next step into my career.

To Dr. Geonsun Lee—my mentor, collaborator, and dearest friend—thank you for being by my side through research, job searching, and life's many turning points. Your support helped me navigate both professional and personal challenges, and your presence made even the hardest moments feel lighter.

I am grateful to my UMD PhD peers—Yoon Kyung Shon, Kyungyeon Lee, and Wooje Chang—for the many conversations about research, grad school, and life that made this journey feel shared rather than solitary.

I also want to thank my collaborators and friends from my time in the Max Planck joint program in Saarbruecken, Germany—Nihar Sabnis, Gaby Vega, Yu Jiang, Madalina Nicolae, Dr. Easa AliAbbasi, and Maelle Roche. The summers we spent together—filled with music, research, and honest conversations —remain some of the most meaningful memories of this journey.

To my friends from CMU 407S—Hongyu Mao, Daehwa Kim, Catherine Tianhong Yu, Yuyu Lin, Wesley Deng, Violet Yinuo Han, Yibo Fu, Vimal Mollyn, Vivian Shen, and many others—thank you for being part of the earliest days of this path. The late nights working together, and the encouragement you gave me before I even began this PhD, gave me the courage to pursue it in the first place.

I am especially grateful to Professor Elizabeth Ximeng Shen, whose friendship and belief in me, along with our late-night conversations, helped me heal, grow, and stay energized throughout this

journey.

Finally, I want to thank my friends—Kriti Singh, Georgia Yixin Bai, Elijah Rakha-Sheketoff, and Dat Dao and many more —for the countless hours spent coworking, talking through anxieties, and supporting each other. Beyond that, thank you for all the moments of joy—our spontaneous hangouts, shared meals, small adventures, and the simple act of spending time together. You made this journey not only more productive, but also fuller, lighter, and far more meaningful.

There are many others who have supported me along the way — through conversations, collaborations, and quiet acts of kindness — whom I may not have named here individually, but whose impact I carry with me and deeply appreciate.

Table of Contents

Acknowledgements ii

Table of Contents v

List of Tables viii

List of Figures ix

Chapter 1: Introduction 1
- 1.1 The Human Middleware Problem . . . 1
- 1.2 The Burden of Translation . . . 3
- 1.3 Thesis: Reasoning Layers as Mediators . . . 6
- 1.4 From Material to Agent . . . 8
- 1.5 DEXAS Framework . . . 12
- 1.6 Contributions . . . 14
- 1.7 Organization of This Dissertation . . . 16

Chapter 2: Mediation at the Physical Layer 18
- 2.1 Pre-Reflective Mediation: Theoretical Grounding . . . 19
- 2.2 Motionless Movement: Towards Vibrotactile Kinesthetic Displays . . . 23
  - 2.2.1 The Problem: A Sense Without a Display . . . 23
  - 2.2.2 A Taxonomy of Kinesthetic Displays . . . 25
  - 2.2.3 Theoretical Grounding: Sensorimotor Contingencies and Material Experience . 27
  - 2.2.4 System Implementation . . . 29
  - 2.2.5 Evaluation: Four Studies . . . 32
- 2.3 Surface I/O: Encoding Affordances in Material Form . . . 36
  - 2.3.1 The Problem: Interface Without Explanation . . . 36
  - 2.3.2 Multi-Scale Architecture: Three Registers of Surface Information . . . 37
  - 2.3.3 Implementation . . . 41
  - 2.3.4 Evaluation . . . 43
  - 2.3.5 Three Example Interfaces . . . 45
- 2.4 Cross-Project Synthesis: What Physical Mediation Requires . . . 47
  - 2.4.1 The Shared Logic . . . 47
  - 2.4.2 Two Requirements for Pre-Reflective Mediation . . . 48
  - 2.4.3 The Boundary This Work Defines . . . 50
- 2.5 DEXAS Principles Established in This Chapter . . . 51

Chapter 3: Mediation at the Threshold of Autonomy 53
- 3.1 The Cognitive Layer: Theoretical Grounding . . . 54
- 3.2 Sustainflatable: When Autonomous Systems Must Show Their Mechanism . . . 57
  - 3.2.1 The Problem: Making Electronic-Free Autonomous Actuation Systems Explainable 57
  - 3.2.2 The Design Tool as Reasoning Layer . . . 60

3.2.3 What Sustainflatable Establishes . . . 62
3.3 In-Situ Presenter: Reading Intent Across Modalities . . . 64
3.3.1 The Problem: Virtual Presentation Without Physical and Spatial Cues . . . 64
3.3.2 ITalk-57: Grounding the Design in Observed Behavior . . . 66
3.3.3 System Architecture: A Unified Semantic Representation . . . 68
3.3.4 Evaluation: Reducing Translation Burden While Preserving Expressive Agency 71
3.4 Cross-Project Synthesis: What Mediation at the Cognitive Layer Requires . . . 74
3.4.1 One Requirement for Two Directions . . . 74
3.4.2 The Boundary Between Pre-Reflective and Reflective Mediation . . . 75
3.4.3 The Irreducibility of Cognitive-Layer Requirements . . . 76
3.5 DEXAS Principles Established in This Chapter . . . 77

Chapter 4: The DEXAS Framework 80
4.1 The Empirical Record: What the Projects Established and Where Their Limits Are At . 81
4.1.1 Surface I/O: Structured Decoding at the Material Layer . . . 81
4.1.2 Motionless Movement: Coupling Feedback to Action . . . 82
4.1.3 Sustainflatable: Transparency as an Architectural Requirement . . . 84
4.1.4 In-Situ Presenter: Unification of Heterogeneous Signals . . . 85
4.2 The Five DEXAS Principles . . . 86
4.3 The Four Structural Components of DEXAS . . . 94
4.3.1 Unified Semantic Workspace . . . 94
4.3.2 Collaborative Agentic Workflows . . . 95
4.3.3 Human-Workflow-Informed Reasoning . . . 96
4.3.4 Extensible Knowledge Base . . . 97
4.4 How the Principles Interact: DEXAS as a Coherent Framework . . . 98
4.5 DEXAS and the Existing Literature . . . 100
4.6 Summary: DEXAS as a Framework for Reasoning Layers . . . 102

Chapter 5: PROMPTS — An Agentic Reasoning Layer for ML System Optimization 104
5.1 The Problem: Human Bottleneck in ML System Tuning . . . 106
5.1.1 Why Sharding Optimization Is Hard . . . 106
5.1.2 Why Existing Approaches Miss . . . 107
5.2 PROMPTS as a DEXAS Instantiation . . . 109
5.2.1 Decoding . . . 109
5.2.2 Embodied Feedback Coupling . . . 110
5.2.3 eXposed Reasoning Transparency . . . 111
5.2.4 Adaptive Input Unification . . . 112
5.2.5 Scalable Knowledge Grounding . . . 113
5.3 Evaluation . . . 114
5.3.1 Search-Space Compression . . . 115
5.3.2 Solution Correctness Under Mediation . . . 116
5.3.3 Reasoning Legibility . . . 117
5.3.4 Extensibility via Knowledge Injection . . . 118
5.3.5 Human-Role Reconfiguration . . . 119
5.4 Cases in Depth . . . 120

5.4.1 Three Archetypal Bottleneck Cases . . . 120
5.4.2 Case 4: Boundary Conditions and Compiler Validation . . . 121
5.4.3 Zero-Knowledge Stress Test . . . 122
5.5 PROMPTS and the Dissertation's Argument . . . 123

Chapter 6: Discussion 126
6.1 The Mediation Spectrum Revisited . . . 126
6.2 Three Cross-Cutting Findings . . . 128
6.2.1 Intermediate Artifacts Are the Unit of Mediation . . . 128
6.2.2 Opacity Is a System Failure, Not a Design Trade-off . . . 129
6.2.3 The Human Role That Reasoning Layers Preserve . . . 131
6.3 Limitations and Future Directions . . . 132
6.3.1 Evaluation and Alignment of Agent Output . . . 132
6.3.2 From Sharding to General Optimization . . . 134
6.3.3 The Scope of DEXAS . . . 134
6.4 Closing . . . 135

Bibliography 138

List of Tables

5.1 Summary of experimental configurations for extensibility testing using real production workloads 114
5.2 Summary of Experimental Results 114

List of Figures

1.1 DEXAS framework: reasoning layers as mediators 13

2.1 Motionless Movement concept overview 24
2.2 Taxonomy of kinesthetic display types 26
2.3 Motionless Movement system pipeline 29
2.4 Motionless Movement prototype hardware 30
2.5 Four vibrotactile haptic signal mappings 32
2.6 Motionless Movement Study 3 and 4 physical and VR setup 34
2.7 Study 4 belief-in-movement results 35
2.8 Surface I/O system overview on a VR headset 37
2.9 Surface I/O stimulus set: 35 macro- and meso-scale features 40
2.10 Surface I/O micro-scale textures and vibroacoustic spectra 42
2.11 Surface I/O Amazon Echo Example. 45
2.12 Surface I/O Driving Wheel Example. 46
2.13 Surface I/O VR headset slider in use 46

3.1 Sustainflatable three-phase energy harnessing architecture 58
3.2 Sustainflatable application examples 59
3.3 Sustainflatable design tool interface 61
3.4 Sustainflatable computational model workflow 63
3.5 The online presentation attention problem 65
3.6 In-Situ Presenter layout-aware placement and gesture-triggered repositioning 66
3.7 ITalk-57 dataset overview 67
3.8 In-Situ Presenter system architecture 69
3.9 In-Situ Presenter two-stage processing pipeline 71
3.10 In-Situ Presenter presenter-side evaluation results 72

5.1 Overview of the PROMPTS agentic performance optimization framework 105
5.2 PROMPTS agent vs. blackbox baseline search effort reduction 115

Chapter 1: Introduction

1.1 The Human Middleware Problem

It is a Wednesday afternoon at a large ML infrastructure lab, and an engineer has just launched a profiling run on a 2048-chip TPU cluster. The job is training a mixture-of-experts language model — one of the largest configurations the team has attempted. A few hours later, the profiler finishes. What it returns is not an answer. It is data: hundreds of lines of HLO trace logs, roofline charts comparing compute and memory bandwidth, utilization breakdowns by device, sharding statistics across a 3D torus interconnect (ICI). The answer — an optimized sharding configuration that will actually make this system run more efficiently — is in there somewhere. The engineer knows this. She also knows that finding it will require days of manual cross-referencing: comparing the roofline data against scaling guides, inferring bottleneck types from utilization patterns, mapping diagnoses to configuration changes, running trial jobs to validate, and starting the loop again when the results are inconclusive. She has done this before. She will do it again. This is not the hard part of her job — it is the part that makes the hard part harder. It is also the part that, after enough Wednesdays, wears her down. A significant portion of her expertise is not being applied to the optimization problem itself, but to the act of translation: converting what the system reports into a form she can reason about, before she can even begin to reason about it.

This problem is not unique to ML infrastructure. Consider a researcher giving an online presentation about her work. Her slides fill most of the screen; her video feed is pinned to a corner — the same corner, for the entire talk. In a physical conference room, she would move. She would step toward the diagram she is describing, gesture at the equation she wants the audience to follow, shift aside when the slide needs full attention. Her body would do the work of spatial coordination, and the audience

would read it naturally. This matters beyond clarity: a talk is a chance to be remembered, to have the work and the person behind it leave an impression. The best speakers do this not just through what they say but through how they inhabit the room — the timing, the emphasis, the presence that makes an idea feel like it belongs to someone. Online, none of that happens automatically. To achieve any spatial coordination at all, she would need to manually translate her intent into real-time controls mid-talk: drag her thumbnail left for this slide, resize when she gestures, reposition when the layout changes. The cognitive cost is high enough that most presenters never bother — they settle for a fixed layout and sacrifice the potential expressiveness.

A similar gap confronts anyone trying to communicate movement. A physical therapist working with a stroke patient wants to convey what a correct arm movement feels like — not how it looks from the outside, but the internal sense of force, timing, and resistance that the patient needs to rebuild. She can demonstrate. She can describe. She can place her hands on the patient's arm and try to guide it through the motion. But none of these are the thing itself. Kinesthesia — the sense of movement — has no established display medium. There is no channel through which one person's embodied experience of motion can be transmitted directly to another. So the therapist becomes the translation layer: converting an internal, pre-reflective experience — one that inherently lives in the body, below the level of words or thoughts — into external, symbolic approximations that the patient must then re-translate back into body knowledge. The gap here is not "informational" but sensory. She knows exactly what she wants to convey. The medium for conveying it does not exist.

These three scenarios — the ML engineer, the presenter, the therapist — span different domains, different scales of system, and different kinds of expertise. But they share a common structure. In each case, an expert's intent — what she is trying to do, express, or communicate — cannot reach its destination directly. Something is lost in the gap between human purpose and system representation.

That gap is what experts are left to manage, each in their own way: the engineer decodes trace logs for days before she can even formulate a hypothesis; the presenter translates spatial intent into pixel coordinates mid-talk; the therapist converts embodied knowledge into symbolic instruction and hopes something survives the translation. The gap takes different forms — sometimes it is sensory, a missing channel between one body's knowledge and another's; sometimes it is representational, a mismatch between what a system produces and what a human mind can directly use. But in every case, it imposes a cost: attention redirected from judgment to translation, from higher-order reasoning to the mechanics of making two systems speak to each other. This is the human middleware problem.

1.2 The Burden of Translation

The term *middleware*, in software engineering, refers to a layer of software that sits between two systems and handles translation between them — converting formats, routing messages, managing state. When no such layer exists, the translation work falls on whoever is operating the systems. In knowledge-intensive domains, that person is typically an expert. The expert becomes human middleware: a cognitive layer responsible for translating between high-level intent and low-level system representations. Human intention, as used throughout this dissertation, means the goal a person is trying to accomplish expressed at her natural level of abstraction — which may be symbolic, embodied, or somewhere in between — before it has been translated into the explicit, low-level representations a system requires to act. That gap between where intention lives and where the system's representations live is where the translation burden accumulates — and where the expert becomes middleware.

The human middleware problem has three properties that make it worth addressing.

First, it is *ubiquitous*. The translation burden appears wherever there is a gap between what a human is trying to do and what a system is built to accept. That gap shows up in physical interaction design, where a user has to figure out how a surface works with no guidance beyond its shape. It shows up in embodied interfaces, where feedback arrives as sensor data or actuator output that the body was never built to read directly — requiring a layer of conscious interpretation before it means anything. It shows up in autonomous systems, where a machine's internal logic has to be made legible to the people responsible for deploying it. It shows up in agentic AI, where the distance between a natural-language request and a valid system configuration can span thousands of decisions that no single person could trace. The problem does not stay contained. It grows with the complexity of the system. A simple device with one function and one feedback signal asks little of the person operating it — she pushes a button, something happens, she knows what happened. But consider what the ML engineer is actually facing: a 2048-chip cluster producing trace logs that span gigabytes, generated by a system whose internal communication topology — a 3D torus interconnect — has no natural human analog. There is no intuition she can reach for. Every inference she draws requires active reconstruction from representations she did not choose and was not designed to read. Now extend that to an agentic system coordinating dozens of models, tools, and memory stores, each producing its own outputs, each consuming others'. The translation problem scales with complexity and spans domains. Any framework that takes it seriously must be designed to travel — extensible enough to apply not just where the problem was first encountered, but wherever it reappears.

Second, it is *cognitively expensive*. Translation work does not run in the background — it occupies the same mental resources as the actual work it is supposed to support. And it is not cheap work. It requires the same domain expertise as the judgment it is delaying. The ML engineer reading HLO traces is not doing her best work. She is decoding. Every hour she spends decoding is an hour

she does not spend reasoning about the optimization problem she was hired to solve. The presenter who is managing her thumbnail position mid-talk is not thinking about her argument — she is thinking about her cursor. The physical therapist who cannot transmit what a correct movement feels like must instead watch, describe, adjust, and watch again, burning session time on approximations when the patient has little of it to spare. In each case, the translation work is not a minor tax on the expert's attention. It is a first claim on it that is paid before any of the actual work begins.

Third, and most importantly for this dissertation, it is *constantly misaddressed*. Researchers in the field have long debated how much control to hand to machines. One side argues for intelligent agents that observe, learn, and act on the user's behalf — systems that reduce burden by taking over. The other argues for keeping the human in direct control, with the machine responding to explicit commands rather than anticipating them [1]. This tension has never been resolved, and it probably cannot be — the right balance depends on the domain, the stakes, and what kind of errors are tolerable. But in the current moment, as large models become capable of absorbing increasingly complex tasks, the pull toward automation is strong. A system that produces an output can be benchmarked; a system that augments human judgment is harder to evaluate, its value showing up slowly in the quality of decisions made downstream. We tend to build what we can measure, and automation produces cleaner metrics.

The cost of this, more often than not, is agency. When a system automates the expert's translation work without exposing how, it tends to absorb her judgment along with it. The output arrives, but the reasoning that produced it is not available for inspection. She cannot tell whether the system understood the problem or got lucky. She cannot intervene halfway through. She cannot build on what it did, because she does not know what it did. The more the system does, the less she can see — and the less she can see, the less she can correct, extend, or trust what she is given. Transparency is

sometimes added afterward — an explanation layer grafted onto a system that was not designed to be explained. But a system that was built to hide its reasoning and then asked to report on it will report what is convenient, not what is true.

This dissertation takes a different position — in search of the sweet spot between direct manipulation and full automation. The argument is that this sweet spot is reachable, and that reasoning layers are what make it reachable. A system that makes its reasoning explicit — that converts low-level complexity into structured, human-legible representations before acting on them — can automate the translation work without taking over the judgment. The engineer does not read raw traces; she reads a diagnosis and decides. The presenter does not manage pixel coordinates mid-talk; the system infers where she wants to be and places her there. The therapist does not approximate in words what can only be known through the body; the system finds a channel that the body can read directly. In each case, the translation work has been offloaded. The human intention has not.

1.3 Thesis: Reasoning Layers as Mediators

The central claim of this dissertation is that reasoning layers — intermediate representations that make system logic explicit, inspectable, and actionable — reduce user cognitive load across physical, multimodal, and agentic systems while preserving human agency. The highest-value role of intelligent systems is not to make decisions on behalf of humans, but to offload the translation work that prevents humans from making good decisions themselves. The goal is not a system that absorbs the reasoning process into something opaque and self-contained, but one that converts low-level complexity into insight that a human can read, evaluate, and act on.

This is a contrarian claim. The current discourse in AI celebrates systems that do more, decide

more, and require less human involvement. But doing more is not the same as helping more. This dissertation argues that the most valuable intelligence is not the intelligence that hides its reasoning but the intelligence that surfaces it — that the goal is not to replace the expert but to reconfigure what expertise is for. The expert should be making judgments, not translations.

To make that argument precise, I need to be specific about what good mediation actually looks like — because "reducing cognitive load" and "preserving agency" are easy things to claim and hard things to achieve simultaneously. A system can reduce cognitive load by simply hiding complexity, which is exactly what the dominant automation paradigm does. That is not mediation. So I define it carefully:

> A system effectively mediates when it reduces the mental work required to translate between what a human is trying to do and what the system needs to execute — without removing human agency over the decision process.

This definition has two components that are both essential and in tension. First, reduction of mental work is the primary goal: the system offloads the translation burden. Second, preservation of agency is the constraint: the system does not absorb decision-making authority along with it. A fully automated system satisfies the first condition but violates the second. A system that merely displays raw data satisfies neither. Effective mediation requires both — and achieving both requires something more specific than good intentions.

That something is what I call a *reasoning layer*: an intermediate representation that sits between raw system state and human action. It transforms low-level signals into human-legible form, exposes the logic by which that transformation occurred, and creates a surface for human intervention. A reasoning layer communicates process and not results — and in doing so, gives the system something

to ground its decisions on, and the human something to interrogate. The engineering challenge is making reasoning layers that work on both sides of that relationship: faithful enough to the system's internals to be technically valid, and legible enough to a human that she can actually do something with what she sees.

1.4 From Material to Agent

I did not start my research with a framework in mind; it began with an observation about how mediation works at different levels of human experience. Some mediation operates below the threshold of conscious attention: the body reads texture, follows a contour, feels resistance, and knows what it is and/or what to do without deliberation. This is pre-reflective — it bypasses the cognitive bottleneck entirely, interfacing directly to the sensorimotor system before the mind has to weigh in. Other mediation operates above that threshold: the mind receives information, interprets it, maps it to a representation, and reasons toward a response. This is reflective — and it is where cognitive load accumulates. When information arrives in a form the mind was not built to read directly, the cost is not just time spent in decoding. Decoding takes the same cognitive resources needed for the actual work and causes fatigue. The projects in this dissertation are presented not in the order they were built, but in the order the questions they answer accumulate — beginning at the material layer, where mediation is most physical and most direct, and ending at the agentic layer, where the cognitive demands are highest and the consequences of poor mediation are most severe.

The first question the research confronted was whether it was possible to create a novel mediation intuitive enough that it works at the pre-reflective level — entering the body's sensorimotor loop directly, rather than arriving as a signal the mind must first decode. For vision, a screen shows you

a photograph of a farm — the green wet grass, the horses, the cows — and you see it the way you would see the farm itself, not through a description of those things. For audio, a speaker fills the room with the sound of ocean waves, water rushing and receding, and you hear it exactly as if you were standing at the shore. In both cases, the medium delivers the sensory experience itself — embodied, immediate, requiring no symbolic intermediary between the signal and the body that receives it. But for movement — for kinesthesia, the felt sense of force, timing, and resistance entailed in one's motion — no equivalent medium exists. What we have instead are demonstrations, verbal instructions, and manual guidance: all symbolic, all requiring the recipient to translate the signal back into body knowledge. *Motionless Movement* (CHI '24) investigated this gap through the lens of kinesthesia. The physical therapist's problem described in the opening of this chapter — conveying what a correct arm movement feels like from the inside — is precisely a failure of pre-reflective mediation: the knowledge she wants to transmit is embodied, but every channel available to her is symbolic. She demonstrates, describes, manually guides — each of these requires the patient to re-translate the signal back into body knowledge, at precisely the moment when that translation is hardest to make. We found that vibrotactile feedback, when synchronized to a user's applied force, produces a stronger and more believable kinesthetic experience than visual feedback, even when no physical movement occurs. The feedback enters the sensorimotor loop rather than requiring conscious interpretation — which pointed to something worth taking seriously: that mediation is most powerful not when it communicates to the mind, but when it communicates to the body directly. The pre-reflective channel, when it can be reached, is the most direct form of mediation there is.

If active feedback can be structured to enter the sensorimotor loop directly, the same question applies to form itself: can surface geometry encode interaction affordances in ways the body can read — not through labels or instructions, but through the logic of shape itself, so that the reflective

overhead is minimized and the finger largely knows where to go before the mind has to decide? *Surface I/O* (CHI '23) tested this at the level of material structure. The hypothesis was that surface geometry — macro-scale contours that guide a finger's exploration, meso-scale textures that signal material properties, micro-scale features that produce distinctive vibro-acoustic spectra enabling input classification — could constitute an interaction language that users read without instruction or visual guidance. At the macro and meso scales, the geometry speaks directly to the sensorimotor system: a rail guides the finger, a valley signals a boundary, a detent marks a discrete state. At the semantic scale — inferring that a circular feature is a dial, a linear one a slider — some learned convention is required. But Surface I/O's contribution is precisely that it structures this vocabulary systematically, so that the reflective overhead is as thin as possible: the user learns the language once, and the geometry does the rest. The result was a single passive surface, fabricated without mechanical components, that can simultaneously encode touch affordances and enable input sensing. Form and interface, which have historically been designed as separate layers, collapse into one. The surface becomes the reasoning layer — not by eliminating interpretation entirely, but by absorbing enough of it that the user no longer needs to look, confirm, and consciously reason their way through the interface. What remains is navigation. What is freed is attention.

Physical mediation, however, operates under a fundamental constraint: it works when the system's role is passive — encoding, responding, reflecting the user's own action back to her in a structured form. It breaks down when the system must act autonomously — when it has its own sense-reason-act loop, makes decisions without direct human instruction, and the human must understand and shape what that loop will do. At that point, the mediation problem changes character. It is no longer about perceiving affordances pre-reflectively. It is about modeling another agent's behavior — understanding not just what a surface feels like, but what an autonomous system will do next and why. This is an

irreducibly cognitive problem.

Sustainflatable (UIST '23) supported my exploration of this topic. In developing an electronic-free pneumatic system that harvests renewable energy and operates autonomously — with no valves, no electronics, and no real-time human control — we discovered that the central design challenge was not the pneumatics. It was human understanding and prediction of autonomous system actions. Without a tool that allowed designers to preview the system's behavior, simulate its temporal dynamics, and configure its autonomous logic through explicit representations of internal state, the system was unusable despite being technically correct. Even in a system designed to operate without human involvement, a reasoning layer is not optional. It is the precondition for the system being deployable at all.

In-Situ Presenter (in submission) sits at the boundary between these two modes. Where Sustainflatable revealed that autonomous systems need to expose their reasoning outward — making internal state legible to the humans responsible for them — In-Situ Presenter revealed that intent-driven systems face the inverse challenge: reading embodied human signals inward and converting them into cognitive-level representations the system can act on. A presenter's intent is not stated — it is expressed, even implicitly. It is distributed across gesture, speech, and slide structure simultaneously, operating at a pre-reflective level for the presenter themself. The system must absorb those signals, unify them into a shared semantic representation, and reason about spatial intent from a multimodal substrate. The reasoning layer here is not an output but an internal architecture: a unified representation space that bridges the embodied and the cognitive, making the system capable of inference it could not otherwise perform. This is where the physical and cognitive threads of the research converge: the system must read the body's language and translate it into something a reasoning process can use.

1.5 DEXAS Framework

These four projects, taken together, revealed something I had not anticipated when I began: that effective mediation, across all of these domains, depends on the same underlying set of requirements. Systems must decode raw signals into structured, human-legible representations. They must couple feedback to human action in ways that the body can read without the mind having to step in — where the system's response feels like a natural consequence of what the user did, rather than a signal that needs to be separately interpreted. They must expose their intermediate reasoning so humans can inspect, correct, and build on it. And they must unify heterogeneous inputs into shared representations that support downstream inference.

But there is a fifth requirement, one that the earlier projects hint at and that the ubiquity of the problem makes unavoidable: the mediation layer must be extensible. The human middleware problem does not respect domain boundaries — it shows up in physical surfaces, embodied feedback, autonomous systems, and agentic AI. A framework that addresses it in one place but cannot travel to another is not a general solution. It is a local fix. The mediation must be able to grow as the system grows, incorporate new knowledge without structural redesign, and compose across agents and tasks as the problem scales.

These five requirements — Decoding, Embodied feedback coupling, eXposed reasoning transparency, Adaptive input unification, and Scalable knowledge grounding — form what I call DEXAS. It is not a framework I designed at the outset. It is a set of principles I found myself returning to, across five very different systems, each time I asked why the mediation worked, why it fell short, and whether it could be made to work somewhere else.

PROMPTS (MLSys '26) takes the principles of DEXAS and applies them to engineers' workflow

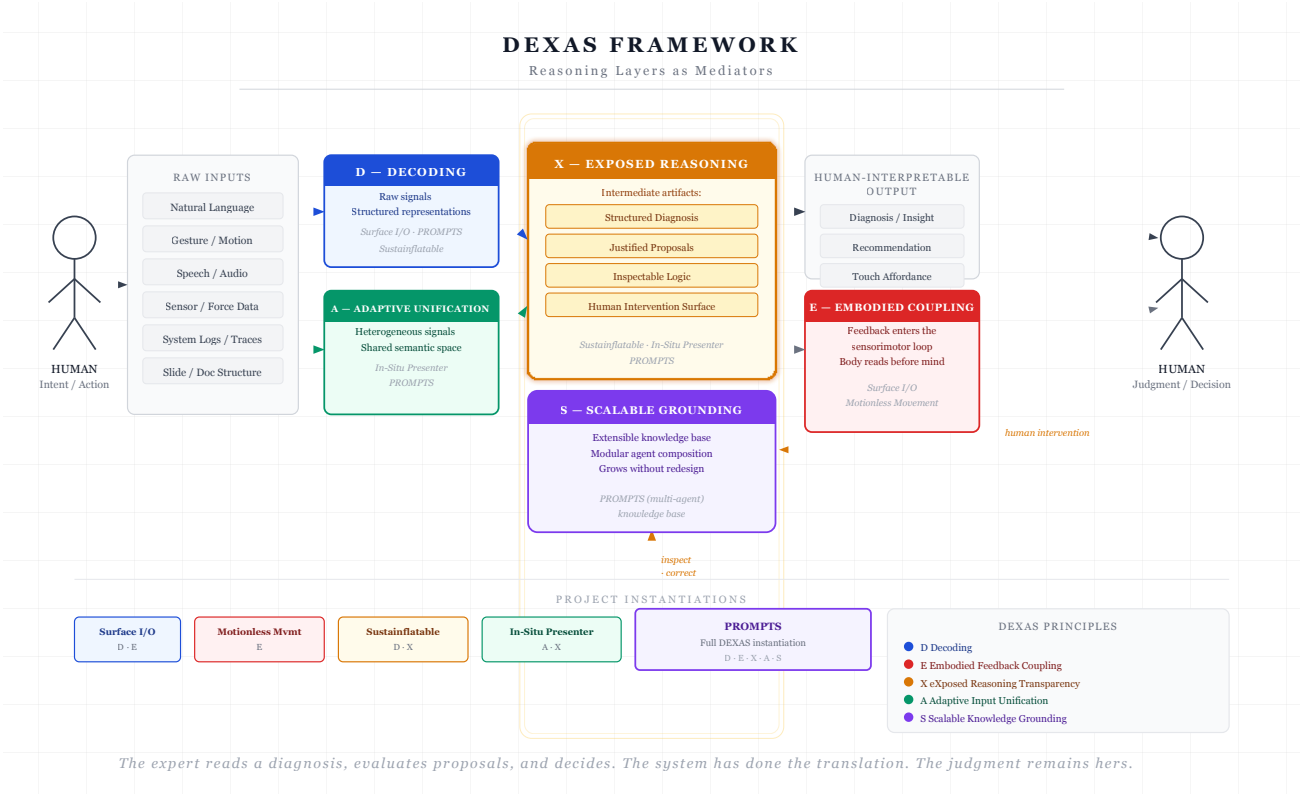


Figure 1.1: The DEXAS framework. A human with intent or action (left) provides raw inputs — natural language, gesture and motion, speech and audio, sensor and force data, system logs and traces, and slide or document structure — which are processed through five principles before producing human-interpretable output for judgment and decision (right). **D (Decoding)** transforms raw signals into structured representations; instantiated in Surface I/O and PROMPTS. **A (Adaptive input unification)** maps heterogeneous signals into a shared semantic space; instantiated in In-Situ Presenter and PROMPTS. **X (eXposed reasoning transparency)** produces intermediate artifacts — structured diagnosis, justified proposals, inspectable logic, and a human intervention surface — making the system's reasoning accessible for inspection and correction; instantiated in Sustainflatable, In-Situ Presenter, and PROMPTS. **S (Scalable knowledge grounding)** maintains an extensible knowledge base through modular agent composition that grows without architectural redesign; instantiated in PROMPTS (multi-agent knowledge base). **E (Embodied feedback coupling)** ensures feedback enters the sensorimotor loop so the body reads before the mind interprets; instantiated in Surface I/O and Motionless Movement. The bottom row maps each of the five projects to the DEXAS principles it instantiates; PROMPTS is the sole full instantiation of all five (D · E · X · A · S).

in optimizing industrial-scale workloads. Designed from the beginning as a full instantiation of all five DEXAS principles, it addresses the ML infrastructure problem described in the opening of this chapter. It takes raw profiler output — HLO traces, roofline data, memory logs — and transforms it into a structured bottleneck diagnosis. From the diagnosis, a multi-agent architecture generates sharding configuration proposals with explicit justifications, grounded in an extensible knowledge base of expert heuristics and past optimizations. The engineer does not read traces. She reads a diagnosis, evaluates proposals, and decides. The translation work has been offloaded. Her judgment has been preserved. Crucially, the system is designed to grow: new agents can be added, new heuristics incorporated, and the knowledge base extended without restructuring the architecture. Across eight real-world LLM workloads, PROMPTS independently identifies expert-validated configurations within its top three recommendations every time, with its top-ranked recommendation matching the production configuration in 87.5% of cases — and achieving up to 434% performance improvement over baseline.

1.6 Contributions

This dissertation makes six contributions to the understanding of reasoning layers as a systems abstraction for human-machine mediation.

Motionless Movement demonstrates that action-coupled vibrotactile feedback is more effective than visual feedback in conveying kinesthetic experience, and that this advantage arises because the feedback operates inside the user's sensorimotor loop rather than requiring symbolic interpretation. This establishes that the representational channel through which mediation operates determines its effectiveness, independent of information content.

Surface I/O demonstrates that surface geometry alone can encode interaction affordances across

multiple scales of abstraction, collapsing the distinction between physical form and interactive interface. This establishes that mediation can be embedded in material structure, not just software behavior.

Sustainflatable demonstrates that autonomous physical systems require human-facing reasoning interfaces as a precondition for deployment — that legibility, predictability, and configurability are not optional features but architectural requirements. Without a reasoning layer, an autonomous system may be technically correct and still be unusable.

In-Situ Presenter demonstrates that intent-driven interaction across heterogeneous signal modalities requires a unified semantic representation as an intermediate layer — that no single modality suffices and that the quality of the shared representation determines the quality of intent inference downstream.

DEXAS is a general framework for reasoning-layer mediation, synthesizing five principles — structured signal decoding, embodied feedback coupling, exposed reasoning transparency, adaptive input unification, and scalable knowledge grounding — discovered across the above projects. DEXAS defines what a system must provide to effectively mediate between human intent and system execution, and to do so in a way that can travel across domains and scale with the complexity of the problem.

PROMPTS is a complete instantiation of DEXAS in the high-stakes domain of large-scale LLM system optimization, demonstrating that expert-style diagnostic reasoning, made explicit and extensible through a multi-agent architecture, can match and in some cases exceed the performance of black-box configuration search — while remaining inspectable and steerable at every step.

Together, these contributions establish reasoning layers as a coherent systems abstraction: a design primitive that can be instantiated across scales, from passive material surfaces to multi-agent agentic workflows, and that consistently produces outcomes that reduce translation burden, align with human intent, and support human decision-making rather than replacing it.

1.7 Organization of This Dissertation

The chapters that follow are organized to build the empirical case before making the theoretical claim, and to validate the claim before discussing its implications.

Chapter 2 examines mediation at the physical layer. I present Surface I/O and Motionless Movement as investigations of how form and feedback can encode interaction without requiring symbolic interpretation. These projects establish the first two DEXAS principles — structured decoding and embodied feedback coupling — and demonstrate that effective mediation often operates below the threshold of conscious reasoning, entering the body's action-sensation loop directly rather than passing through it.

Chapter 3 examines mediation in autonomous and multimodal systems. Sustainflatable and In-Situ Presenter reveal that effective mediation requires bidirectionality: systems must infer intent from heterogeneous human signals and expose their own reasoning in inspectable form. These projects establish the next two DEXAS principles — reasoning transparency and adaptive input unification — and motivate the framework that follows.

Chapter 4 presents DEXAS as a synthesis. I derive the five principles from my empirical record, define them precisely, and show how each prior project instantiates a subset of the full framework. I argue that DEXAS is not a design prescription but a set of requirements that any effective mediating system must satisfy, and I show how the principles interact and reinforce one another.

Chapter 5 presents PROMPTS as a DEXAS instantiation and primary empirical validation. I describe the system's multi-agent architecture, map its components to all five DEXAS principles — including scalable knowledge grounding, which only becomes fully visible at this scale — and present evaluation results across eight real-world LLM workloads. I argue that PROMPTS demonstrates not

only that reasoning layers are effective in practice but that when reasoning is made explicit, inspectable, and extensible, it can match or exceed black-box methods while remaining open to human intervention.

Chapter 6 discusses the implications of this body of work for the design of intelligent systems. I examine the relationship between transparency and trust, the design of extensible knowledge representations, and the broader question of what human-agent partnership should look like as agentic AI systems become more capable. I close with directions for future work, including evaluation frameworks for reasoning quality and methods for aligning agent behavior with human judgment over time.

* * *

The argument of this dissertation is a simple one: the value of an intelligent system does not lie in how much it does without human input, but in how much it supports human intent and decision-making. The engineer who reads a diagnosis is more powerful than the engineer who reads a trace — not because the system has done her work for her, but because the system has done the translation, and left the decision to her. That is the goal. That is what reasoning layers make possible.

Chapter 2: Mediation at the Physical Layer

This chapter examines how mediation operates when it is most direct — when the intermediary representation is not language, software, or an autonomous agent but physical form itself: the geometry of a surface, the coupling of feedback to action, the texture of a material. The two projects discussed here, Motionless Movement (CHI '24) [2] and Surface I/O (CHI '23) [3], were not conceived as a pair. They emerged from different research groups, address different perceptual channels, and deploy different technical approaches. But read together, they reveal a shared logic — and that shared logic is worth naming precisely before either project is described in detail, because it shapes how the empirical findings should be interpreted.

Both projects investigate the conditions under which mediation can operate below the threshold of conscious interpretation — where the body reads a surface, or receives feedback, without first routing that information through deliberate reasoning. Motionless Movement approaches this question through feedback coupling: when vibrotactile pulses covary with force application in the same way tactile signals covary with actual movement across a surface, the body treats the feedback as evidence of movement rather than as a signal to be decoded. Surface I/O approaches the same question through tactile affordances encoded in surface geometry: at the macro and meso scales, contours guide the finger, detents mark discrete states, and rough patches signal boundaries — all legible through the exploratory procedures by which humans naturally investigate objects through touch [4–6]. In Norman's terms, these are perceived affordances — properties of the surface that signal to the user what actions are possible, without instruction or explanation [7]. Some interpretive inference remains — a circular feature reads as a dial only through learned convention, which Norman would call a signifier rather than an affordance proper — but the system is structured to make that inference thin and fast. In both cases, the target is what Don Ihde calls embodied mediation — where the technology becomes quasi-

transparent, felt through rather than looked at, as a blind person's cane eventually dissolves into felt contact with the ground — rather than hermeneutic mediation, where a representation must be actively interpreted [8, 9].

The chapter proceeds as follows. Section 2.1 provides the theoretical background that frames both projects: the concept of sensorimotor contingencies as the basis for pre-reflective interaction, the distinction between embodied and hermeneutic mediation, and the specific problem of kinesthesia as a channel without a medium. Section 2.2 presents Motionless Movement — the design problem of kinesthetic display, the prototype system, the four studies, and the empirical findings. Section 2.3 presents Surface I/O in full — its design rationale, its multi-scale architecture, its implementation, and its evaluation results. Section 2.4 draws the cross-project synthesis: what the two projects share, what that commonality establishes about the requirements for pre-reflective mediation, and how those requirements inform the DEXAS framework developed in Chapter 4.

2.1 Pre-Reflective Mediation: Theoretical Grounding

The distinction between pre-reflective and reflective cognition runs through phenomenology, ecological psychology, motor control research, and a long tradition of HCI work on directness, tangibility, and embodied interaction. What it has not, until recently, been connected to is the specific engineering problem of interface design — the question of what a system must do to achieve mediation at the pre-reflective level rather than merely aspiring to it.

The foundational theoretical resource here is the work of O'Regan and Noë on sensorimotor contingencies [10]. Their central claim is that what distinguishes one sensory modality from another is not the physical phenomenon it detects but the learned patterns of correspondence between motor

activity and sensory change that accompany it. When the eye moves right, the visual field shifts left — that is a visual sensorimotor contingency. When the finger moves across a textured surface at speed, vibrations occur at a frequency proportional to that speed — that is a tactile sensorimotor contingency. Perception, on this account, is not a passive registration of signals but an active exploration of the world structured by the body's learned expectations about how action and sensation covary in synchronization. The body does not receive information and then decide what to do. It acts, and it knows through acting.

This matters for interface design because it implies that a display medium does not need to reproduce a physical phenomenon in order to produce the corresponding perceptual experience. What it needs to do is reproduce the sensorimotor contingency — the lawlike relationship between the user's motor activity and the resulting sensory change. If that relationship is preserved, the experience follows. If it is broken — if the feedback arrives too late, too uniformly, or in a form that does not covary with the user's action in the expected way — the perceptual loop collapses. The user stops feeling through the interface and starts reading it. A surgeon operating through a robotic system that strips haptic feedback must infer tissue resistance from visual cues alone — the sensorimotor loop that constitutes surgical skill is severed, and what remains is interpretation under pressure. A driver whose electric power steering has decoupled road feel from steering input loses the covariance between hand effort and surface feedback that experienced drivers use to sense traction at the limit; they are still receiving information about the road, but now through a channel that requires conscious attention rather than one that flows directly into action. What was quasi-transparent becomes an object requiring interpretation. The interface has not failed to provide information — it has failed to provide information in a form the body can absorb without the mind having to intervene. The implication for kinesthetic display is direct: the key engineering problem is not to identify the physical phenomenon underlying kinesthesia and reproduce it, but to identify the sensorimotor contingencies that accompany

kinesthetic experience and build a system that instantiates those contingencies in a different medium.

The second theoretical resource is Ihde's taxonomy of technological mediation [11]. In embodied mediation, the technology becomes quasi-transparent: the user acts through it rather than attending to it. A blind person's cane is the paradigm case — after practice, the experienced cane user does not consciously attend to the pressure of the cane against the palm; they attend to the texture of the pavement at the cane's tip. The cane has become part of the body schema. In hermeneutic mediation, by contrast, the technology presents a representation that must be read and interpreted. A thermometer gives information about the weather, but that information arrives as a number the mind must convert back into a felt sense of hot or cold. The number is not pre-reflective. Neither are most digital interfaces: a slider position, a progress bar, a roofline chart.

What makes hermeneutic mediation costly is precisely its reflective character. Interpretation takes the same cognitive resources as the work it is supposed to enable. An engineer reading a utilization percentage to infer a bottleneck type is doing a cognitive task before she can do the task she was asked to do. A patient reading a physical therapist's verbal description of what a correct arm movement feels like must translate that description back into body knowledge — at precisely the moment when that translation is hardest to make. The cost is not merely attentional. It is motivational, temporal, and structural: it inserts an interpretive step between intention and action that the interface was supposed to eliminate.

The third resource is the specific problem of kinesthesia — the felt sense of movement — as what we might call a displayless modality. Vision has screens. Audition has speakers. Even the chemical senses, like warm and cool sensations, have research-grade display devices. But kinesthesia has no established display medium. The reasons are structural: unlike vision and audition, kinesthesia is an internal, proximal sense whose phenomenon originates within the body rather than arriving from

outside. There is no single clearly bounded physical correlate — kinesthetic experience depends on a combination of muscle spindle activity, joint receptor signals, cutaneous mechanoreceptors, and vestibular input, all interacting in ways that have resisted systematic manipulation. Critically, the experience is not simply a readout of these signals but is shaped by whether movement is self-generated. When the motor system issues a movement command, it simultaneously generates an efference copy — a predictive signal against which incoming sensory feedback is compared [12]. Active movement feels as it does partly because the nervous system anticipated it. Passive movement — an arm guided by a therapist, a wrist steered by an exoskeleton — activates the same peripheral receptors but arrives without that predictive context, and the experience is categorically different: felt as something happening to the body rather than something the body is doing. Standard display approaches that attempt to reproduce kinesthetic experience by producing the relevant physical phenomenon — moving the limb, stimulating the muscles — run directly into this distinction. The phenomenon they reproduce is peripheral; the experience they are trying to evoke is central.

What O'Regan and Noë's framework suggests is an alternative: instead of reproducing the physical phenomenon of kinesthesia, reproduce the sensorimotor contingency that accompanies it. When a finger moves across a surface, vibrations occur at a rate proportional to movement speed. That is the contingency. If a display can couple its feedback to the user's action attempt in a way that preserves that contingency — making the signal change as the user exerts force, at a rate that matches what movement over a surface would produce — the body may experience that feedback as movement, even if no movement occurs. This is the insight that drives Motionless Movement, and it is worth stating it at this level of abstraction before the specific implementation is described, because the insight is more general than any particular prototype.

These three theoretical foundations do not bear equally on both projects. The embodied/hermeneutic

distinction is genuinely shared — in each case, the design target is mediation the body can absorb without conscious interpretation. But the sensorimotor contingencies framework and the problem of kinesthesia as a displayless modality are the theories behind Motionless Movement specifically, which is an attempt to instantiate a kinesthetic contingency in a new medium. Surface I/O draws on a related but distinct tradition — Norman's account of perceived affordances and Lederman and Klatzky's haptic exploratory procedures — to ask a parallel question through geometry rather than feedback coupling. What unifies them is the underlying research vision: to push as much of the work of mediation as possible below the threshold where conscious cognition has to intervene. The two sections that follow examine how far that vision can be realized, and where it runs out.

2.2 Motionless Movement: Towards Vibrotactile Kinesthetic Displays

2.2.1 The Problem: A Sense Without a Display

The physical therapist's problem stated in Chapter 1 is this: she knows what a correct movement feels like from the inside, and that knowledge is not propositional. It cannot be stated fully in words or demonstrated with sufficient accuracy through imitation. It cannot even be transmitted through manual guidance — guiding a patient's arm through the correct trajectory produces passive movement, and passive movement, as the previous section established, feels categorically different from the active movement the patient needs to learn. Every channel available to her delivers the knowledge in a form the patient must translate back into body knowledge at precisely the moment when that translation is hardest. The underlying problem is that kinesthesia has no display medium. Vision and audition both have established technologies that can transport a user into a perceptual experience — a screen renders the view of an ocean, speakers render its soundscape — without the user being physically

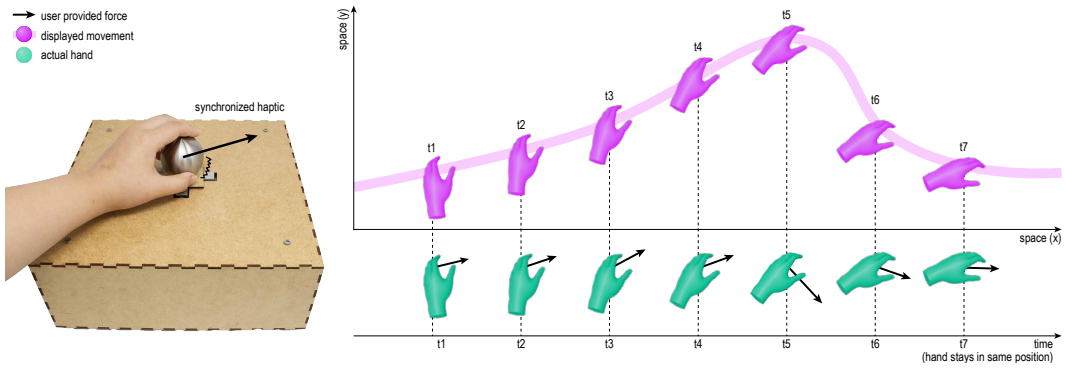


Figure 2.1: Motionless Movement concept overview. Left: the prototype device — the user applies isometric force to the spherical metal handle, which remains stationary while synchronized vibrotactile feedback is delivered through the handle. Right: space-time diagram contrasting the *displayed* movement trajectory (pink, virtual position computed from applied force) with the *actual* hand position (teal, stationary). User-provided force direction is indicated by arrows at each time step. The physical handle does not move; the body experiences movement through the action-coupled haptic signal.

present. No equivalent exists for kinesthesia. There is no device that can deliver the felt experience of movement to a body that is not moving. This is surprising given how central kinesthetic experience is to human activity. Every skilled performance — sport, musical instrument, surgical technique, manual craft — depends critically on the felt sense of movement. Yet while vision and audition have both well-established display technologies and rich research literatures on how to design them, the study of kinesthetic display was, prior to this work, fragmented across isolated prototypes with no shared theoretical framing. The contribution here is to name kinesthetic displays as a category, establish a taxonomy of their possible forms, and empirically investigate whether action-coupled vibrotactile feedback can constitute one such form.

2.2.2 A Taxonomy of Kinesthetic Displays

Before presenting the Motionless Movement prototype, this section establishes a four-way taxonomy of kinesthetic display types. The taxonomy clarifies what the motionless movement approach is and is not, and what distinguishes it from prior work in body-actuating and movement-augmenting systems.

Direct actuation: the display directly moves the user's body. EMS-based systems that steer the wrist during drawing [13, 14] or exoskeletons that train motor patterns through physical manipulation are examples. Control lies with the display; the user's own motor intent is bypassed.

Augmented perception: additional feedback is layered on top of the user's actual, self-initiated movement, providing richer or more precise information about what they are already doing. The difference between drawing with a pencil on paper versus on the glass surface of a tablet illustrates the principle: the paper provides richer sensorimotor feedback of the same movement. Friction rendering devices are a more engineered instance of the same logic. Control lies with the user; the kinesthetic display in this case amplifies rather than replaces.

Altered perception: the display manipulates the user's experience of their own movement without changing the movement itself — making people believe they can perform movements they actually cannot, or changing how they experience their bodies. Haptic retargeting in VR, where a user's physical hand is guided to touch a proxy object while the visual rendering shows them touching a different object, is the paradigm case [15]. Control can lie with either the user or the display depending on implementation.

Motionless movement: the display provides an experience of movement in the absence of any actual movement [16]. The user initiates action — exerts force, makes an effort — and the display responds in a way that makes the body believe movement is occurring. This is the hardest case in

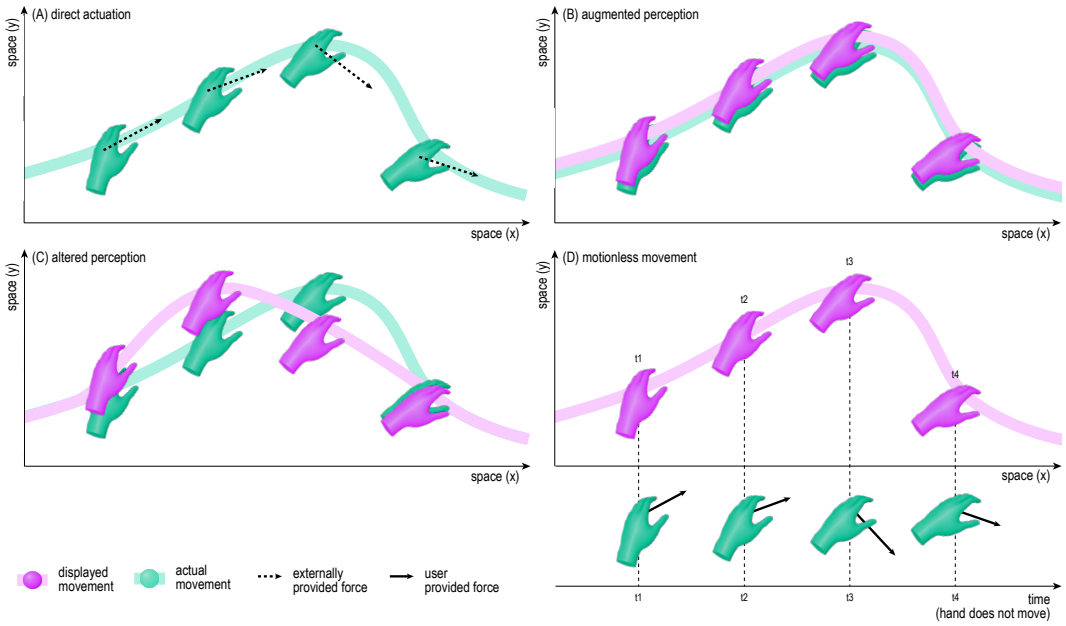


Figure 2.2: Four-way taxonomy of kinesthetic display types, illustrated as space-time diagrams contrasting displayed movement (pink) and actual movement (teal). (A) *Direct actuation*: externally provided force (dashed arrows) drives the body along a trajectory the system controls; actual and displayed movement coincide but are externally imposed. (B) *Augmented perception*: the user moves freely (user-provided force, solid arrows) while the display layers additional feedback onto the self-initiated movement. (C) *Altered perception*: displayed movement diverges from actual movement while the user provides force. (D) *Motionless movement*: the user provides force throughout, but the hand does not move — displayed movement is constructed entirely from the action-coupled feedback signal.

the taxonomy: there is no actual movement for the display to enhance or redirect, and the movement experience must be constructed entirely from the coupling of feedback to action attempt. Like altered perception, the movement experience here is user-initiated rather than externally imposed — the distinction from direct actuation is precisely that the user, not the display, holds the locus of control.

A dimension that runs through all four types is whether the movement experience is *embodied* (pre-reflective, felt as part of the user's own movement experience) or *symbolic* (hermeneutic, requiring conscious interpretation). The target in motionless movement is explicitly the embodied end of this spectrum. A good kinesthetic display should provide movement cues in a manner the user can understand

pre-reflectively, without consciously attending to the feedback at all [11]. The goal is feedback indistinguishable from intrinsic feedback — the sensory information generated by the user's own movement rather than by an external source [17].

2.2.3 Theoretical Grounding: Sensorimotor Contingencies and Material Experience

The motionless movement approach derives directly from O'Regan and Noë's account of sensorimotor contingencies [10], as applied to tactile material perception. When a finger moves across a surface, tactile impulses occur at a rate proportional to movement speed — slowly when the finger moves slowly, rapidly when it moves fast. This is a sensorimotor contingency: a systematic, predictable relationship between motor activity and resulting sensory cognition. The key insight is that this contingency does not require actual movement to be instantiated. If a system can couple vibrotactile feedback to the user's force application — producing pulses that covary with action attempt in the same way that tactile impulses covary with actual movement — the body's perceptual system may treat the feedback as evidence of movement.

A second theoretical resource concerns the distinction between isometric and isotonic action [17]. Isotonic actions involve joint movement alongside muscle contraction — the arm reaching across a table, the finger sliding across a surface. Isometric actions involve muscle contraction without joint movement — pressing against a fixed object, exerting force without displacement. The prototype exploits a known perceptual asymmetry: when isometric input is supplemented with compelling sensory feedback, users can experience the illusion of isotonic movement. Prior work on isometric input devices has shown that visual feedback alone can induce such illusions through visual dominance [16, 18]. Motionless Movement investigates whether action-coupled vibrotactile feedback can induce

the same illusion through the haptic channel — and whether the resulting experience is more embodied than its visual equivalent.

This is supported by prior work on vibrotactile compliance illusions, in which grain-based haptic feedback triggered at fixed pressure thresholds makes users believe that a rigid surface is deforming under their touch [19–23]. For a compliance illusion to work — for a user to believe the material under their finger is receding — the user must also believe their finger is moving into the space the material is vacating. Compliance illusions are implicitly kinesthetic: they succeed only if the body treats the feedback as evidence of movement. This observation motivates the explicit investigation of whether the same coupling can produce a direct kinesthetic experience — not compliance as a byproduct of a material illusion, but movement as the primary target. The paper frames material experience and kinesthetic experience as two sides of the same coin: the distinction between feeling a material deform and feeling one's hand move into it is a matter of frame of reference, not of underlying mechanism [22, 24].

The project also draws on Schmidt et al.'s distinction in motor control research between intrinsic and extrinsic feedback [17]. Intrinsic feedback is the sensory information generated by one's own movement — the vibration of a tennis racket due to air friction as it is swung. Extrinsic feedback is information about movement provided by an external source — a visual analysis of the racket's swing velocity. The goal is to provide feedback the user experiences as intrinsic: a sensation that feels like a consequence of their own action, rather than a signal to be interpreted. This aligns directly with Ihde's distinction between embodied and hermeneutic mediation [11] — the target is mediation the body absorbs without the mind having to interpret it.

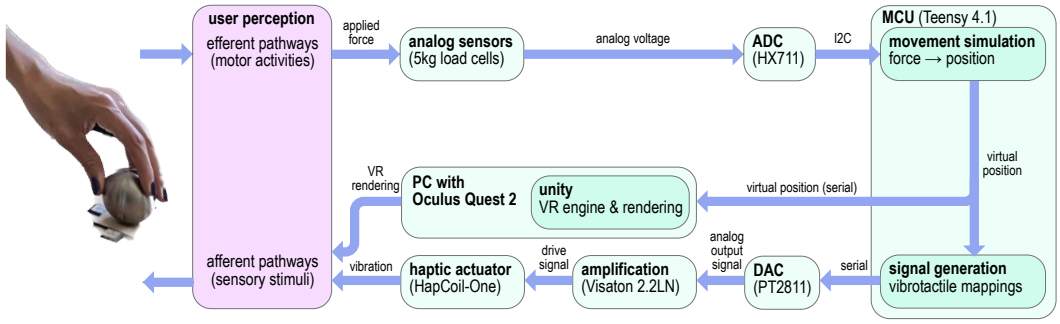


Figure 2.3: System flowchart for the Motionless Movement prototype. User efferent motor activity applies force to the 5 kg load cells (analog sensors), read by the HX711 ADC and passed over I^2C to the Teensy 4.1 microcontroller. The MCU runs the movement simulation (force → virtual position) and the vibrotactile signal generation algorithm in parallel. Virtual position is sent via serial to a Unity VR engine rendering on an Oculus Quest 2 (used in visual-feedback conditions only). The signal generation output passes through a PT8211 DAC, Visaton 2.2LN amplifier, and HapCoil-One vibrotactile actuator to deliver afferent haptic stimuli directly through the spherical handle.

2.2.4 System Implementation

The prototype is a desktop-mounted device with a spherical metal handle (55 mm diameter) mounted on an enclosure housing four 5 kg load cells and a vibrotactile actuator. The hardware pipeline proceeds as follows: load cells detect force applied to the sphere; an HX711 ADC (80 Hz maximum sampling rate) reads the signals; a Teensy 4.1 microcontroller runs the movement simulation and signal generation algorithms; a PT8211 DAC converts the digital signal to analog; a Visaton 2.2LN amplifier drives the HapCoil-One vibrotactile actuator (Actonika, frequency range 10–1000 Hz), which is mounted to directly vibrate the spherical handle. The haptic signal is updated at greater than 200 Hz, faster than the force sampling rate, to maintain signal fidelity within each update window.

For the studies that include visual feedback, the system communicates virtual position to Unity via serial port, which renders a hand avatar pushing the sphere across a textured plane in an Oculus

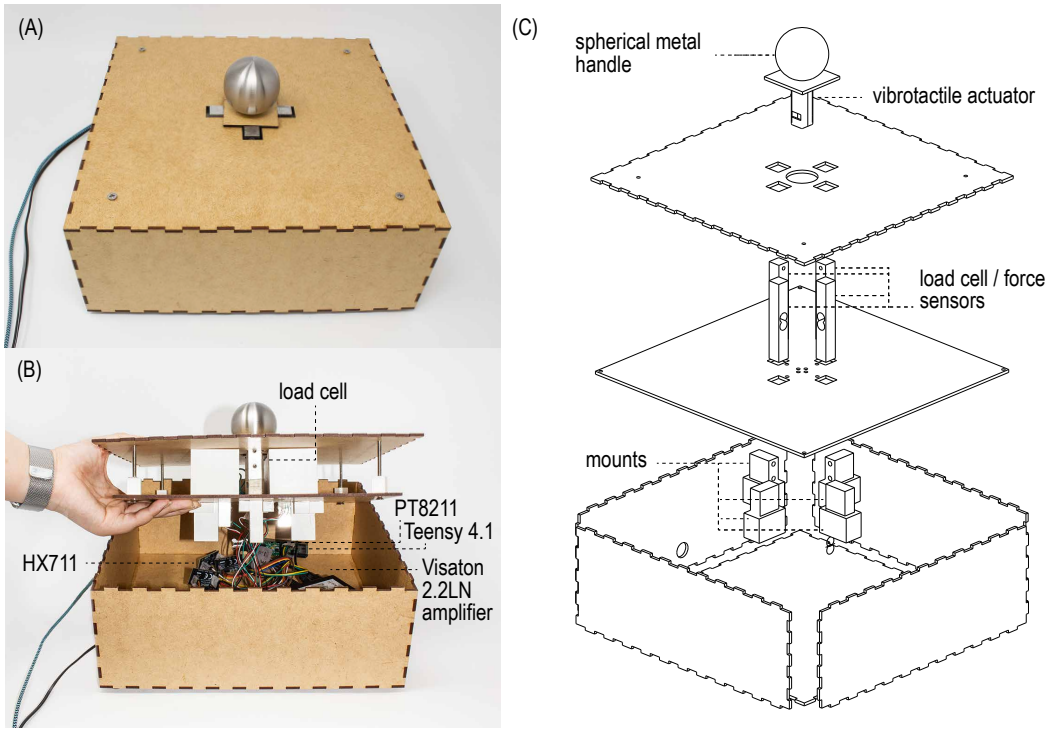


Figure 2.4: Physical prototype of the Motionless Movement vibrotactile kinesthetic display. (A) Exterior view: the spherical metal handle mounted on the laser-cut MDF enclosure. (B) Interior view with hardware components labeled: HX711 ADC, Teensy 4.1 microcontroller, PT8211 DAC, and Visaton 2.2LN amplifier visible inside the enclosure; the HapCoil-One vibrotactile actuator is mounted directly below the handle stem. (C) Exploded engineering diagram showing the assembly layers: spherical handle, vibrotactile actuator, load cell mounting plate, four load cells and mounts, and outer enclosure.

Quest 2. The VR environment serves to increase ecological validity in the visual conditions; it is not present in the haptic-only studies.

Movement simulation model. The core of the system converts the user's applied force into a virtual position. The model assumes a constant mass ($m = 0.5$ kg) and no friction. Force values from the load cells are decomposed into x and y dimensions; acceleration is computed from Newton's second law ($a = F/m$); velocity and displacement are integrated across successive system loops ($\sim$12 ms each). This virtual position drives haptic signal generation and, where applicable, the VR rendering.

Haptic signal mappings. The action-coupled feedback algorithm divides the sensor range into a number of discrete bins (grains). When the measured signal crosses a bin boundary, a vibrotactile pulse is generated. Four mapping strategies were designed and evaluated:

1. *Pressure-based*: grain pulses triggered by instantaneous force magnitude (300 grains, maximum force 10 N). Produces a natural, compliance-like experience, but with a small felt sense of movement.

2. *Distance-based*: grain pulses triggered by virtual position from the movement simulation (400 grains, maximum displacement 0.01 m). The primary candidate mapping — it couples feedback to simulated travel rather than instantaneous force, producing friction-like cues along a simulated movement path [25].

3. *Hybrid*: pressure-based below a threshold of 1.2 N, distance-based above it. Designed to address weak onset and termination cues in the distance-based mapping alone.

4. *Continuous vibration*: vibration sustained while force is applied, with amplitude and frequency

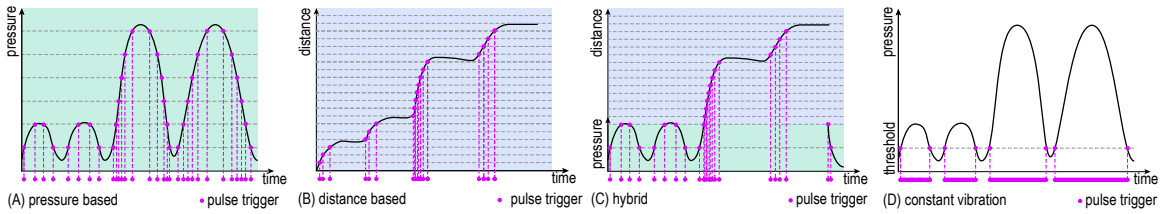


Figure 2.5: Visualization of the four vibrotactile signal mappings as a function of user action over time. The black curve represents the user's action measure (pressure or computed virtual distance); horizontal dotted lines represent grain bin boundaries; each vertical pink tick marks a pulse trigger. (A) *Pressure-based*: pulses triggered when instantaneous force crosses bin thresholds — dense during peak force, absent otherwise. (B) *Distance-based*: pulses triggered by cumulative virtual displacement — evenly distributed along the simulated travel path. (C) *Hybrid*: pressure-based at low force (green region), switching to distance-based above the 1.2 N threshold. (D) *Continuous vibration*: a sustained signal active whenever force exceeds the threshold, serving as the hermeneutic control.

modulated by force magnitude. This serves as the hermeneutic control condition — a symbolic signal rather than an action-coupled one.

Across all conditions, vibration frequency increases logarithmically from 60 Hz to 100 Hz and amplitude from 0.1 to 0.6 as force approaches 10 N.

2.2.5 Evaluation: Four Studies

The evaluation was structured as a progression of four studies, each building on the findings of the previous. Results are reported as 95% confidence intervals on means rather than p-values, consistent with recommendations in the HCI methods literature [26, 27].

Study 1: Movement Perception of Haptic Mappings. Ten participants (9 right-handed, 1 left-handed) compared the four haptic mappings plus a no-vibration control using magnitude estimation. Participants were blindfolded and wore noise-canceling earphones playing white noise to mask audiovisual cues. They rated each condition on two dimensions: estimated magnitude of movement (how far

does the object feel like it can travel?) and naturalness (does this feel like a physical material or a machine vibration?). Conditions were presented in balanced Latin square order with four repetitions per participant.

For naturalness, no-vibration and constant vibration were the two extremes — no-vibration rated most natural (it felt like a real, stationary object), constant vibration rated least natural. Pressure-based, distance-based, and hybrid were statistically similar in naturalness, all perceived as natural material interactions. For magnitude of movement, pressure-based and no-vibration were significantly lower than all other conditions; distance-based, hybrid, and constant vibration were similar. This dissociation is theoretically important: the constant vibration condition achieved high movement magnitude ratings through hermeneutic means — participants interpreted the amplitude-frequency modulation as a symbolic cue for distance — while distance-based and hybrid achieved comparable movement ratings through embodied means, producing experiences participants described in terms of physical materials, friction, and surface texture. The distance-based mapping was selected as the primary condition for subsequent studies because the added complexity of the hybrid algorithm produced no measurable benefit on either dimension.

Study 2: Effect of Haptic Grain Level. Twelve participants performed magnitude estimation comparing three grain levels in the distance-based mapping: 200, 400, and 800 grains. Increasing granularity increased perceived movement magnitude, but with diminishing returns. The difference between 200 and 400 grains was statistically significant (95% CI for the difference: $[-0.61, -1.07]$). The difference between 400 and 800 grains was not (95% CI: $[0.06, -0.36]$). Participant comments illustrated the two failure modes at high granularity: for some, 800 grains felt overwhelmingly convincing; for others, the threshold fell so quickly that the feedback collapsed into what felt like continuous vibration, losing the grain-coupled quality that made lower densities feel like friction.

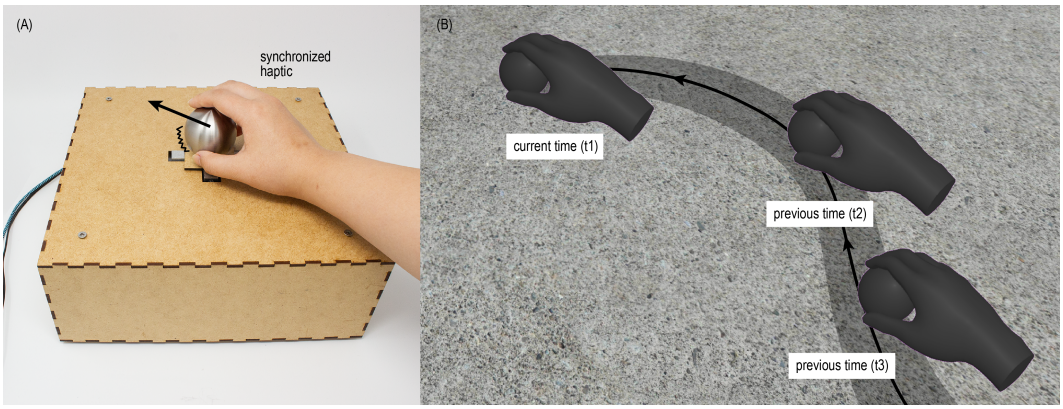


Figure 2.6: Study 3 and 4 setup. (A) Physical prototype during use: the user applies isometric force to the stationary spherical handle while synchronized vibrotactile feedback is delivered. (B) Corresponding VR rendering (Oculus Quest 2 via Unity): the hand avatar and sphere trace the virtual movement path computed from applied force over successive time steps t_1, t_2, t_3. In Study 3, participants adjusted a visual gain parameter until the VR motion matched their haptic percept. In Study 4, the visual and haptic feedback modalities were presented in all four combinations (both, haptics only, visual only, neither) to assess their independent and combined contributions to belief in movement.

Study 3: Haptic Grain and Visual Gain Matching. The same 12 participants adjusted a visual gain parameter — the multiplier on the movement simulation's position output in the VR rendering — until the VR sphere's movement matched their haptic feedback. No clear pattern emerged across participants: matching was mediated by individual mental models of the material being simulated. This theoretically interesting negative result suggests that the relationship between haptic and visual movement cues in multi-modal kinesthetic display is not fixed but depends on context and prior experience. Future work in richer task contexts, where mental models are more constrained, may find more stable correspondences.

Study 4: Effect of Visual Feedback on Belief in Movement. The same 12 participants compared four conditions using the visual-haptic pairings they had selected in Study 3: both haptics and visual, haptics only, visual only, and neither. Participants rated belief in embodied movement — how strongly

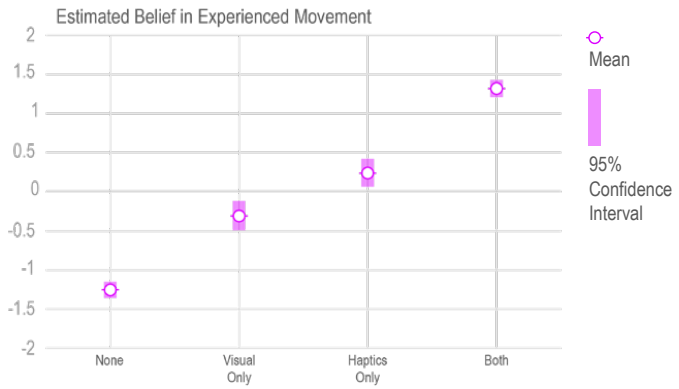


Figure 2.7: Study 4 results: estimated belief in experienced movement across the four feedback conditions (none, visual only, haptics only, both), with means and 95% confidence intervals. The y-axis shows standardized estimates (units: standard deviations relative to the per-participant mean). Haptics only outperforms visual only, establishing that action-coupled vibrotactile feedback is a more effective standalone kinesthetic channel than visual display. The combined condition produces the highest belief in movement overall.

did they feel their hand or the sphere was actually moving, relative to real-world experience?

Belief in movement was highest in the both-haptics-and-visual condition, followed by haptics only, then visual only, then neither. Critically, haptics alone outperformed visuals alone — a striking finding given that visual feedback dominates in most multi-sensory contexts. Participants who found the haptic-only condition more convincing described the visual-only condition as feeling like using a computer mouse: they were translating their movement into something else, rather than experiencing the movement directly [16, 18]. The visual rendering produced a hermeneutic experience; the haptic feedback produced an embodied one. The combination produced the strongest experience because the haptic grounding provided the embodied substrate on top of which the visual confirmation could reinforce rather than replace.

2.3 Surface I/O: Encoding Affordances in Material Form

2.3.1 The Problem: Interface Without Explanation

As computing devices have proliferated in everyday life, interaction with them has remained constrained to two dominant interface types: seamless, continuous surfaces — most often flat, with integrated touch sensing — and mechanical inputs, typified by buttons, sliders, and dials. Touchscreens offer a reconfigurable, robust platform that designers can tweak and test, but being entirely flat, they lack tangibility and rely on visual attention to operate. Mechanical controls offer rich tactile affordances and direct interaction methods that have stood the test of time, but they are difficult to design, expensive to implement at scale, and add mechanical complexity and failure modes that lead some manufacturers to remove them altogether — the removal of the home button from smartphones and the replacement of instrument clusters with flat touchscreens in the Tesla Model 3 being the most visible examples. The costs of this resolution are real: drivers reaching for a volume control while watching the road cannot locate a flat surface by touch; VR users trying to adjust a headset parameter without removing the headset have no tactile reference point to orient them; users with visual impairments navigating an appliance interface designed for sighted users encounter an entirely symbolic environment they cannot read through touch.

Surface I/O proposes an intermediary approach: instead of choosing between mechanical complexity and tactile impoverishment, it functionalizes the exterior surface of the device itself. The hypothesis is that surface geometry, designed with intent across multiple physical scales, can provide haptic affordances without dedicated mechanical components while simultaneously enabling input tracking — no moving parts, no added electronics beyond an inexpensive contact sensor. A surface designed

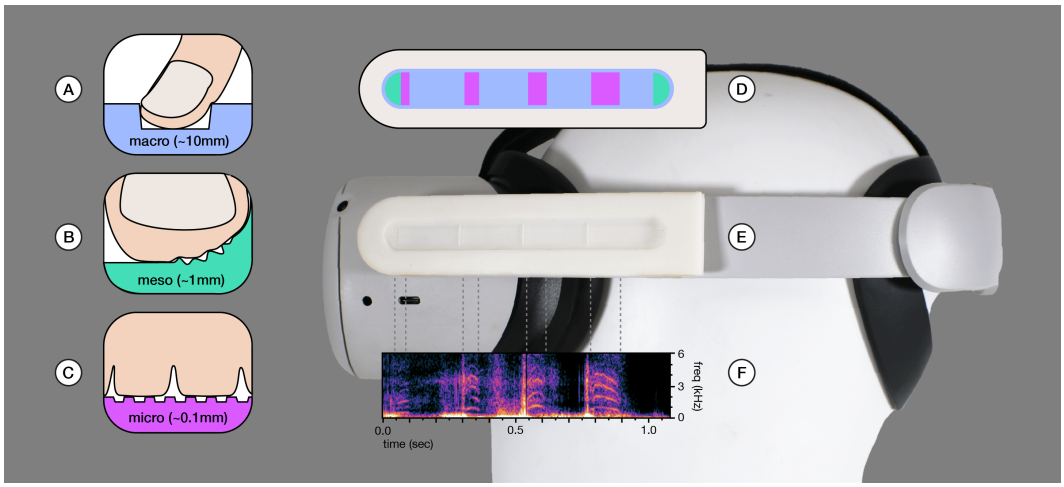


Figure 2.8: Surface I/O system overview. Left column (A–C): cross-sections illustrating the three feature scales as they engage the fingertip — macro-scale ($\sim$10 mm) creating static skin deformation, meso-scale ($\sim$1 mm) producing texture sensation during lateral scanning, and micro-scale ($\sim$0.1 mm) generating a distinctive vibro-acoustic spectrum. Right: the system instantiated as a one-piece plastic widget (D, schematic; E, photo) installed on the Elite Head Strap of an Oculus Quest 2. The widget combines a macro-scale valley (blue, for lateral thumb guidance), meso-scale detents and rough-texture end stops (teal), and micro-scale ridge arrays (pink) for absolute position tracking. An inexpensive piezoelectric vibration sensor affixed to the widget's underside (F, spectrogram shown) captures the vibroacoustic signal for input classification.

well enough should orient the hand, signal discrete states, mark boundaries, and track position, all through feel alone. The paper frames this as a contribution toward "invisible ubiquitous interactivity at scale" [3,28–31]: interfaces so naturally integrated into the form of an object that interactivity becomes a property of the surface rather than a feature added to it.

2.3.2 Multi-Scale Architecture: Three Registers of Surface Information

Surface I/O's central contribution is the identification of three physically distinct scales of surface geometry, each of which encodes a different kind of interaction information and engages a different aspect of the human tactile perceptual system. These scales are not arbitrary divisions of a continuum;

they correspond to established distinctions in tactile psychophysics research concerning the mechanoreceptor populations that mediate surface perception at different spatial frequencies [32–34].

Macro-scale features (5 cm to 1 mm). At this scale, surface geometry creates discernible static deformation across the fingerpad and surrounding skin. The relevant mechanoreceptors — SA-I, FA-I, and SA-II units — respond to skin deformation and stretch, preserving spatial information about the shape encountered. Macro-scale features encode two kinds of information. The first is location: a finger-sized hole, bump, or ring captures the finger and tells it where to rest. Users immediately recognized these features as button-like — points of interest that invite pressing, dwelling, or fine rotary manipulation. The second is direction of travel: valleys (linear indentations that capture the finger and permit lateral movement), rails (raised ridges that resist perpendicular motion while permitting parallel translation), and asymmetric sawtooth-like features that resist movement in one direction more than the other all communicate a path before the finger has consciously processed it. The valley design received the highest intuitiveness ratings in Surface I/O's user studies, and 87.5% of participants spontaneously reported that it communicated horizontal movement. The geometry does not describe the motion. It makes one direction feel like the natural continuation of where the finger already is.

Meso-scale features (1 mm to 200 μm). At this scale, features are too small to lock-in or redirect the finger but large enough to be felt during lateral scanning — the exploratory procedure by which humans assess texture when moving the hand across a surface. These features engage primarily FA-I and FA-II mechanoreceptors, sensitive to vibrotactile signals in the range produced by finger movement across repeated micro-structures. Meso-scale textures in Surface I/O serve as semantic markers: a rough strip signals that the finger has reached an extreme position; a smooth-to-rough transition communicates that a boundary has been crossed; a subtle directional texture (fine parallel

lines) produces different resistance in different directions, providing a guidance cue without the strong topographic commitment of a macro-scale valley. These features do not constrain the finger. They inform it — the tactile equivalent of a visual landmark.

Micro-scale features (below 200 μm). At this scale, features cannot be felt by a static finger and are barely detectable even during slow scanning — they exist below the threshold of direct tactile perception [35, 36]. Their function is not to communicate affordances but to enable input sensing. When a finger slides across a surface patterned with micro-scale ridges, it produces a distinctive vibro-acoustic spectrum whose frequency content depends on the pitch of the ridges and the speed of the swipe [37, 38]. A small piezoelectric sensor affixed to the underside of the Surface I/O object can capture this spectrum, and a machine learning classifier can infer the user's position, direction, and speed from the signal. By operating below the threshold of human tactile perception, micro-scale textures encode sensing information without competing with the affordance information encoded at larger scales. The surface communicates what to do to the finger; separately and silently, it reads what the finger is doing.

The three scales work together as a system. A Surface I/O widget installed on the side of a VR headset might use a macro-scale valley to guide the thumb along a horizontal axis, meso-scale rough patches at the ends of the valley to signal that the travel limit has been reached, and micro-scale ridges of varying pitch between pairs of tactile detents to allow the system to infer absolute position within the slider's range. No screen is required. No label is needed. The thumb finds the valley, follows it, feels the detents mark discrete states, feels the rough patch signal the stop. All of this happens below the threshold of deliberate reasoning.



Figure 2.9: The 35-stimulus evaluation set for Surface I/O, organized by scale and feature type. Top row (A–E): five meso-scale textural primitives (smooth, velvet, rough, grippy, subtle guide). Rows 2–7: 30 macro-scale features spanning point-of-interest locators (large hole, small hole, large bump, small bump, large ring, small ring, barrier), discrete state markers (detent), linear guidance features (valley, single rail, multiple rails, one-way, valley variants with detents and hills), rotational features (circular valley, circular rail, circular rails, spiral rails), switch-type features (valley switch, smooth valley switch, rail switch, smooth rail switch), and compound features. Each stimulus is centered on an 8×8 cm panel and 3D-printed. Brown overlines indicate stimuli used in Study 1; blue overlines indicate stimuli used in Study 2.

2.3.3 Implementation

Surface I/O prototypes were fabricated primarily using a masked stereolithography (MSLA) 3D printer (ELEGOO Mars 3, approximately $250 USD) capable of producing features as fine as 35 μm. A custom laser cutter capable of 20 μm resolution was used for the finest micro-scale textures. The authors confirmed that Surface I/O is compatible with mass-production methods — specifically injection molding and stamping — at all three scales, because comparable feature sizes are already routinely produced in holographic injection molding and vinyl record stamping. The primary added electronic component is a small piezoelectric disc affixed to the underside of the widget; existing internal microphones in devices such as VR headsets might capture the necessary vibroacoustic signal without any additional hardware.

For input sensing, Surface I/O uses a 7 mm diameter piezoelectric element connected to a custom pre-amplifier, digitized at 192 kHz. Classification uses an ExtraTrees machine learning model (100 estimators) operating on a 246-dimensional feature vector combining FFT features and Mel-frequency cepstral coefficients, with a five-frame voting window to stabilize output. The pipeline was evaluated on four exemplary micro-scale textures.

The representative stimuli set developed for evaluation comprised 35 distinct surface features organized across the three scales: 5 meso-scale textural primitives, 8 macro-scale point-of-interest features (holes, bumps, rings of varying sizes), 4 macro-scale guidance features (valleys, rails, one-way asymmetric textures, detents), and 18 more complex widget-level stimuli combining primitives into functional interface elements. These were presented to participants across four studies.

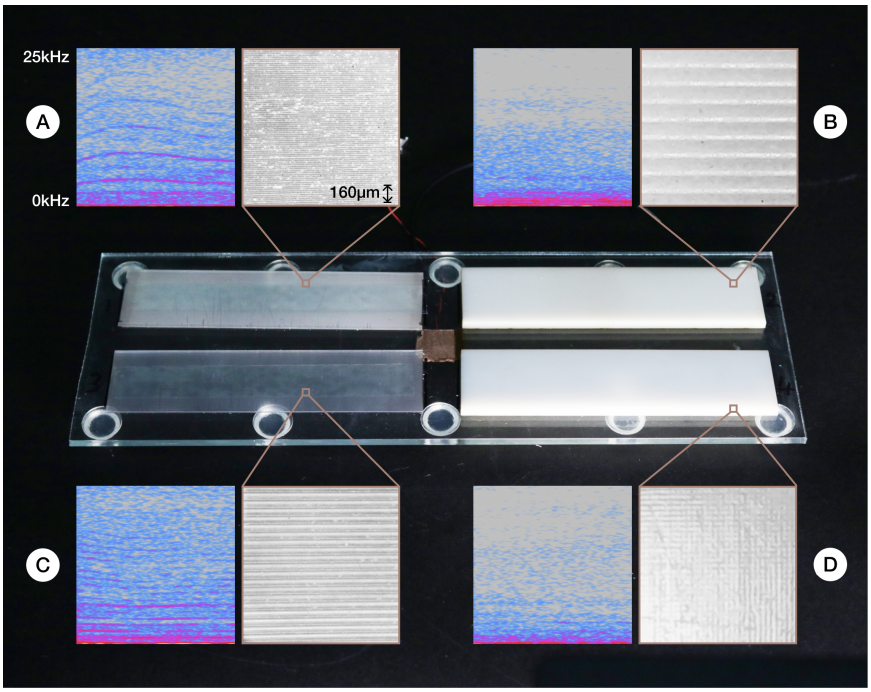


Figure 2.10: Four micro-scale texture samples (A–D) mounted on an acrylic evaluation plate with a copper-shielded piezoelectric element affixed to the underside. Each panel shows a microscopic view of the periodic ridge structure (right inset, common scale; ridge pitch visible at $\sim$160 μm for texture A) alongside an example spectrogram captured during a finger swipe (left inset, 0–25 kHz). The distinctive harmonic patterns in each spectrogram — determined by ridge pitch and swipe speed — are the basis for the ExtraTrees classifier's position and texture inference. Mean classification accuracy across the four textures: 90.1%.

2.3.4 Evaluation

Study 1: Efficacy of Primitives. Nine participants explored 17 surface primitives in a semi-structured interview study. A box prevented visual access; participants used the index finger of their dominant hand. They rated each stimulus on tactile intensity, pleasantness, smoothness, stickiness, bumpiness, sharpness, continuity, and directionality (five-point Likert scales), and provided open-ended descriptions of what they felt and what they inferred the designer was communicating.

Three findings from Study 1 are worth stating precisely. First, the more tactilely intense a stimulus, the less pleasant users found it ($R^2 = 0.558$), with sharpness being the strongest predictor of unpleasantness ($R^2 = 0.595$). This is a design constraint: affordance information should be encoded in features intense enough to be legible but not so sharp as to produce aversion. Second, macro-scale guidance features — particularly the valley — received the highest directionality ratings: 87.5% of participants reported it communicated horizontal movement without prompting. Third, meso-scale features that conveyed no directional information were perceived as neutral background textures, confirming that meso-scale features can serve as semantic marks without being misread as affordance signals.

Study 2: Functional Associations. Eight participants (all Study 1 veterans) explored 18 thematic sets of widget-level stimuli. Each set isolated a specific design variable — size of locating features, detents versus hills, linear versus rotational geometry, number of rails — and participants described what each set might control on a computer interface, then rated each stimulus for intuitiveness and complexity.

Key findings: larger locating features (finger-sized holes and bumps) were consistently rated more intuitive than smaller ones and universally associated with high-frequency interactions such as

power buttons and touch ID; smaller features were perceived as indicators for infrequent or cautious interactions. Detents were perceived as discrete controls suggesting modal switching; hills (sinusoidal rather than stepped transitions) were perceived as continuous controls. Linear stimuli were associated with sliders and continuous value change; circular and spiral features with dials and mode selection. These findings establish that Surface I/O features not only convey direction and location but carry functional semantics — users make reliable inferences about what a widget does from how it feels, without labels or explanation.

Study 3: Input Accuracy. Ten participants each performed 30 swipes across each of four micro-scale textures (1,200 trials total). Using leave-one-participant-out cross-validation, the machine learning classifier achieved a mean texture classification accuracy of 90.1%. While the authors acknowledge this falls short of the 99%+ accuracy consumers would demand from a production system, they position it as a proof of concept establishing feasibility, with substantial headroom for improvement through more sophisticated classification pipelines.

Study 4: Durability. Seven materials were subjected to 120,000 reciprocating cycles (240,000 swipes) using artificial finger probes constructed per ASTM F1597-02 standard [**?**]. No visible abrasion was observed in any material across all test intervals; all materials preserved identifiable harmonic spectra. Notably, the soft primer coating on the artificial finger probes required replacement every 40,000 cycles — the simulated skin wore faster than the surface. Micro-scale Surface I/O features, at the very scale most vulnerable to wear, proved more durable than the simulated skin pressing against them.



Figure 2.11: Example application of Surface I/O. The left three images (A – C) show the top layers of an Alexa Echo. The right two images (D – E) show our single-piece, Surface I/O alternative top plate.

2.3.5 Three Example Interfaces

Surface I/O's generality is illustrated through three worked examples. The first augments a VR headset with a linear slider on the Elite Head Strap — combining a macro-scale valley to guide thumb travel, meso-scale rough patches at both ends to signal limits, and varying ratios of two micro-scale textures between detents to enable absolute position tracking. The second augments an automobile steering wheel with a four-way directional control and a small spiral dial for continuous value adjustment, enabling mode navigation and parameter control without visual attention. The third redesigns the top plate of an Amazon Echo as a single-piece Surface I/O widget — replacing four spring-loaded mechanical buttons with a repeated-rail dial for volume and a toggle widget built from rails with stops, using micro-scale texture transitions for direction sensing. Across all three, the same design principle holds: every meaningful interaction state or boundary is communicated through felt geometry before any electronic feedback is required.



Figure 2.12: Example application of Surface I/O. Photo of our Surface I/O-augmented steering wheel (B), with closeup views of a four-way directional control made out of four tear-drop finger wells (A) and small spiral dial (B) where a thumb could be used to manipulate a continuous value, such as volume or temperature.



Figure 2.13: The Surface I/O VR headset slider installed on the Elite Head Strap of an Oculus Quest 2, shown in use. The thumb finds the macro-scale valley without visual guidance, follows it laterally, registers the meso-scale detents as discrete states, and stops at the meso-scale rough-patch end limit. Micro-scale ridge arrays between detent positions allow the piezoelectric sensor to infer absolute thumb position along the slider's range. No screen, label, or electronic feedback is required for the affordance; only the piezoelectric sensor is added electronically.

2.4 Cross-Project Synthesis: What Physical Mediation Requires

2.4.1 The Shared Logic

Motionless Movement and Surface I/O were developed by different research groups, address different perceptual channels, deploy different technical approaches, and draw on different literatures within HCI. What they share is a commitment to the same design target — mediation that operates at the pre-reflective level — and the same discovery about what achieving that target requires.

In Motionless Movement, the design question is: can vibrotactile feedback produce kinesthetic experience without movement? The answer is yes, but only if the feedback is coupled to action attempt in a way that preserves the sensorimotor contingency of tactile movement cues. Continuous vibration achieves high movement magnitude ratings hermeneutically — participants interpret the amplitude modulation as a symbolic cue for distance — but it does not feel like movement; it feels like operating a machine (to quote one participant, "it feels like using a mouse"). Distance-based grain coupling succeeds because the feedback changes as the user's applied force changes virtual position, and the body's perceptual system treats this covariance as evidence that an object is moving. The feedback does not say *you are moving*. It produces the sensory pattern that accompanies movement, and the body draws the inference automatically.

In Surface I/O, the design question is: can the geometry of a surface communicate what to do without labels, instructions, or visual feedback? The answer is yes, but only if the geometry exploits the sensorimotor system's existing competences, matching how users naturally perform haptic explorations. A valley captures a finger because the finger naturally settles into the lowest-energy path. A detent registers as a discrete state boundary because the sensorimotor system treats a sudden

transition in resistance as categorically significant. A rough patch signals a limit because the hand associates texture discontinuities with zone boundaries in any exploratory task. The geometry does not explain. It exploits what the hand already knows about the physical world.

Both systems create mediation by encoding the consequences of action into the interface rather than describing the state of the system to the user. A Surface I/O valley does not work by trying to indicate the fact that lateral movement is possible — it directly provides the path, structurally, haptically, and in the body's experience. A Motionless Movement kinesthetic display does not communicate movement by somehow showing the virtual sphere is moving — it recreates the sensorimotor contingency of movement, making the body experience movement as the natural consequence of what it is doing. In both cases, the gap between what the user is trying to do and what the system provides closes not because the system explains itself better but because the system has become, in Ihde's terms [11], quasi-transparent — felt through rather than looked at.

2.4.2 Two Requirements for Pre-Reflective Mediation

Taken together, these projects establish two requirements for mediation to operate at the pre-reflective level. These requirements will recur in the DEXAS framework developed in Chapter 4, where they appear as the first two principles. Stated here in the context of the physical layer, before the framework is generalized:

Requirement 1: Encoding of affordances into the representational structure of the interface. Pre-reflective mediation requires that the interface encode what actions are possible and where — as structures that the sensorimotor system can read directly, instead of learned or textual descriptions. In Surface I/O, this encoding happens through macro-scale geometry: the valley encodes lateral travel,

the detent encodes discrete state boundaries, the rough patch encodes zone limits. What makes this encoding pre-reflective rather than merely intuitive is that it is complete at the point of contact. A finger that finds a valley has already begun moving laterally — the geometry has committed the body to a path before the mind has had occasion to consider it. There is no moment of inference, no gap between perceiving the affordance and acting on it. This is what distinguishes the valley from a label reading "slide here": the label addresses the mind and waits for a response; the valley addresses the hand and receives one immediately.

Requirement 2: Coupling of feedback to action attempt in a way that preserves sensorimotor contingency. Pre-reflective mediation requires that the system's responses to user action arrive in a form the body treats as consequences of that action rather than as signals about the system's state. In Motionless Movement, this coupling is implemented through the grain-to-distance mapping: as force increases, virtual position increases, and the density of vibrotactile pulses increases proportionally — exactly as tactile signal density would increase if the finger were moving across a real surface. The feedback does not report that the sphere has moved X millimeters. Instead, it instantiates the contingent sensorimotor pattern that would accompany moving the finger X millimeters across a friction surface, and the body reads that pattern as movement. This is why motion-coupled haptics outperforms visuals as a standalone kinesthetic channel: visuals provide information about the state of an external object; haptics — when properly coupled — enters the body's own sensorimotor loop.

These two requirements define a design space for pre-reflective mediation. A system satisfying both can communicate with the body directly — passing action possibilities and movement experiences through channels that bypass the cognitive bottleneck. A system satisfying only the first (encodes affordances, provides symbolic feedback) will be understandable but not embodied: users understand what to do but will not feel it. A system satisfying only the second (provides coupled feedback, does

not encode affordances) will be embodied in its response but uninformative about possibility: users feel what they are doing but may not know what else is possible. Both requirements are necessary.

2.4.3 The Boundary This Work Defines

It is worth being precise about what these projects establish and what they do not. They establish that pre-reflective mediation is achievable at the physical layer — that bodies can read affordances from geometry and receive kinesthetic information from vibrotactile coupling — and they specify the conditions under which this is possible. They do not establish that pre-reflective mediation is sufficient for all forms of human-machine interaction.

Physical mediation operates under a structural constraint: it works when the system's role is passive — encoding, reflecting, responding to the user's action. It breaks down when the system must act autonomously, make decisions, or manage a process the user needs to understand and shape. An autonomous pneumatic system that harvests renewable energy and actuates without valves or electronics does not fit within the pre-reflective register at all — not because it is too complex to be made legible, but because legibility in that context requires exposing something about internal state and temporal logic that geometry alone cannot carry. The user does not just need to know where to put their finger. They need to model what the system will do next and why.

This is the boundary the remaining projects cross. Sustainflatable and In-Situ Presenter operate above the threshold of conscious reasoning — in the register where the mediating system must make its logic explicit, not merely encode its affordances. Chapter 3 examines what mediation requires at that level: instead of designing for physical geometry and actuation patterns that speak to the sensorimotor system, we look at representations that speak to the mind without overwhelming it, at

the right abstraction level.

2.5 DEXAS Principles Established in This Chapter

The empirical record of this chapter contributes directly to two of the five DEXAS principles developed in Chapter 4. I preview them here not to anticipate the full framework but to make explicit what Chapter 2 contributes to the dissertation's argument.

Embodied feedback coupling (E). Motionless Movement demonstrates that feedback can be designed to enter the user's sensorimotor loop directly rather than arriving as a signal that requires conscious interpretation. The specific mechanism is action-coupling: the feedback varies as the user's action attempt varies, in a way that preserves the sensorimotor contingency of real movement. The result — haptic feedback that outperforms visual feedback as a kinesthetic channel — establishes that the channel through which mediation operates determines its effectiveness independently of the information content being transmitted. Haptic coupling and visual display can carry the same semantic content and produce fundamentally different experiences, because one enters the body's action loop and the other requires the mind's interpretation loop. This distinction — coupling versus reporting — is the empirical content of the E principle in DEXAS.

Structured signal decoding (D). Surface I/O demonstrates that raw sensory signals — the vibro-acoustic spectra produced by finger movement across micro-patterned surfaces — can be transformed into structured, actionable information (position, direction, speed) through appropriately designed surface architecture and machine learning classification. The multi-scale system encodes signals such that they are decodable at multiple levels simultaneously: the macro-scale speaks to the sensorimotor system about affordance; the meso-scale speaks about boundary and zone; the micro-scale speaks to

the sensing system about input. This layered decoding is an instance of the general DEXAS principle that effective mediation transforms raw complexity into structured representations at the appropriate level of abstraction.

These two principles share a common implication that the next chapter will complicate: they both describe mediation that can be accomplished without the system exposing its internal logic. The valley does not tell the user how the surface was designed. The vibrotactile display does not explain its mapping algorithm. The mediation is effective precisely because it makes those details irrelevant — the body does not need to know them. This is the ideal case. It is also the edge case. Once the system must act autonomously — once there are decisions being made that the user needs to understand, correct, or build on — this opacity becomes a liability rather than a feature. The next chapter examines what happens at that transition.

Chapter 3: Mediation at the Threshold of Autonomy

Chapter 2 established the conditions under which mediation can operate below the threshold of conscious interpretation. Motionless Movement and Surface I/O each showed that when a system encodes affordances into its representational structure and couples its feedback to user action in ways that preserve sensorimotor contingency, the body can absorb that mediation without deliberate interpretation. The two DEXAS principles those projects instantiate — embodied feedback coupling and structured signal decoding — describe mediation that succeeds by bypassing the cognitive layer. The interface becomes quasi-transparent in Ihde's sense: felt through, not interpreted.

This mode of mediation has a structural limit. It works when the system is passive: encoding, reflecting, responding to what the user does. The valley guides the finger because the finger is already moving; the vibrotactile display couples its output to the user's force because the user is already applying force. The mediation is reactive. The system has no agenda of its own. There is no gap between user action and system response that would require the user to model the system's internal logic.

The two projects examined in this chapter operate differently. Sustainflatable (UIST '23) [39] is a fully autonomous pneumatic system that harvests renewable ambient energy and actuates without valves, electronics, or real-time human input. In-Situ Presenter (UIST '26, in submission) infers what a presenter means to communicate and dynamically repositions her video feed during a live talk. Both systems act. They have their own sense-reason-act loops, generate behavior independently of direct user instruction, and present the user with outcomes rather than surfaces. Neither can be mediated at the pre-reflective level, because what the user needs to understand and shape is not a surface but a process: the sequence of decisions the system makes on its own.

The shift is not merely technical. In the physical layer, mediation meant making the interface

legible to the body. Here, it means making the system's reasoning legible to the mind — exposing the logic by which the system acts in a form that a person can inspect, evaluate, and intervene on. The transparency required is not the transparency of a well-designed surface, which in the phenomenological sense disappears into use. It is the transparency of a process: the system must show its work.

Section 3.1 provides the theoretical grounding for this shift, connecting the problem of autonomous and intent-driven systems to Ihde's hermeneutic mode of mediation and to the human-automation literature on explainability, predictability, and controllability. Section 3.2 presents Sustainflatable and argues that its design tool exemplifies the core requirement: autonomous systems need human-facing reasoning interfaces as an architectural precondition for deployment. Section 3.3 presents In-Situ Presenter, including the ITalk-57 dataset, the system's unified semantic architecture, and the evaluation results read through the dissertation's frame. Section 3.4 draws out what the two projects share and where they differ, locating the precise boundary between pre-reflective and cognitive-layer mediation. Section 3.5 maps the chapter's findings onto two new DEXAS principles: exposed reasoning transparency and adaptive input unification.

3.1 The Cognitive Layer: Theoretical Grounding

The theoretical resources that anchored Chapter 2 — sensorimotor contingencies as the basis for pre-reflective perception, Ihde's distinction between embodied and hermeneutic mediation, the problem of kinesthesia as a displayless modality — remain relevant here, but their point of application has shifted. In Chapter 2, the goal was to push mediation toward the embodied pole: to design systems the body could absorb without conscious interpretation. That was achievable because those systems were passive. They encoded and responded. They did not decide.

When systems begin to decide — when they have autonomous behavior, inferential architectures, or intent models that generate actions independently of real-time human instruction — embodied mediation is no longer available as a primary strategy. A person cannot feel pre-reflectively what a pneumatic valve will do next cycle. She cannot sense which slide element a language model has aligned her script segment to. The system's reasoning is not a surface property the sensorimotor system can read; it is a process unfolding over time, often in ways not observable from the system's external behavior alone.

What becomes necessary is what Ihde calls hermeneutic mediation: the technology presents a representation that must be read and interpreted [11]. A thermometer does not make you feel cold — it gives you a number you must convert. Hermeneutic mediation is more cognitively demanding than embodied mediation, but in many contexts it is the only form available. The question is not whether to require interpretation, but how much, of what kind, and at what level of abstraction.

The literature on human-automation interaction has approached this question from the side of system design, and it has converged on four related concepts: legibility, predictability, transparency, and controllability. Legibility refers to whether a system's current state and recent actions are interpretable by a human observer. Predictability refers to whether its future actions can be anticipated from that observation. Transparency refers to whether the system's internal reasoning — the process by which it moves from inputs to outputs — is accessible for inspection. Controllability refers to whether a human can intervene in that process meaningfully [40, 41].

These concepts have been studied across aviation automation, autonomous vehicles, and AI-assisted clinical decision support. In each domain the finding is consistent: when systems fail, the proximate cause is often opacity rather than technical malfunction. The operator cannot tell what the system is doing, cannot predict what it will do, and therefore cannot intervene before the failure

cascades [42, 43]. The lesson is not that automation is dangerous but that opacity is. A system that is technically correct but epistemically inaccessible to the people responsible for it is, practically speaking, unusable.

The projects in this chapter confirm this at a finer grain. Sustainflatable was technically functional before its design tool existed — the pneumatic physics worked, the valves opened and closed at appropriate thresholds. What was missing was any way for the designer to reason about what the system would actually do: how pressure would build across multiple environmental cycles, when actuation would trigger, how sensitive the behavior was to variations in ambient conditions. Without simulation in the design tool, the physics remained opaque — a designer could assemble a correct configuration and still have no way to predict when it would actuate, under what conditions, or how sensitive that behavior was to her design choices.

In-Situ Presenter confronts the problem in an inverse pathway. Where Sustainflatable needed to expose its reasoning outward, In-Situ Presenter needed to read the presenter's intent inward — to build an internal representation of what the presenter means to do that is rich enough to act on. The mediating representation is not an output for the human to inspect. It is an internal architecture: a semantic map between script segments and slide elements, constructed in preprocessing and queried in real time, that bridges the presenter's embodied, distributed signaling (speech, gesture, slide structure) and the system's need to compute a placement decision. The transparency here is partial and asymmetric — the presenter sees the system's behavior, not its reasoning. But the system's reasoning must be reliable enough, and the behavioral indicators visible enough, that the presenter can develop an adequate mental model and intervene when it breaks down.

These two cases establish the theoretical terrain for the chapter: the requirement for exposed reasoning transparency, and the requirement for adaptive input unification, as conditions under which

mediation at the cognitive layer becomes effective rather than merely present.

3.2 Sustainflatable: When Autonomous Systems Must Show Their Mechanism

3.2.1 The Problem: Making Electronic-Free Autonomous Actuation Systems Explainable

Sustainflatable began as an investigation into a different problem: whether ambient environmental energy — thermal fluctuation, wet-dry cycles, wind, water flow — could power and control pneumatic interfaces without electronics. The motivation was both technical and environmental [44]. Most pneumatic HCI systems rely on electric pumps and electric valves, creating energy inefficiencies, material complexities, and dependence on electronic infrastructure that limits deployment in outdoor, unattended, and resource-constrained contexts [45–60]. The Sustainflatable system proposed a three-phase alternative: non-electric pumps that harvest ambient energy to compress air, storage units with distinct volume-pressure characteristics that accumulate that air, and stimuli-responsive valves triggered by environmental factors — temperature, moisture, pressure differential — that release it autonomously at appropriate times [61–63].

The system's physical mechanisms span three pump types — thermal, moisture-driven, and kinetic — and a corresponding set of stimuli-responsive valves triggered by temperature, humidity, and accumulated pressure. Through combinations of pump type, storage compliance, and valve logic, the system can execute complex temporal behaviors: releasing compressed air only when conditions meet multiple simultaneous thresholds, accumulating pressure across many environmental cycles before triggering a burst, or tracking a slowly changing environmental variable. A Sustainflatable system placed in a garden can harvest wind energy over weeks, accumulate it as compressed air, and release it at a precisely conditioned moment — protecting plants from cold snaps, dispersing seeds when

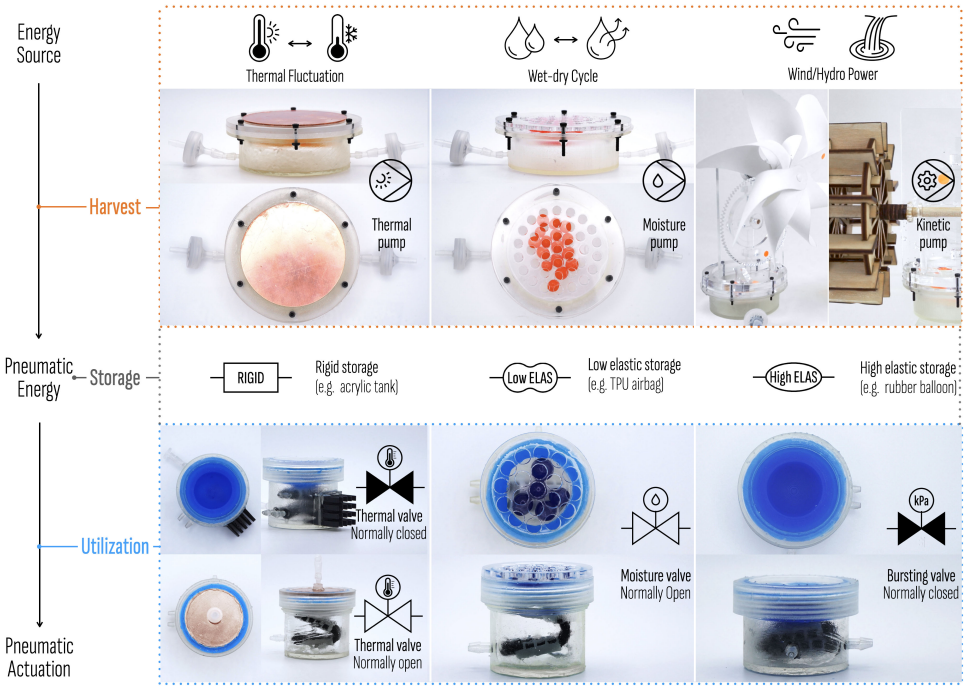


Figure 3.1: The three-phase energy harnessing architecture of Sustainflatable. Top: three ambient energy sources — thermal fluctuation, wet-dry cycle, and wind/hydro power. Middle: the three corresponding non-electric pump types (thermal, moisture, kinetic) and three storage unit variants (rigid, low-elastic, high-elastic). Bottom: five stimuli-responsive valve types triggered by temperature, moisture, or accumulated pressure [64–66]. Complex autonomous actuation behaviors emerge from combinations of these components, with no electronics required.

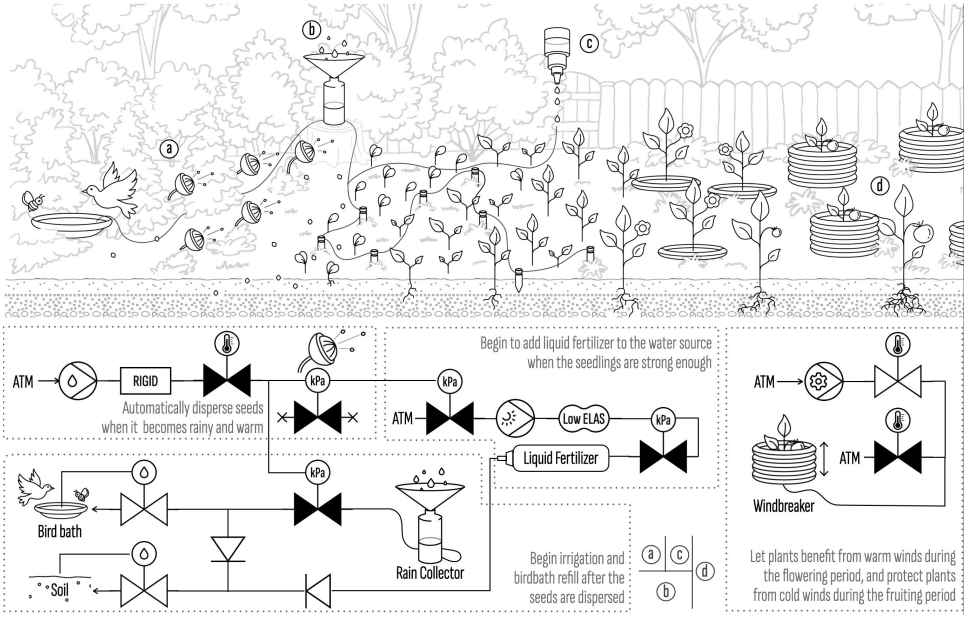


Figure 3.2: Four garden automation applications of Sustainflatable illustrating the system's ability to compose pump, storage, and valve components into complex conditional behaviors. (a) Automatic seed ejection triggered when conditions become warm and rainy. (b) Subsequent self-feedback irrigation and bird-bath refill. (c) Appropriately timed fertilization when seedlings are strong enough. (d) Self-regulating plant insulation toggled between flowering and fruiting periods. Each application is realised through a distinct combination of component types without any electronics.

conditions become favorable, or triggering irrigation when soil moisture drops.

The problem the researchers encountered was epistemic rather than physical. Without a tool that allowed designers to understand the system's temporal dynamics — how pressure would build over multiple environmental cycles, how different storage compliance curves would affect actuation timing, how valve thresholds would interact with pump output under realistic environmental conditions — the system could not be reliably designed or deployed. A designer who assembled a physically correct configuration might still not know when it would actuate, at what pressure, or under what conditions. The system's behavior was fully determined by its physical parameters, but those parameters were too

numerous, too interdependent, and too sensitive to environmental variability to reason about directly. The gap between the system's internal physics and the designer's need to understand and configure its behavior was where a reasoning layer became necessary.

3.2.2 The Design Tool as Reasoning Layer

The Sustainflatable design tool is, at one level, a simulation environment: a Rhino Grasshopper platform in which the designer configures a template system — pump, valve, storage, valve in series — and generates a preview of its behavior across specified environmental cycles. More precisely, it is an intermediate representation sitting between the system's physical behavior and the designer's need to understand, configure, and deploy it.

The tool's architecture addresses three requirements for mediation at this level. First, it decodes raw physical complexity into structured, legible representations. The system's behavior is determined by interacting differential equations — gas laws, pump models, valve hysteresis characteristics — that are technically correct but cognitively opaque. The design tool transforms these into visual representations: three-dimensional previews of pneumatic component states, transparency-coded pressure indicators, temporal plots of pump pressure and storage fill across environmental cycles. The designer reads the representations the equations generate, at a level of abstraction her judgment can work with.

Second, the tool exposes the system's reasoning by making its internal logic explicit and inspectable. The simulation algorithm — a quasi-static iterative model that advances system state at each time step based on current environmental factors and previous system status — is not hidden. Its parameters are tunable. The designer can modify pump type, storage volume, or valve threshold and immediately observe the effect on predicted behavior [64–66]. The tool represents not just what the system will

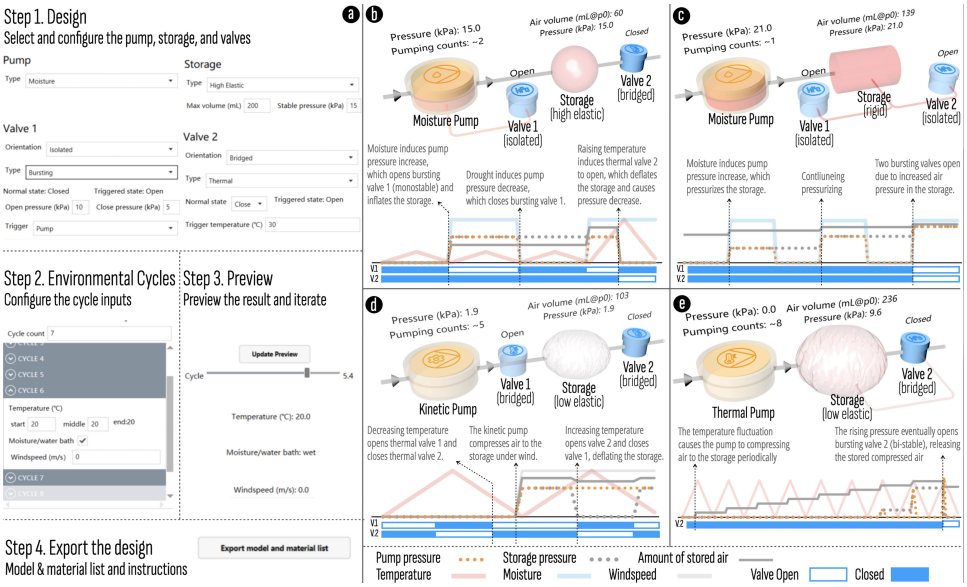


Figure 3.3: The Sustainflatable design tool interface (Rhino Grasshopper). Left panel: Step 1 (component configuration — pump type, storage type, valve parameters) and Step 2 (environmental cycle specification). Center and right panels: Step 3 behavioral previews for five distinct configurations (b–e). Each preview shows three-dimensional component-state models (volume changes indicated by geometry, internal pressure by red-fill transparency, valve state by translucent vs. solid appearance) alongside temporal plots of pump pressure, storage pressure, stored air volume, temperature, moisture, windspeed, and valve open/closed state. Step 4 (not shown) exports fabrication files and a material list.

do but why: through what sequence of pressure accumulation, valve triggering, and actuation. The designer can trace the causal chain from environmental input to physical output through an explicit representational pathway.

Third, the tool makes the system's behavior configurable through explicit representations of internal state. The designer does not adjust pneumatic behavior by modifying physical components directly; she selects parameters in a structured configuration interface and observes the predicted effect before fabricating anything. This closes the loop between design intent and system behavior at the representational level, rather than requiring physical iteration to discover mismatches.

The tool also structures a practical workflow that embeds the reasoning layer into the design process as a precondition for physical construction: component selection and parameter configuration, environmental cycle specification, behavioral preview, and fabrication export. Without the reasoning layer, the system is physically sound but practically inaccessible. With it, complex autonomous behavior becomes designable.

3.2.3 What Sustainflatable Establishes

Sustainflatable is not primarily a finding about sustainability or electronic-free pneumatics, though it demonstrates both. What it contributes to this dissertation is a concrete case for something more specific: that when a system acts on its own logic rather than real-time human instruction, a reasoning interface is not optional. It is what makes the system buildable in the first place.

The simplest configuration makes this clear without needing the complex cases. A thermal pump, a single storage unit, and a bursting valve already involve enough interacting parameters — pump output pressure, storage volume-pressure characteristics, valve threshold, environmental cycle

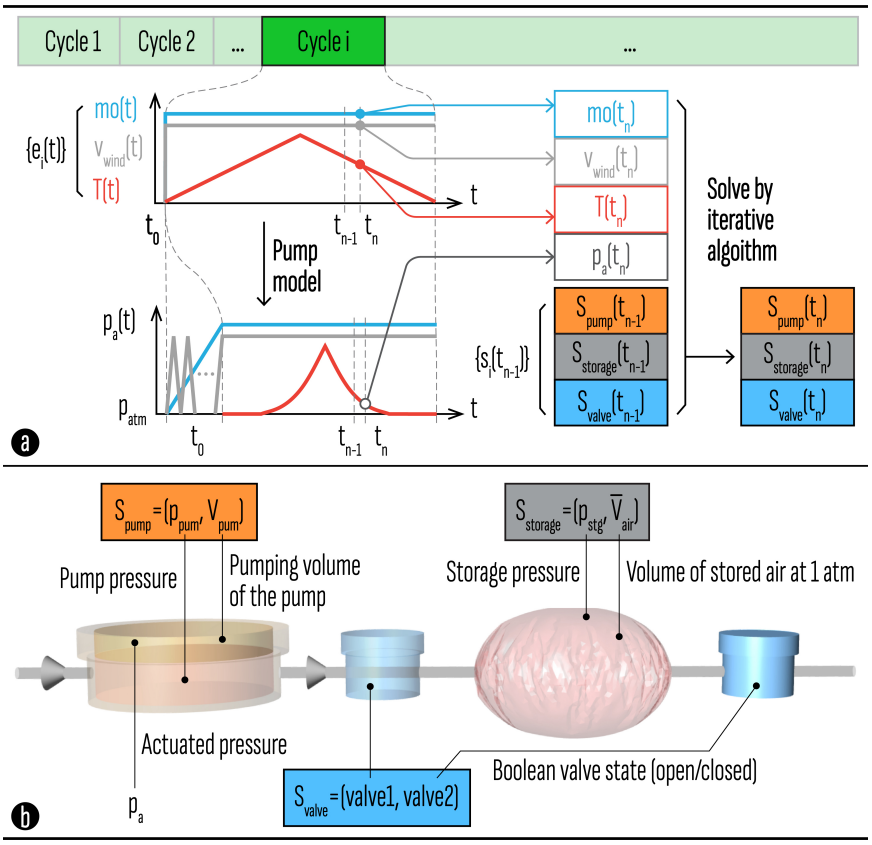


Figure 3.4: The computational model underlying the Sustainflatable design tool. (a) The main simulation workflow: environmental inputs $\{e(t)\}$ (moisture $mo(t)$, windspeed $V_{\text{wind}}(t)$, temperature $T(t)$) are sampled at discrete time steps within each environmental cycle i; the pump model converts environmental state to pump pressure $p_a(t)$; a quasi-static iterative algorithm advances the system state vector $\{S(t_n)\}$ — comprising pump pressure, storage pressure, and valve boolean states — at each time step t_n. (b) Definition of the three system status parameters: $S_{\text{pump}} = (p_{\text{pump}}, V_{\text{pump}})$, $S_{\text{storage}} = (p_{\text{stg}}, \bar{V}_{\text{air}})$, and $S_{\text{valve}} = (\text{valve}_1, \text{valve}_2)$.

length — that a designer working without simulation cannot reliably predict when the system will actuate. She can assemble a physically correct configuration and still not know whether it fires in the third environmental cycle or the ninth, or whether a ten-degree shift in ambient temperature pushes it past threshold a week ahead of schedule. More complex configurations — chained valves, multiple pumps, conditional logic across cycles — compound the problem, but they did not create it [67–70].

The same situation arises in any system that generates behavior from internal state rather than from direct human commands at each step. Before deployment, the designer needs to know what the system will do, under what conditions, and through what sequence of internal transitions. Without that knowledge, a technically correct system is still practically inaccessible — it cannot be deployed with confidence, diagnosed when something goes wrong, or modified when requirements change. The design tool is not a usability layer sitting on top of the physics. It is how the physics becomes something a designer can actually work with.

3.3 In-Situ Presenter: Reading Intent Across Modalities

3.3.1 The Problem: Virtual Presentation Without Physical and Spatial Cues

In a physical conference room, a presenter moves. She steps toward the diagram she is describing, gestures at the equation she wants the audience to follow, shifts aside when the slide needs full attention. Her body does the work of spatial coordination, and the audience reads it naturally [71–75]. Online, none of this translates. Her video feed is pinned to a corner — the same corner, for the entire talk — regardless of what slide she is on, what content she is referencing, or how she is gesturing [76–80]. The spatial expressiveness that is second nature in physical space has no equivalent channel in online presentation.

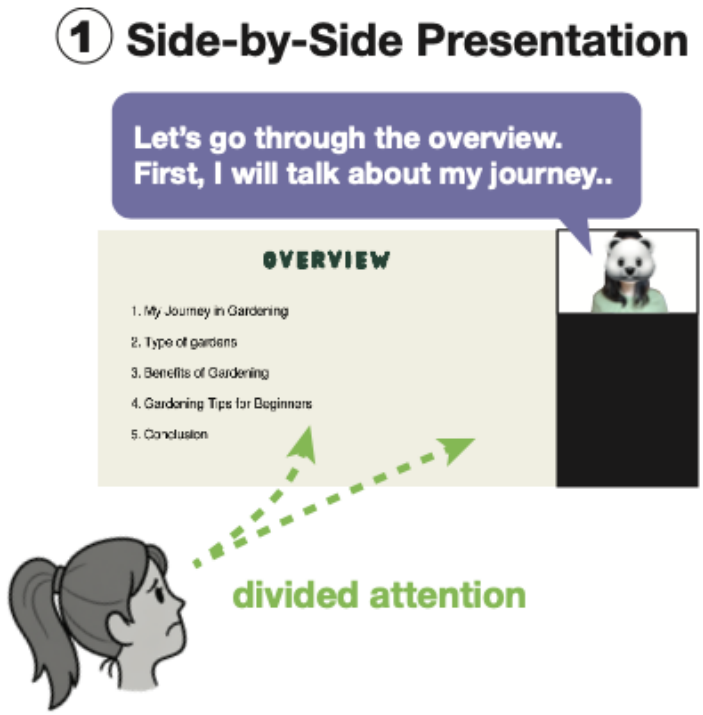


Figure 3.5: The online presentation problem. In the conventional side-by-side format, audience attention is divided between the slide and the presenter's video thumbnail, requiring frequent gaze transitions that interrupt comprehension and reduce perceived social presence.

Prior systems have approached this gap in two ways. Manual overlay tools — Zoom's Slides as Virtual Background, Microsoft Cameo, OBS — allow the presenter to reposition her video feed but require her to manage this positioning in real time, mid-talk, at exactly the moment when her cognitive resources are most fully committed to the presentation itself [81–84]. Content-overlay systems — RealityTalk, Chalktalk, Interactive Body-Driven Graphics — go in the opposite direction, overlaying interactive content on the presenter's video rather than positioning the presenter relative to slide content [85, 86]. Neither approach resolves the core problem: the former imposes cognitive overhead the presenter cannot afford; the latter changes the medium rather than closing the gap.

In-Situ Presenter proposes a different architecture: a system that reads the presenter's multimodal

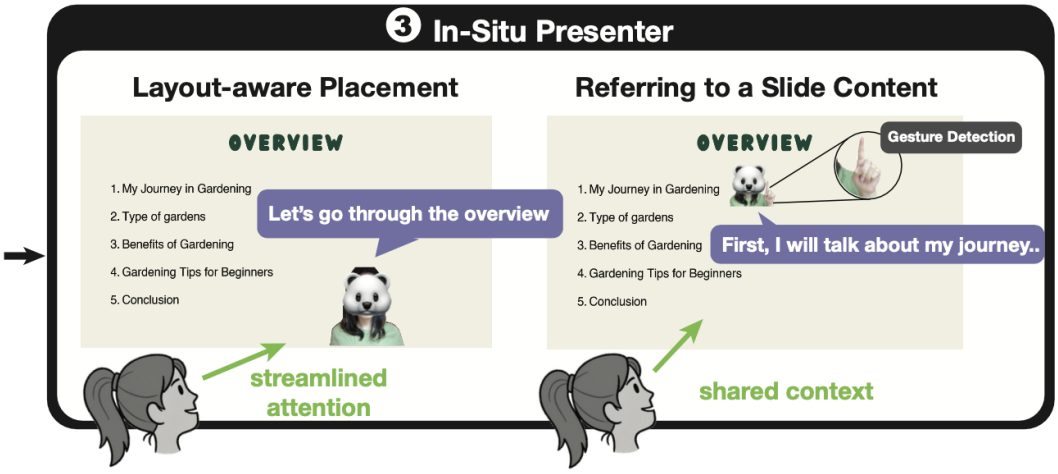


Figure 3.6: In-Situ Presenter's two core behaviors. Left: layout-aware placement — the system detects blank regions in the slide and positions the presenter's video in unoccupied space, streamlining audience attention by co-locating presenter and content. Right: gesture-triggered content referencing — when the presenter points, the system combines gesture detection with speech-to-script matching to reposition the video adjacent to the specific slide element currently being discussed, creating shared context between presenter and audience.

signals — speech, gesture, slide structure — and infers from them where she intends to be, then positions her there automatically. The mediation problem this involves is the inverse of Sustainflatable's. Where Sustainflatable needed to expose autonomous system reasoning outward for human inspection and configuration, In-Situ Presenter needs to read embodied human signals inward — to convert distributed, multimodal, often implicit expressions of intent into a structured representation the system can act on. The presenter is not configuring the system. The system is modeling the presenter.

3.3.2 ITalk-57: Grounding the Design in Observed Behavior

Before designing the system, the researchers conducted a two-phase formative study grounded in observed presenter behavior. The first phase was semi-structured interviews with experienced presenters who had created video presentations in in-situ formats for academic conferences and educational

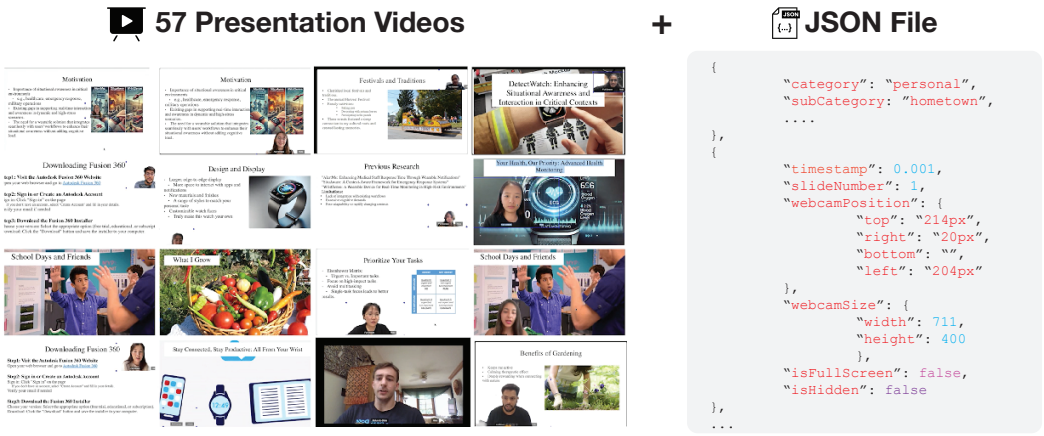


Figure 3.7: The ITalk-57 dataset. Left: a sample of the 57 annotated presentation videos spanning diverse slide layouts, content types (personal narrative, tutorial, academic), and presenter demographics. Right: the accompanying JSON annotation format, which records per-frame webcam bounding-box position and size, slide number, full-screen and visibility flags, and categorical metadata. The dataset provides an empirical behavioral record of how presenters use dynamic video placement when given unconstrained control — the observational foundation for In-Situ Presenter's design principles.

channels. These interviews established the primary motivation — enhancing audience engagement and social presence — and the primary friction: manual workflows relying on combinations of OBS, Zoom, and post-production video editing that were tedious, fragmented, and cognitively expensive.

The second phase produced ITalk-57: a dataset of 57 annotated presentation recordings totaling 275 minutes, each including time-stamped presenter video bounding boxes, slide content, and corresponding presentation scripts. Twenty participants used a custom presentation tool with drag-and-drop positioning, corner-based resizing, a full-screen mode, and a visibility toggle, instructed to adjust their video feed throughout the talk as they saw fit. The result is a behavioral record of how presenters use dynamic video placement when given control over it: not a normative ground truth, but an empirical characterization of the design space.

Analysis of ITalk-57 revealed several patterns that directly shaped the system's architecture.

Placement strategy was strongly predicted by slide layout archetype: on title slides, presenters clustered in the bottom corners; on bullet-point slides, they repositioned frequently to create visual-text associations between themselves and the specific content they were discussing [84,87]. Sizing adapted to presentation genre, with personal stories eliciting larger video feeds than tutorials, suggesting that social connection motivates more prominent self-presentation in narrative contexts. Temporal analysis revealed a consistent rhythm: dwell time was longest on full-figure slides and shortest on text-heavy slides, where transition frequency spiked. Nearly half of participants reported that the constant manual adjustment added to their cognitive load even as they valued the control it provided.

Two design principles emerged from this analysis. *Content-awareness*: the system must identify blank regions in the slide layout and avoid occluding authored content, because presenters consistently sought unoccupied space for their video feeds. *Presenter-intent alignment*: the system must position the video adjacent to the slide element semantically linked to the presenter's concurrent speech and gesture, because repositioning to create content associations was the most consistently reported motivation for dynamic placement. These two principles shaped the system's architecture directly.

3.3.3 System Architecture: A Unified Semantic Representation

In-Situ Presenter's architecture separates computationally expensive preprocessing from low-latency real-time inference. This separation reflects the structure of the mediation problem. The system needs two kinds of knowledge to act: knowledge about the slide's semantic structure (what regions contain what content, where blank space exists) and knowledge about the presenter's moment-to-moment intent (what content she is currently referencing, whether she is signaling a desire to reposition). The first kind is stable and computable in advance. The second is dynamic and must

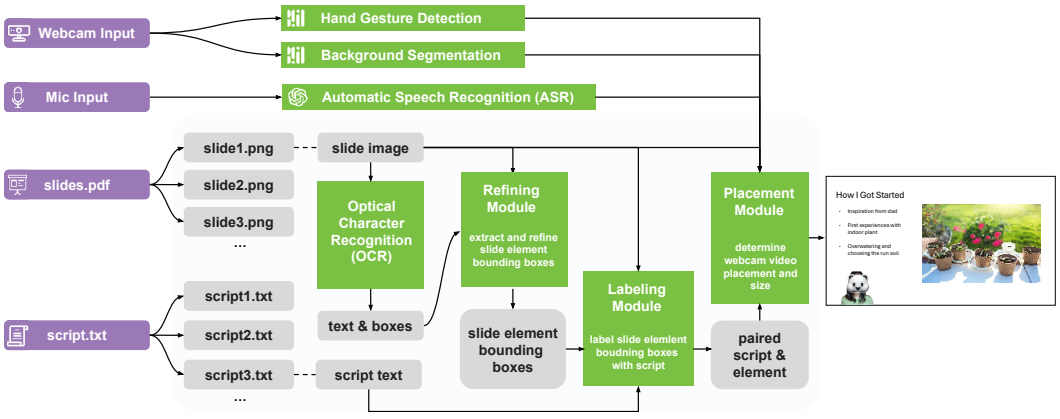


Figure 3.8: In-Situ Presenter's multimodal inference architecture. Offline preprocessing (purple): slides are processed through Tesseract OCR and GPT-4o to extract and refine semantic bounding boxes for text regions and graphics [88, 89]; the script is aligned to slide elements through a labeling module combining TF-IDF and dense sentence embeddings. Real-time pipeline (green): webcam input feeds parallel streams for hand gesture detection (MediaPipe) and background segmentation [90]; microphone input feeds an ASR module (Web Speech API / Whisper). The placement module fuses all streams — precomputed semantic map, real-time speech recognition output, and gesture detections — to compute and render the final video position on the slide.

be inferred in real time from imperfect signals.

The preprocessing pipeline addresses the first kind of knowledge. Slides are converted to images and processed through Tesseract OCR to extract word-level text and bounding boxes [88]. GPT-4o then refines these initial regions — merging word boxes into semantic units like full titles or bullet blocks, adjusting bounding boxes to better fit visual elements, and adding regions corresponding to non-text graphics that OCR missed. The result is a structured JSON description of each slide's semantic content, with normalized coordinates and consistent labels. This labeled structure serves two functions: as an occupancy grid for blank-space detection and as a vocabulary of alignment targets against which script segments can be matched.

Script-layout alignment transforms the presentation script into the same semantic vocabulary.

Script segments are aligned to slide elements through a two-step matching strategy: TF-IDF cosine similarity identifies lexical candidates, and dense sentence embeddings computed by the All-MiniLM-L6-v2 model re-rank these by semantic similarity. Each script segment is associated with the slide element it most closely references, and this association is stored as a semantic map from script positions to slide elements. The map is the reasoning layer's central data structure: it encodes the relationship between what the presenter plans to say and what the slide shows, in a form that can be queried efficiently when a placement decision is needed.

The real-time pipeline reads three input streams: automatic speech recognition (via Web Speech API for low-latency real-time behavior, with Whisper for higher-accuracy archival transcription), gesture recognition (MediaPipe Hands for stable pointing gestures), and the precomputed semantic map. Speech-to-script matching advances a current index through the script as recognized tokens match the pre-authored text, using Levenshtein edit distance to tolerate minor ASR errors. When a stable pointing gesture is detected with a valid speech-to-script match, the system repositions the video feed adjacent to the referenced slide component, computing candidate anchor positions and validating them against the occupancy grid. When no gesture is present, the system defaults to the highest-ranked unoccupied corner region computed in preprocessing.

The architecture's key contribution is not any individual component — OCR, semantic embedding, and gesture detection are each established methods — but their integration into a unified semantic representation that bridges the gap between embodied, distributed human signaling and the system's need to compute a discrete placement decision. No single modality suffices: speech without layout knowledge cannot determine where to place the video; layout knowledge without speech cannot determine what the presenter is currently discussing; gesture detection without script alignment cannot distinguish a pointing gesture toward the current slide element from an incidental hand movement. The

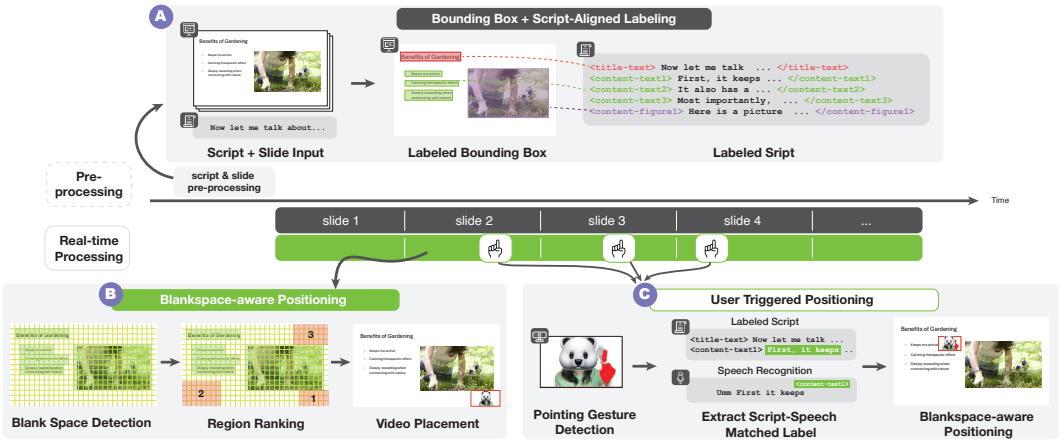


Figure 3.9: The In-Situ Presenter two-stage process visualized over time. (A) Preprocessing: script and slide inputs are analyzed to produce script-aligned semantic labeling — matched bounding boxes and labeled script segments. (B) Real-time default behavior: blankspace-aware positioning computes blank regions in the slide, ranks candidate anchor positions, and places the video feed in the highest-ranked unoccupied region as the presentation advances. (C) Real-time gesture-triggered behavior: a detected pointing gesture is fused with the current speech-to-script matched label to reposition the video adjacent to the specifically referenced slide element.

unified representation is what makes intent-driven placement possible.

3.3.4 Evaluation: Reducing Translation Burden While Preserving Expressive Agency

The evaluation of In-Situ Presenter used a dual-perspective design: a presenter-side experiment comparing the system against a manual overlay baseline, and an audience-side experiment comparing recordings in the In-Situ Presenter format against a conventional side-by-side layout. The two studies address the dissertation's central question from both ends of the communication channel: whether the system reduces the presenter's cognitive burden without absorbing her expressive agency, and whether the audience experiences the result as more coherent and connected.

The presenter-side results are clear. Cognitive load, as measured by the NASA Task Load Index, was significantly lower with In-Situ Presenter ($M = 26.13$) than with the manual baseline ($M =$

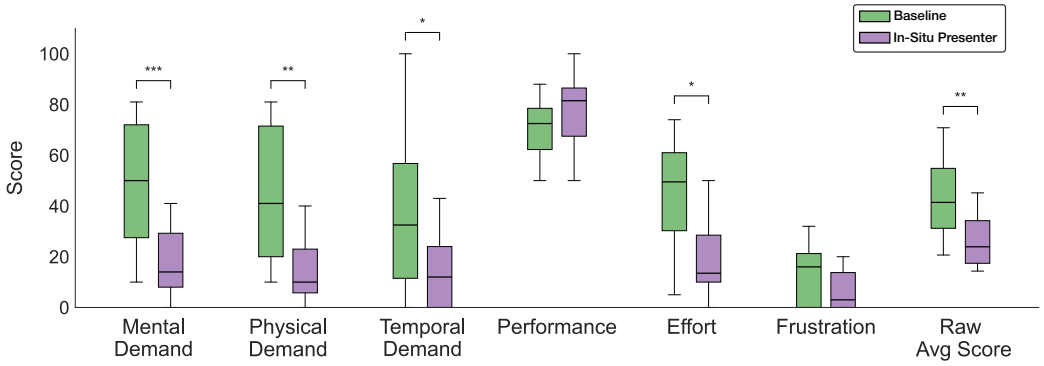


Figure 3.10: Presenter-side evaluation results (NASA Raw-TLX). Raw-TLX subscale scores comparing In-Situ Presenter (purple) against the manual overlay baseline (green). Significant reductions ($p < .05$, $p < .01$, $p < .001$) were observed in mental demand, physical demand, temporal demand, effort, and overall raw average score. Performance ratings did not differ significantly.

43.74), with reductions across the mental, physical, and temporal demand subscales. Preparation time was significantly shorter ($M = 58.75$s vs. 137.50s). These reductions in overhead were accompanied by increases in expressive behavior: presenters' hands were visible for nearly triple the duration ($M =$ 42.58s vs. 16.00s), and the system generated an average of 4.33 automated repositioning events per presentation, compared to 0.50 manual adjustments in the baseline. Presenters reported feeling more expressive ($M = 6.25$ vs. 4.67) and experienced significantly greater flow and engagement. When the translation work — managing video placement mid-talk — was offloaded to the system, the cognitive resources it was consuming became available for the actual work of presenting.

The audience-side results show a corresponding effect. Gaze analysis revealed a 40% reduction in transitions per minute between the presenter and slides (from 15.0 to 9.0), consistent with a reduction in the divided attention cost that the side-by-side format imposes. Disengaged gaze time dropped from 14.8% in the baseline to 3.5%. Perceived social connection was significantly higher for the In-Situ Presenter format ($M = 5.00$ vs. 2.67). Content comprehension did not differ significantly between

conditions, suggesting the dynamic format enhances connection without harming learning. Not all participants preferred the in-situ format: five of twelve found the moving video distracting, a finding the authors attribute to prior exposure and presentation context. This split points to an important design implication — the appropriate degree of presenter movement varies by context, and future systems should expose parameters that allow audiences or presenters to calibrate movement frequency and dwell time.

From this dissertation's perspective, the evaluation's most important finding is the alignment between subjective and behavioral evidence for increased expressiveness. Presenters did not merely report feeling more expressive — they gestured more, repositioned more, and prepared faster. The system's offloading of layout management created measurable behavioral changes, not just attitudinal ones. This is what effective mediation looks like empirically: not that users rate the system highly, but that the cognitive resources the system frees are actually redirected to the work the system was supposed to support.

The cases where the system misfired are equally instructive. When the speech recognizer lost track of the script, or when a gesture was detected against the wrong slide element, the presenter had no direct window into why the video moved where it did [1, 91]. The shared semantic representation that makes intent inference possible is also what makes misalignment opaque: the presenter sees the outcome but not the reasoning behind it. This points to a design requirement that goes beyond reducing workload. Intent-driven systems that act on inferred rather than stated goals need to make their reasoning accessible enough that users can identify and correct discrepancies — not just observe that something went wrong, but understand what the system thought they meant. The reasoning layer must be legible in the failure case, not only when the system is working correctly.

3.4 Cross-Project Synthesis: What Mediation at the Cognitive Layer Requires

3.4.1 One Requirement for Two Directions

Sustainflatable and In-Situ Presenter address reasoning problems from two inverse pathways. Sustainflatable must expose system reasoning outward to the human: make the logic of an autonomous physical system legible, predictable, and configurable to the humans responsible for designing and deploying it. In-Situ Presenter must interpret human intent inward to the system: convert distributed, multimodal, often pre-reflective signals from a human presenter into a structured internal representation the system can act on. The direction of the reasoning flow runs opposite ways. In Sustainflatable, the reasoning layer serves the human by making machine logic accessible. In In-Situ Presenter, the reasoning layer serves the machine by making human intent computable.

What these projects share is the discovery that effective mediation above the pre-reflective threshold requires an intermediate representation that sits between the human and the system, makes the logic of their interaction explicit, and creates a surface for intervention. In Sustainflatable, this representation is the simulation environment: temporal plots, pressure transparency indicators, and behavioral previews that translate pneumatic physics into a form a designer can reason about and configure. In In-Situ Presenter, it is the semantic map: the precomputed alignment between script segments and slide elements, constructed in preprocessing through lexical and semantic matching, which the real-time pipeline queries continuously as speech recognition advances a current script index — repositioning the video feed to a blank-space-aware default position as the talk progresses, and to a position adjacent to the currently referenced element when a pointing gesture is detected.

Both representations are, in DEXAS terms, reasoning layers: intermediate structures that sit

between raw complexity — pneumatic physics, or distributed multimodal signals — and the human actions those structures are supposed to support. In both cases, the reasoning layer is the scaffold that makes the system's behavior tractable — whether that means making machine logic legible to a human designer, or making human intent computable by a machine acting in real time.

3.4.2 The Boundary Between Pre-Reflective and Reflective Mediation

Chapter 2 established the conditions for pre-reflective mediation. Chapter 3 establishes the conditions for its limitations and what principles to follow when pre-reflective mediation is not possible. The boundary between the two runs through a specific point: where the system's behavior ceases to be a direct consequence of the user's action and begins to follow a logic of its own.

In Surface I/O, the system's "behavior" is entirely determined by the user's finger. The valley guides where the finger goes because the finger is going there; the detent marks a state because the finger has crossed a threshold. There is no gap between user action and system's passive response that requires explanation. In Motionless Movement, the system generates vibrotactile feedback, but that feedback is coupled to the user's force application in a way that mimics what the body would feel during actual movement. The system has a mapping algorithm, but that algorithm's purpose is to instantiate a sensorimotor contingency, rather than to make decisions or take actions on the user's behalf.

In Sustainflatable, the system decides through the sense-reason-react loop. Its valves open and close according to environmental thresholds the designer set, but those decisions unfold over time in ways that depend on environmental conditions the designer cannot fully anticipate. In In-Situ Presenter, the system infers. Its placement decisions follow from a combination of precomputed

alignment maps, real-time speech recognition, and gesture detection — processes that are probabilistic, error-prone, and only partially observable from outside. In both cases, a gap opens between user intent and system behavior that better surface design cannot close. What closes it is a reasoning layer that makes the gap itself more explainable.

The projects in this chapter are not harder versions of the same problem. They are a different problem — one that the physical layer principles were never designed to solve, and that required working out what mediation looks like when the system has a logic of its own.

3.4.3 The Irreducibility of Cognitive-Layer Requirements

Two requirements emerge from this chapter's empirical record that are distinct from those established in Chapter 2 and cannot be derived from them by extension or generalization.

The first is exposed reasoning transparency. The system must make its internal logic accessible in a form the human can actually use — not just observe, but evaluate, correct, and build on. In Sustainflatable, this means simulation outputs: behavioral previews that translate physical parameters into predicted temporal dynamics a designer can iterate on. In In-Situ Presenter, it means the subtitle display showing which script words are being matched, and the video placement showing which slide element the system has tied the current utterance to. Neither system exposes its reasoning fully — the designer never sees the differential equations, the presenter never sees the embedding similarity scores. But enough is visible that the user can form a working mental model and catch it when something goes wrong.

The second is adaptive input unification. When a system draws on signals from heterogeneous sources, those signals have to be integrated into a shared representation before any downstream inference

is possible. In Sustainflatable, the design tool's computational model does this work, bringing pump type, storage compliance, valve thresholds, and environmental cycle inputs into a single simulation architecture. In In-Situ Presenter, the semantic map does it, connecting script segments, slide element labels, speech recognition output, and gesture detections into a space where placement decisions can actually be computed. Without that unification step, neither system could function: the design tool cannot predict behavior without integrating pump, storage, and valve models together; the presentation system cannot determine placement from any one of its input streams alone.

These two requirements cannot be reduced to the principles from Chapter 2 because they address a different problem entirely. Embodied feedback coupling and structured signal decoding are about making information legible to the body — they work below the threshold of deliberate reasoning, where the goal is to avoid interpretation rather than support it. The cognitive layer principles work above that threshold, where interpretation is unavoidable and the quality of the representation directly determines the quality of the judgment it supports.

3.5 DEXAS Principles Established in This Chapter

Chapter 2 contributed the first two DEXAS principles: embodied feedback coupling (E) from Motionless Movement, and structured signal decoding (D) from Surface I/O. This chapter contributes the next two.

Exposed reasoning transparency (X). Sustainflatable and In-Situ Presenter together establish that systems operating above the pre-reflective threshold must expose their intermediate reasoning — not just their outputs — in a form that the humans responsible for them can inspect, evaluate, and act on. In Sustainflatable, this is the simulation environment that transforms pneumatic physics into a behavioral

preview a designer can iterate on. In In-Situ Presenter, it is the real-time subtitle display and video placement behavior that gives the presenter feedback on the system's current alignment state. The transparency is not total in either case — the full computational model is not directly observable — but it is sufficient. The user can tell whether the system is doing what she intends, and she can intervene when it is not. The X principle's empirical content is not that reasoning must be fully transparent, but that it must be transparent enough for the human to govern it.

Adaptive input unification (A). In-Situ Presenter demonstrates that intent-driven interaction across heterogeneous signal modalities requires a unified semantic representation as an intermediate layer: no single modality suffices, and the quality of the shared representation determines the quality of intent inference downstream. The semantic map between script segments and slide elements is the unifying architecture that makes multimodal placement possible. Sustainflatable's design tool instantiates the same principle in a different register: the computational model that unifies pump, storage, and valve parameters into a single simulation space is what makes complex autonomous behavior designable. In both cases, the unification is adaptive — the representation is updated as new inputs arrive, and its structure is determined by the demands of the downstream inference task rather than by the structure of any individual modality. The A principle's empirical content is that effective mediation of heterogeneous inputs requires not just aggregation but integration into a shared semantic representation that supports the inference the system needs to make.

These two principles, combined with the D and E principles established in Chapter 2, define four of the five requirements for effective mediation that the DEXAS framework formalizes in Chapter 4. The fifth — scalable knowledge grounding — becomes visible only at the scale of the PROMPTS system, where the mediation problem extends to multi-agent agentic workflows and the knowledge base required to support expert-level diagnostic reasoning cannot be fixed in advance. The progression

from Chapter 2 through Chapter 3 to Chapters 4 and 5 is not a survey of applications. It is an accumulation of empirical evidence for requirements that only became fully visible at each successive scale.

Chapter 4: The DEXAS Framework

From Empirical Findings to a General Theory of Reasoning Layers

The four projects covered in Chapter 2 and Chapter 3 were not initially planned as a continuous trajectory toward a single goal. Instead, they represent a series of distinct investigations into different corners of human-computer interaction. It was only in retrospect that their shared lineage became clear. They are united by a set of recurring failures: moments where the system's logic became illegible or where the handoff between human and machine broke down. These breakdowns provided the necessary evidence to build the framework I present here, proving that failure is often more instructive than a seamless user experience.

This chapter pulls together the key lessons from the projects covered so far to build a framework for what reasoning-layer mediation requires—DEXAS (Decoding, Embodied feedback coupling, eXposed reasoning transparency, Adaptive input unification, and Scalable knowledge grounding). To be clear, the projects covered so far aren't just instantiations of DEXAS. Instead, DEXAS is the 'big picture' that emerges when we look at these projects side-by-side—especially when we look at where they fell short. DEXAS is a set of standards for how a 'reasoning layer' should bridge the gap between what a human wants, and is able to understand vs. what a computer actually does, in its own low-level reasoning and representations.

The chapter proceeds in four stages. Section 4.1 extracts the empirical record: what each project established, where it fell short, and what those shortcomings imply about requirements the project itself did not fully satisfy. Section 4.2 derives the five DEXAS principles from that record, defines each precisely, and establishes why the set is necessary and sufficient for effective mediation at the level of abstraction this dissertation addresses. Section 4.3 describes the four structural components through which those principles are instantiated in a deployable system. Section 4.4 shows how the five

principles interact—how each depends on the others, and what breaks when any one is absent—and establishes DEXAS as a coherent framework rather than a list.

4.1 The Empirical Record: What the Projects Established and Where Their Limits Are At

4.1.1 Surface I/O: Structured Decoding at the Material Layer

Surface I/O demonstrated that raw sensory signals—the vibro-acoustic spectra generated by a finger moving across micro-patterned surfaces—can be transformed into structured, actionable information (position, direction, speed) through a multi-scale surface architecture and a machine learning classification pipeline. At the macro and meso scales, surface geometry encodes affordances that the sensorimotor system can understand without conscious interpretation: a valley guides lateral finger travel; a detent marks a discrete state boundary; a rough patch signals a zone limit. At the micro scale, features invisible to touch produce distinctive vibroacoustic signatures that a classifier can decode into input events. The result is a surface that simultaneously communicates what to do and registers what is being done—without instructions, labels, or dedicated mechanical components.

What Surface I/O established, for the purposes of this framework, is the general principle of structured signal decoding: that raw complexity—whether the vibro-acoustic signal of a finger movement, the utilization logs of a TPU cluster, or the distributed output of a multi-agent system—must be transformed into structured, human-legible representations, at the right abstraction level, before it can support human judgment. It's not just a nice-to-have feature. Without this transformation, the gap between what a system spits out and what a human can actually use remains unbridgeable. Think of a raw signal: it's just noise until it's been broken down into something meaningful. It only

becomes a true representation when the person on the other end can actually do something with it.

Surface I/O also established a constraint that the other projects would repeatedly encounter: structured decoding alone is not sufficient for effective mediation. The surface encodes what actions are possible; it does not, however, expose the logic by which it acts on those actions. A user who encounters an unexpected input classification result from a micro-scale texture has no window into the classifier's reasoning. This opacity is tolerable when the system is passive and the stakes are low—a misclassified swipe on a VR headset is a minor inconvenience. It becomes intolerable when the system is autonomous and the stakes are high, when the human responsible for the system's behavior needs to understand, predict, and correct it. Surface I/O's opacity is a feature at the physical layer. At every subsequent layer this dissertation examined, it becomes a liability.

4.1.2 Motionless Movement: Coupling Feedback to Action

Motionless Movement demonstrated that vibrotactile feedback, when coupled to the user's applied force through a grain-to-distance mapping that preserves the sensorimotor contingency of tactile surface friction, can produce a stronger kinesthetic experience than visual feedback—even in the absence of any actual movement. The feedback enters the user's sensorimotor loop rather than arriving as a signal that requires conscious interpretation. This is the core distinction the project uncovers: when information feels like a direct consequence of your own movement, rather than just an external report, the experience changes completely. It's the difference between the body "owning" the data and simply "receiving" it.

What this established, for the framework, is the principle of embodied feedback coupling: that mediation is most effective when system feedback is coupled to user action in a way that preserves

sensorimotor contingency, bypassing the cognitive layer rather than loading it. The study results are unambiguous on this point. Haptic feedback outperforms visual feedback as a standalone kinesthetic channel not because it carries more information, but because it enters the body's action loop directly. Visual feedback produces hermeneutic mediation: the user interprets the VR hand avatar's movement as evidence of their own motion. However, it lacks a crucial component of kinesthetic feedback. As a result, the visual movement of the VR hand can feel less like one's own hand moving and more like controlling an external device—similar to moving a mouse and watching the cursor move on screen. Haptic coupling produces embodied mediation—the user's body treats the vibrotactile response as a consequence of what it is already doing. These are not different degrees of the same experience. They are categorically different modes.

Motionless Movement also pointed toward a limit that the project did not itself transcend. Embodied feedback coupling works when the user's action is the primary driver of the system's response—when the coupling is direct and the feedback is a transformation of what the user is doing. It cannot be the primary mediation strategy when the system's behavior is determined by its own internal logic rather than by the user's current action. A surgeon's robotic assistant that acts on its own inference about tissue type cannot couple its feedback to the surgeon's force in the same way that a vibrotactile display couples to finger pressure. The system's autonomy introduces a gap between user action and system response that coupling alone cannot bridge. Closing that gap requires making the system's reasoning visible—a different mode of mediation, addressed by different principles.

4.1.3 Sustainflatable: Transparency as an Architectural Requirement

Sustainflatable demonstrated that an autonomous physical system—one that acts without valves, electronics, or real-time human instruction—cannot be reliably designed or deployed without a human-facing reasoning interface. The system's physical behavior was technically correct before the design tool existed. What was not correct was the designer's ability to reason about it: to predict what the system would do under varying environmental conditions, to configure its autonomous logic through explicit representations of internal state, to anticipate how its temporal dynamics would unfold across multiple environmental cycles. The design tool made that physics legible, predictable, and configurable—and in doing so, transformed a technically correct system into a deployable one.

This established, for the framework, the principle of exposed reasoning transparency: that systems operating above the pre-reflective threshold must expose their intermediate reasoning in a form that the humans responsible for them can inspect, evaluate, and act on. Not the full computational model—the differential equations and quasi-static iteration algorithms that underlie the simulation are not directly presented to the user. What is presented are the representations those computations produce: temporal plots of pressure accumulation, three-dimensional previews of component states, behavioral forecasts under specified environmental inputs. These representations give the designer what she needs to govern the system without requiring her to be a pneumatic engineer.

Sustainflatable also established that transparency alone is not sufficient. The design tool makes the system's behavior legible and configurable; it does not make the system responsive to the designer's intent in real time. The designer configures the system in preprocessing; the system then acts on its own. The reasoning layer is unidirectional: it exposes system logic outward but does not read designer intent inward. This is adequate for the Sustainflatable use case—the deployment context is outdoor,

unattended, and long-horizon, so real-time intent inference is not required. But it points toward a requirement that In-Situ Presenter would need to address: a mediating system that must act in real time on behalf of a human who cannot stop to configure it must do more than expose its reasoning. It must also read the human's reasoning—infer intent from the multimodal signals the human is producing, without requiring the human to state that intent explicitly.

4.1.4 In-Situ Presenter: Unification of Heterogeneous Signals

In-Situ Presenter demonstrated that intent-driven interaction across heterogeneous signal modalities requires a unified semantic representation as an intermediate architecture: no single modality suffices, and the quality of the shared representation determines the quality of intent inference downstream. The system unifies speech recognition tokens, script-to-slide alignment maps, gesture detections, and slide layout occupancy grids into a single semantic space where placement decisions can be computed. The unified representation is not an output the user sees—it is an internal architecture that makes real-time intent inference possible.

What this established, for the framework, is the principle of adaptive input unification: that mediating systems must integrate heterogeneous input signals into shared semantic representations that support downstream inference. The requirement is not aggregation—simply collecting multiple input streams—but integration: constructing a representation in which the different signals' contributions are commensurable, so that the system can reason across them rather than sequentially consulting each. The semantic map between script segments and slide elements is the clearest instance of this. It is not a speech representation, not a visual layout representation, not a gesture representation—it is a representation in which these three are unified, so that a pointing gesture toward a specific slide region

can be interpreted in the context of what the presenter is currently saying and what the slide shows.

In-Situ Presenter also exposed a gap that neither the physical layer projects nor Sustainflatable had encountered at full scale: the extensibility problem. The system was built for a specific genre of presentations—academic conference talks with slide-based content and scripted delivery. Its classification pipeline, its semantic alignment architecture, and its gesture detection model were all calibrated to this genre. When the presentation format changes—a lecture with handwriting on a whiteboard, a demonstration without a script, a panel discussion without prepared slides—the system's assumptions break down. Not all of them: the principle of unifying heterogeneous signals into a shared representation travels to any multimodal intent-inference task. But the specific representations, the specific modalities, and the specific alignment strategies must be rebuilt from scratch for each new domain.

This is the fifth requirement—the one that only becomes fully visible when the framework must travel. A mediation system that works well in one domain but cannot be extended to another without architectural redesign is not a general solution. It is a domain-specific implementation. The requirement for scalable knowledge grounding—a modular substrate of domain knowledge that can be extended without structural redesign—does not arise urgently in any of the four projects examined so far. It becomes unavoidable in PROMPTS, where the domain is large-scale ML system optimization and the knowledge required to mediate expert-level reasoning cannot be anticipated in advance. Before turning to that instantiation, the principles must be stated with full precision.

4.2 The Five DEXAS Principles

The empirical record of Chapters 2 and 3 points to five requirements for effective mediation between human intent and system execution. These requirements are stated here as principles—

not design prescriptions, since the form of their instantiation varies across domains, but necessary conditions that any effective mediating system must satisfy. Each principle addresses a distinct failure mode; together, they define what DEXAS requires.

Principle D: Structured Signal Decoding

Raw system signals must be transformed into structured, human-legible representations at the appropriate level of abstraction before they can support human judgment.

The physical embodiment of this principle is Surface I/O's multi-scale architecture: micro-scale vibroacoustic spectra are classified into input events; macro-scale geometry is structured into affordance maps. Its cognitive embodiment is Sustainflatable's design tool: differential equations governing pneumatic behavior are transformed into temporal plots and behavioral previews. In PROMPTS, the same principle operates at the largest scale: gigabytes of HLO trace logs, roofline data, and memory utilization statistics are transformed into structured bottleneck diagnoses that identify the type, location, and likely cause of each performance bottleneck—before the engineer is asked to reason about them.

The principle has two components that must both be satisfied. First, the decoding must be structural: it must produce representations in which the relevant distinctions are explicit rather than embedded in raw data. A utilization percentage that requires the engineer to infer bottleneck type is not decoded; a report that states "memory-bound bottleneck at layer 47, likely caused by inefficient sharding of the embedding table" is. The distinction is not between compressed and verbose representation, but between a representation that delegates inference to the human and one that has already performed it. Second, the decoding must be calibrated to the appropriate level of abstraction for the downstream task. A representation that is too low-level requires additional interpretation; one that is too high-level elides details necessary for valid judgment. The right level of abstraction is the level at which the

human's expertise can most efficiently be applied: the level at which her judgment adds value rather than decoding overhead.

The failure mode this principle guards against is cognitive overhead from raw complexity: the engineer who spends days cross-referencing trace logs before she can formulate a hypothesis, the presenter who must parse her own slide structure before she can understand why the system placed her video where it did, the designer who cannot reason about a physically correct autonomous system because its behavior is opaque. In each case, the system has produced output. The output is not yet a representation the human can work with. Structured signal decoding is the transformation that closes that gap.

Principle E: Embodied Feedback Coupling

System feedback should be coupled to user action in ways that preserve sensorimotor contingency, entering the user's action-sensation loop rather than arriving as a signal that requires conscious interpretation.

This principle is most directly instantiated in Motionless Movement, where it takes its purest form: vibrotactile feedback coupled to force application in a way that mimics the sensorimotor contingency of surface friction, producing kinesthetic experience without movement. But the principle is not confined to haptic interfaces. Its general form is: the system's response to user action should feel like a consequence of that action, not like a report about it. The key distinction is between feedback that enters the sensorimotor loop—that the body processes as part of the action itself—and feedback that arrives outside that loop and must be consciously decoded.

At the physical layer, this distinction maps cleanly onto Ihde's embodied versus hermeneutic modes of mediation: the valley that guides the finger without requiring deliberate attention versus the label that describes where the finger should go. At higher levels of abstraction, the principle requires

a different formulation. In a cognitive system like PROMPTS, there is no sensorimotor loop to enter in the haptic sense. But the principle still applies, transposed to the cognitive register: feedback should be coupled to the user's reasoning process in a way that flows into it rather than interrupting it. A system that returns a diagnosis whose structure mirrors the way an expert would reason about the problem—diagnosing bottleneck type, then hypothesizing cause, then proposing configuration changes, then justifying the proposal—provides feedback that enters the expert's reasoning workflow directly. A system that returns a ranked list of configuration options with no explanatory structure provides feedback that must be consciously decoded: the expert must reconstruct the reasoning herself, from the output backward.

The failure mode this principle guards against is interpretive overhead: the cognitive cost of decoding feedback that arrives in a form the user was not built to read directly. This overhead is not merely a tax on efficiency. It occupies the same cognitive resources as the actual work the feedback is supposed to support. An expert reading a bottleneck report whose structure does not match her reasoning workflow is doing two tasks at once: interpreting the report and applying her expertise. A system that couples its output to the structure of her expertise gives her one task: judgment.

Principle X: Exposed Reasoning Transparency

Systems operating above the pre-reflective threshold must expose their intermediate reasoning—the logic by which they move from inputs to outputs—in a form that the humans responsible for them can inspect, evaluate, and act on.

The physical layer projects instantiate this principle only minimally—and intentionally so. Surface I/O's opacity is appropriate: a surface that exposed its classification algorithm to the user would impose the very interpretive overhead the multi-scale affordance architecture was designed to eliminate. Motionless Movement's opacity is similarly appropriate: a vibrotactile display that narrated its sensorimotor

contingency mapping would defeat the purpose of making feedback enter the body's action loop rather than the mind's interpretation loop. Pre-reflective mediation succeeds precisely because it makes its machinery invisible.

Above the pre-reflective threshold, this invisibility reverses from feature to liability. Sustainflatable's design tool exposes the system's reasoning through simulation outputs—behavioral previews that translate pneumatic physics into a form a designer can iterate on. In-Situ Presenter exposes its reasoning through behavioral feedback—the subtitle display showing which script words are being matched, the video placement indicating which slide element the system has aligned the current utterance to. In PROMPTS, the reasoning is exposed through explicit diagnostic reports and justified proposals: the system does not return a configuration recommendation without a bottleneck diagnosis that explains why that configuration addresses the identified problem.

The principle has a precise formulation that distinguishes it from the weaker claim that systems should be "explainable" in some general sense. The exposure must be sufficient for governance: the human must be able to tell whether the system is doing what she intends, and she must be able to intervene when it is not. This is not total transparency. The user of Sustainflatable's design tool does not see the differential equations; she sees the temporal plots they generate. The engineer using PROMPTS does not see the embedding similarity scores that drove the knowledge retrieval; she sees the bottleneck diagnosis and proposal that resulted. What she sees is enough: enough to evaluate whether the system's reasoning is sound, enough to identify when it has made a wrong assumption, enough to override or extend the proposal with her own judgment. Transparency that is insufficient for governance is not transparency in the sense this principle requires. Transparency that overwhelms the human with unstructured internal detail is not transparency either—it is a different form of the raw-complexity problem that the D principle addresses.

The failure mode this principle guards against is epistemic opacity: the engineer who cannot tell whether the system understood the problem or got lucky, the designer who cannot predict what the autonomous system will do, the presenter who cannot diagnose why the video moved to the wrong position. In each case, the system has acted. The human cannot govern what it did, cannot correct it when it fails, and cannot build on it when it succeeds. Exposed reasoning transparency is the condition under which autonomous and agent-driven systems remain within the scope of human judgment rather than escaping it.

Principle A: Adaptive Input Unification

Mediating systems must integrate heterogeneous input signals into a shared semantic representation that supports downstream inference, rather than processing each modality independently.

In-Situ Presenter is the clearest instantiation: speech recognition tokens, script alignment maps, gesture detections, and slide layout structures are unified into a single semantic space where placement decisions are computed. No individual modality suffices. Speech without layout knowledge cannot determine where to place the video. Layout knowledge without speech cannot determine what the presenter is currently discussing. Gesture detection without script alignment cannot distinguish a purposeful pointing gesture from incidental hand movement. The unified representation is what makes multimodal intent inference possible.

The principle is not confined to multimodal interaction in the conventional sense. Sustainflatable's design tool instantiates it in a different register: pump type parameters, storage compliance curves, valve threshold specifications, and environmental cycle inputs are unified in a single simulation architecture that produces coherent behavioral predictions. The heterogeneity here is not perceptual-modality heterogeneity but parameter-type heterogeneity: the inputs are numerically incommensurable, and only a computational model that defines a shared representational space across them can produce

predictions that integrate their effects. In PROMPTS, the principle operates at the largest scale: KPI metrics, HLO trace profiles, roofline analysis data, and expert knowledge retrieved from a document store are unified in a diagnostic representation that mediates between raw system state and the structured proposals the system generates.

Adaptive input unification has two components. The first, unification, is the requirement that inputs from different sources be represented in a common semantic space where their contributions are commensurable. The second, adaptive, is the requirement that the unified representation be dynamically updated as new inputs arrive, rather than being fixed at system initialization. In-Situ Presenter's semantic map is recomputed as the presenter advances through the script; PROMPTS' diagnostic workspace accumulates intermediate hypotheses as the analysis proceeds. In both cases, the unification is not a static data structure but an evolving representation that tracks the current state of the system's understanding of the task.

The failure mode this principle guards against is modality fragmentation: the system that interprets each input source independently and produces outputs that fail to integrate what the different sources jointly imply. A presentation system that processes speech and gesture in separate pipelines and combines their outputs by a fixed voting rule will fail when the modalities conflict. A diagnostic system that processes utilization data and memory logs in separate analysis modules and presents their findings side by side will leave the engineer to integrate them herself, reinstating the translation burden the system was supposed to eliminate.

Principle S: Scalable Knowledge Grounding

The knowledge structures that support a mediating system's reasoning must be organized in a form that can be extended with new domain knowledge without requiring architectural redesign.

The first four principles describe what a system must do with the information available to

it: decode it structurally, couple feedback to user action, expose intermediate reasoning, and unify heterogeneous inputs. The fifth principle describes a constraint on how the system's knowledge must be organized: in a form that can grow. It is the only DEXAS principle that addresses the long-run sustainability of the mediation rather than its immediate quality.

The first four projects do not fully surface this requirement because they were each designed for a relatively bounded domain. In each case, the knowledge required to support the system's reasoning could be fixed in advance—either because the domain was narrow enough that the relevant knowledge was bounded, or because the system was a research prototype not expected to generalize beyond its evaluation context. The extensibility problem is latent in each project; it is not encountered at a scale that demands a general solution.

PROMPTS surfaces it unavoidably. The domain—large-scale LLM system optimization—is not bounded. New model architectures, new hardware configurations, new sharding strategies, and new performance bottleneck types emerge on a timescale measured in months. A system whose diagnostic reasoning is fixed in its architecture will become obsolete as the domain evolves. A system that requires architectural redesign to incorporate new knowledge will not be maintained. The requirement is for a knowledge substrate that is modular—organized so that new knowledge can be added as discrete units—and compositional—organized so that the reasoning system can combine knowledge units flexibly in response to the specific problem at hand. In PROMPTS, this takes the form of a retrieval-augmented knowledge base over expert documents, past optimization records, and hardware-specific scaling guides: the diagnostic agent retrieves relevant knowledge at inference time rather than having that knowledge baked into its parameters or its prompts.

The scalable knowledge grounding principle has an important implication that extends beyond the technical requirement of extensibility. It establishes the boundary between what a mediating system

must know and what it must be. The first four principles describe properties of the system's behavior. The fifth principle describes a property of the system's architecture: its capacity for growth. A system that satisfies D, E, X, and A but not S will work well in the domain it was designed for and degrade as the domain evolves. A system that satisfies all five will improve as the domain evolves, incorporating new knowledge without requiring the humans responsible for it to redesign it from scratch.

4.3 The Four Structural Components of DEXAS

The five principles describe requirements. They do not prescribe architecture. Different instantiations of DEXAS can satisfy the same principles through different structural choices, depending on the domain, the scale, and the nature of the human-machine interaction the system must support. Nevertheless, the principles converge on four structural components that recur across the systems examined in this dissertation and that appear to be necessary features of any deployable DEXAS instantiation.

4.3.1 Unified Semantic Workspace

The Unified Semantic Workspace is the shared representational environment in which the system maintains a common semantic form across raw signals, intermediate hypotheses, user intent, and system state. It is the structural home of the A principle (adaptive input unification) and the D principle (structured signal decoding): the workspace is where heterogeneous inputs are unified, and where the decoding from raw signals to structured representations occurs.

In In-Situ Presenter, the unified semantic workspace is the script-layout alignment map and the slide occupancy grid, maintained in memory and updated as the presentation progresses. In Sustainflatable's design tool, it is the simulation state—the current values of pump parameters, storage

fill, and valve status—that the tool maintains across design iterations. In PROMPTS, it is the diagnostic workspace: a structured representation of the current bottleneck analysis that accumulates intermediate diagnoses, retrieved knowledge, and proposed configurations as the agentic pipeline proceeds.

The key property of the Unified Semantic Workspace is that it is a shared representation accessible to both the human and the system. The system uses it to reason; the human inspects it to understand what the system is doing and to intervene when the system's reasoning diverges from her intent. This shared access is what makes the workspace a mediation surface rather than merely an internal data structure. In PROMPTS, the diagnostic report that the Analyzer agent produces is not only input to the Proposal agent—it is the primary output the engineer reads, the representation she uses to evaluate whether the system has correctly characterized the optimization problem before she reviews its proposals.

4.3.2 Collaborative Agentic Workflows

The Collaborative Agentic Workflows component is the orchestration architecture through which the system coordinates multiple reasoning processes, each operating at a different level of abstraction or with a different knowledge focus, in service of a shared task. It is the structural home of the X principle (exposed reasoning transparency) and a partial instantiation of the E principle (embodied feedback coupling) at the cognitive level: the workflow is designed so that the system's sequential reasoning steps mirror the structure of the expert's reasoning process, coupling the system's output to the form of the expert's thinking.

The collaborative architecture is designed not to replace expert reasoning but to scaffold it. Each step in the workflow is a point of potential human intervention: the expert can review the diagnostic

report before the proposal is generated, modify the framing of the problem if the diagnosis is incorrect, or override the proposed configuration if her judgment diverges from the system's. The workflow does not proceed by presenting the human with a final recommendation and asking for approval or rejection. It proceeds by surfacing intermediate reasoning at each step, giving the expert something to evaluate and intervene on before the next step is taken.

In Sustainflatable, the collaborative workflow is the design iteration cycle: configure, simulate, review behavioral preview, refine parameters. In In-Situ Presenter, the real-time constraint compresses the full design iteration into a continuous inference-and-act cycle, but the collaborative structure is maintained through the transparency of the subtitle display and the override capacity available to the presenter. In PROMPTS, the multi-agent architecture makes the collaborative workflow fully explicit: the Coordinator orchestrates a Diagnose $\rightarrow$ Propose $\rightarrow$ Justify pipeline in which each agent's output is available for human inspection before the next agent proceeds.

4.3.3 Human-Workflow-Informed Reasoning

Human-Workflow-Informed Reasoning is the principle that the system's internal reasoning process should follow the structure of the expert's existing workflow, producing intermediate outputs that align with the stages of the expert's own reasoning rather than with the stages of the system's computation. It is a cognitive-level instantiation of the E principle: the system couples its reasoning structure to the expert's reasoning structure, so that its outputs enter the expert's thinking as natural next steps rather than arriving as foreign representations requiring translation.

The significance of this component is easiest to see in its absence. A black-box optimizer that returns a ranked list of sharding configurations has no Human-Workflow-Informed Reasoning

component. It produces output that requires the engineer to do the diagnostic work herself—to reconstruct, from the ranked list, why these configurations were recommended, which bottleneck they address, and whether the system's implicit diagnosis matches her own reading of the profiling data. The system has done the search; it has not done the reasoning. The engineer is still doing translation work before she can do judgment work.

PROMPTS inverts this. The system mirrors the expert workflow—diagnose bottleneck → hypothesize cause → propose configuration → justify proposal—so that its output arrives in the form the expert would use if she were walking a junior engineer through the analysis. The expert reads the diagnosis first. If it matches her own reading, she has strong grounds for trusting the proposal. If it does not, she has a specific point of divergence to interrogate rather than an opaque recommendation to accept or reject wholesale. The reasoning is not hidden behind the result. It is the result—the structured form through which the expert's judgment is engaged rather than bypassed.

4.3.4 Extensible Knowledge Base

The Extensible Knowledge Base is the modular substrate through which domain knowledge is stored, retrieved, and incorporated into the system's reasoning. It is the structural home of the S principle (scalable knowledge grounding) and provides the substrate on which the other three components depend for domain-specific inference quality.

The Extensible Knowledge Base has three properties that distinguish it from a static knowledge representation. First, it is modular: domain knowledge is organized in discrete, independently retrievable units—documents, heuristics, past optimization records, hardware-specific scaling guides—rather than baked into the system's parameters or prompts. This modularity is what makes extension possible:

a new hardware architecture can be incorporated by adding new documents to the store, without modifying the reasoning agent's instructions. Second, it is retrievable rather than prescriptive: the system retrieves relevant knowledge at inference time based on the specific problem at hand, rather than applying a fixed procedure for every problem of a given type. Third, it is separable from the system's architecture: the reasoning agents can be designed, evaluated, and modified independently of the knowledge they consult.

In PROMPTS, the Extensible Knowledge Base is a retrieval-augmented store indexing expert documents on ML system optimization, past successful optimization records, and hardware-specific performance guides. The zero-knowledge stress test reported in Chap. 5 directly demonstrates the principle: when the knowledge base lacks information about a new hardware architecture, the system's reasoning fails—but factually, not structurally. The reasoning process remains valid; it simply lacks the domain knowledge to apply it correctly. When the relevant hardware specifications are added to the knowledge base, performance recovers without any modification to the reasoning agents. This is the empirical signature of Scalable Knowledge Grounding: knowledge failures are local and recoverable by knowledge extension; reasoning failures require architectural intervention.

4.4 How the Principles Interact: DEXAS as a Coherent Framework

The five principles are not independent requirements that a system can satisfy in isolation. They form a system of mutual dependencies in which the value of each principle depends on the others, and the failure of any one undermines the rest.

Structured signal decoding (D) is the precondition for everything else. A system that does not decode raw signals into structured representations cannot expose its reasoning (X)— there is nothing

structured to expose. It cannot couple its feedback to the expert's workflow (E)—the feedback arrives in a form the workflow was not designed to receive. It cannot unify heterogeneous inputs (A)—unification requires that inputs be in a commensurable semantic form. It cannot ground its knowledge scalably (S)—knowledge retrieval operates on structured queries, not raw signals. The D principle is not the most important of the five. But it is the one that must be satisfied first, because every other principle depends on its products.

Embodied feedback coupling (E) and exposed reasoning transparency (X) are in structural tension—a tension that turns out to be productive. Embodied feedback coupling pushes toward opacity: the best haptic interface is one that disappears into use, that requires no conscious attention, that makes its machinery invisible. Exposed reasoning transparency pushes toward legibility: the best autonomous system is one that makes its logic fully visible, that invites interrogation, that shows its work. The resolution is domain-dependent: below the pre-reflective threshold, E takes precedence; above it, X does. This is not a contradiction in the framework. It is the framework's most precise empirical finding: the effective balance between coupling and transparency depends on where in the cognitive stack the mediation is operating.

Adaptive input unification (A) is the condition under which both D and E can operate with full effectiveness. The D principle requires that raw signals be decoded into structured representations; the A principle ensures that those representations are commensurable across modalities, so that the decoding produces a coherent picture rather than a collection of independent readings. Without unification, decoding produces fragmented representations and coupling produces responses that are sensitive to one signal while ignoring the others.

Scalable knowledge grounding (S) is the condition under which the other four principles can be sustained as the domain evolves. A system that satisfies D, E, X, and A at deployment but

lacks a scalable knowledge substrate will degrade as the domain changes—not because its reasoning architecture has failed, but because the knowledge that architecture requires is no longer current. S is the only principle that is explicitly prospective: it addresses not the quality of the mediation at a given moment but the system's capacity to remain a quality mediator as the problem changes around it.

These dependencies have a practical implication for system design. DEXAS is not a checklist. A system that satisfies four of the five principles and neglects the fifth will not achieve eighty percent of the mediation quality it would achieve if all five were satisfied. The principles interact nonlinearly: the missing principle tends to cascade. A system without D produces representations that neither E, X, nor A can work with. A system without X produces proposals the expert cannot evaluate, undermining the value of the D and A principles that produced them. A system without S works well until the knowledge base goes stale, at which point the quality of D, E, X, and A decays together. The framework is robust only when all five conditions are met.

4.5 DEXAS and the Existing Literature

DEXAS draws on several traditions without being reducible to any of them. Situating the framework relative to adjacent bodies of work clarifies what it adds and where it departs.

The human-automation interaction literature has long investigated the conditions under which automation improves human performance and when it degrades it. The central finding of this literature is that automation can reduce workload while also reducing situational awareness, creating conditions for catastrophic error when the automated system fails and the human must take over from a degraded understanding of the current state [**?**, **?**]. DEXAS addresses this failure mode directly through the X and A principles: exposed reasoning transparency maintains situational awareness by keeping the

system's logic visible; adaptive input unification maintains the quality of the human's model of the system's current understanding. But DEXAS departs from most of the human-automation literature in its emphasis on the structure of the mediation rather than the degree of automation. The literature tends to ask: how much should the system automate? DEXAS asks: what must the system make visible, regardless of how much it automates?

The explainable AI (XAI) literature has produced a substantial body of work on methods for making machine learning systems interpretable—feature importance scores, attention visualization, counterfactual explanation generators, and so on [92, 93]. DEXAS is sympathetic to this literature's goals but distinguishes itself on two axes. First, XAI typically treats explanation as a post-hoc addition to a system whose architecture was not designed for transparency—an explanation layer grafted onto a system that was not built to be explained. DEXAS treats transparency as an architectural requirement, not an afterthought: the X principle is a design constraint from the beginning, not a feature added after the fact. A system designed to hide its reasoning and then asked to report on it will report what is convenient. A system designed from the outset to expose intermediate reasoning will produce transparency that is functionally grounded rather than decorative. Second, XAI focuses primarily on explaining individual decisions. DEXAS is concerned with the ongoing governance of the system's reasoning over a task: not just why it produced this output, but whether the overall trajectory of its reasoning is sound and where the human can most effectively intervene.

The mixed-initiative systems literature [94, 95] is the closest ancestor of DEXAS in the AI and HCI literature. Mixed-initiative systems are designed so that both human and machine can initiate actions and respond to each other's contributions, with control flowing between them as the task evolves. DEXAS shares this commitment to bidirectional agency but adds two requirements that mixed-initiative systems have not consistently addressed. First, the D principle: mixed-initiative

systems often assume that both parties are working with representations the human can read directly. DEXAS takes the decoding problem seriously—the system's contribution to the shared task is not only to initiate actions but to transform the raw complexity of the problem into representations that make the human's contribution possible. Second, the S principle: mixed-initiative systems were conceived before the era of large-scale knowledge-intensive AI, and the question of how the system's knowledge substrate scales over time was not central to their design. DEXAS treats scalable knowledge grounding as a first-class architectural requirement.

4.6 Summary: DEXAS as a Framework for Reasoning Layers

DEXAS is a framework for reasoning layers—intermediate representations that sit between human intent and system execution, making the logic of that mediation explicit, inspectable, and actionable. Its five principles define what a reasoning layer must provide. Its four structural components define how those principles are instantiated in a deployable system. Its theoretical grounding establishes where it fits in the existing landscape of human-machine mediation research and what it contributes to that landscape.

The framework is not a design prescription. Different instantiations will look different: a reasoning layer for a haptic interface looks nothing like a reasoning layer for a multi-agent ML optimization system. What unifies them is the set of requirements they satisfy—the principles, not the implementation—and the function they serve: reducing the translation burden between human intent and system execution while preserving human agency over the decisions that matter.

The argument of this dissertation up to this point has been empirical: four projects that each, in different ways, uncovered one or more of the requirements that DEXAS formalizes. The argument

from this point forward is instantiational: Chap. 5 presents PROMPTS as a complete instantiation of all five DEXAS principles in a high-stakes domain, demonstrating that the framework is not only theoretically coherent but practically effective at the scale where the human middleware problem is most severe.

Chapter 5: PROMPTS — An Agentic Reasoning Layer for ML System Optimization

Chapter 4 synthesized the preceding projects into DEXAS: five principles—Decoding, Embodied feedback coupling, eXposed reasoning transparency, Adaptive input unification, and Scalable knowledge grounding—that characterize what effective mediation requires when a system must translate between human intent and low-level system representations. DEXAS was not designed at the outset. It is a set of requirements that emerged across five projects each time the question was asked: why did the mediation work here, why did it fall short there, and whether the same approach could be made to work somewhere else. Those questions were asked in relatively bounded domains—passive physical interfaces, autonomous pneumatic systems, intent-driven presentation tools. PROMPTS asks them in a domain with no such bounds.

PROMPTS is an instantiation of DEXAS in a domain with no such bounds. Large-scale ML system optimization—specifically, the problem of identifying efficient sharding configurations for distributed LLM training and serving—is combinatorially complex, hardware-specific, and deeply expertise-dependent. Engineers who work in this domain routinely manage search spaces exceeding 150 valid configurations per workload, interpret dense profiling traces that require years of domain experience to read correctly, and repeat the entire process from scratch when the model architecture or hardware platform changes [96–98]. Existing automated approaches either perform black-box search without domain understanding—requiring hundreds of costly evaluations—or build specialized performance models that are brittle and non-transferable [99–104]. Neither approach addresses the problem at the level where it actually occurs: the translation between what a profiling trace says and what an engineer needs to do.

PROMPTS reframes this translation as a reasoning problem. Its Analyzer Agent converts raw profiling data—key performance indicators, HLO operation profiles, roofline analysis—into structured

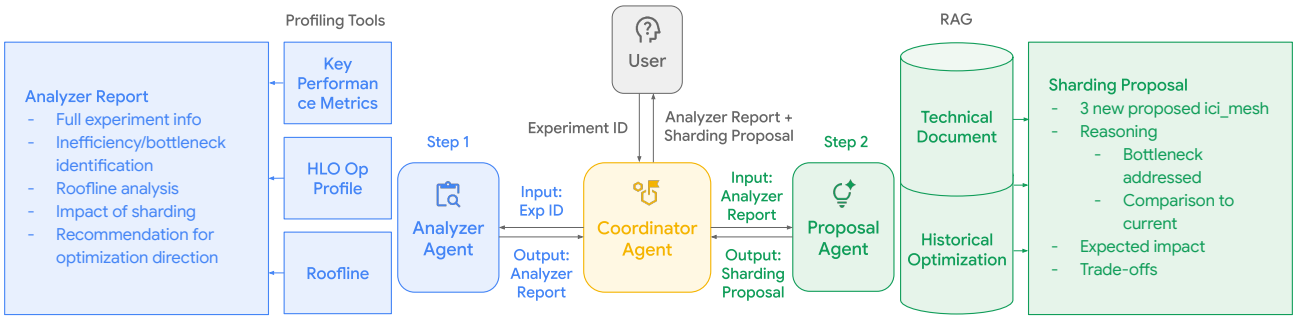


Figure 5.1: Overview of the PROMPTS agentic performance optimization framework. The Coordinator Agent routes a user-supplied experiment ID through the Analyzer Agent—which ingests KPI metrics, HLO operation profiles, and roofline data from the Xprof API—to the Proposal Agent, which performs retrieval-augmented generation over a two-source knowledge base (historical optimization records and expert-written technical guides) to produce three ranked sharding proposals with justifications.

bottleneck diagnoses. Its Proposal Agent takes those diagnoses and queries a knowledge base of expert-written optimization guides and historical optimization records to generate justified sharding configurations. Together they operationalize the two-step reasoning process that experienced engineers perform mentally: diagnose, then propose. The difference is that PROMPTS performs this in under a minute, with no prior exposure to the specific workload, and produces recommendations that engineers adopt as their top choice 87.5% of the time across eight real-world production cases.

This chapter presents PROMPTS as a full instantiation of the DEXAS framework and evaluates what that instantiation achieves. Section 5.1 characterizes the optimization problem and explains why existing approaches fail to address it. Section 5.2 maps PROMPTS's architecture onto each of the five DEXAS principles. Section 5.3 presents the evaluation results across five lenses: search-space compression, solution correctness, reasoning legibility, extensibility, and human-role reconfiguration. Section 5.4 examines three cases in depth, including the boundary case where the system fails and the zero-knowledge stress test that isolates the value of domain knowledge. Section 5.5 situates PROMPTS within the dissertation's broader argument.

5.1 The Problem: Human Bottleneck in ML System Tuning

5.1.1 Why Sharding Optimization Is Hard

The central task of sharding optimization is to identify, from a combinatorially large search space, an ICI-mesh configuration that fits the model within device memory while minimizing communication overhead and maximizing compute utilization. An ICI mesh defines how a model's data, parameters, and activations are partitioned across the high-speed interconnect of a TPU slice: how many chips handle data parallelism, how many handle model parallelism, how many handle sequence parallelism. For a moderately large workload on a 256-chip system, the number of valid configurations—those that pass compiler validation at all—can exceed 165. The optimal configuration depends on the interaction among the model's architecture, the hardware's network topology, and the workload's computational profile. Changing any of these three factors can invalidate conclusions drawn from extensive prior experimentation.

The difficulty compounds because performance trade-offs are non-linear and often counterintuitive. Increasing model parallelism reduces per-device memory pressure but introduces collective communication operations that can dominate step time if the interconnect becomes saturated. Increasing sequence parallelism reduces activation memory but generates 'ragged' tensor collectives whose overhead is not proportional to sequence length. The same configuration that is optimal on a 2D Torus network may be deeply suboptimal on a 3D Torus network, because the two topologies favor different communication patterns. An engineer who has built up intuitions about one hardware generation must partially relearn them for the next.

The standard workflow that engineers follow in this environment is essentially iterative trial-

and-error: run an experiment, collect a profiling trace, analyze the trace for bottlenecks, adjust the configuration, repeat. Each iteration takes minutes to hours, depending on model size and hardware availability. The analysis step—reading the profiling trace and inferring what it implies for configuration changes—requires deep domain expertise and is not automatable by the profiling tools themselves, which surface what is slow but not what to do about it. This is precisely the gap that PROMPTS targets.

5.1.2 Why Existing Approaches Miss

Automated black-box search methods address the iteration problem by replacing manual trial-and-error with guided search algorithms—Bayesian optimization, integer linear programming, randomized search over structured spaces [99, 103–106]. These approaches have demonstrated success navigating large configuration spaces, and they form the backbone of production optimization workflows at most major ML labs [107, 108]. Recent work has further leveraged machine learning to build performance models that improve search efficiency [109, 110], and agentic LLM-based systems have begun addressing code transformation and tensor optimization [111–115]. But they have three structural limitations that become visible under the conditions where PROMPTS is tested.

First, black-box methods are knowledge-blind. They cannot represent the fact that a workload is memory-bound rather than compute-bound, and they therefore cannot avoid spending evaluations on configurations that would exacerbate the memory bottleneck rather than resolve it. They treat the search space as a black box to be sampled rather than as a structured space whose high-potential regions can be identified before sampling begins. As a result, they frequently require evaluating the entire valid search space before finding the configuration that domain knowledge would have identified

immediately.

Second, the performance models they build are brittle. The surrogate function a Bayesian optimizer learns over one hardware platform encodes something like the performance landscape of that specific combination of model, hardware, and workload type. When any of these three factors changes—a new model architecture, a new TPU generation, a different parallelism topology—the model becomes unreliable, and in some cases misleading. The knowledge that makes the model good at predicting performance in the seen case is not transferable to the unseen case. PROMPTS's knowledge base, by contrast, encodes general optimization principles that remain valid across hardware generations: the relationship between model parallelism and communication overhead, the symptom signature of HBM-bound workloads, the conditions under which sequence parallelism introduces memory violations. Seven of PROMPTS's eight evaluation cases required reasoning not present in the historical database.

Third, black-box search scales poorly with the dimensionality of the optimization space. ICI-mesh sharding is one variable among many: batch size, rematerialization strategy, compiler flags, and offloading decisions also affect performance, and their interactions are just as complex as the sharding problem alone. A complete joint optimization over all of these dimensions would produce a search space that is computationally intractable for any sample-based approach. PROMPTS's reasoning-based architecture scales to joint optimization by adding knowledge modules for each new dimension rather than by increasing sample count [116–121].

5.2 PROMPTS as a DEXAS Instantiation

Each of the five DEXAS principles addresses a specific failure mode in the human middleware problem. Decoding fails when raw system signals—profiling traces, log files, execution graphs—require expert translation before they carry any actionable meaning. Embodied feedback coupling fails when system feedback is decoupled from the human's action attempt, requiring an additional interpretive layer. eXposed reasoning transparency fails when a system's intermediate steps are hidden, leaving the human with outputs she cannot interrogate. Adaptive input unification fails when heterogeneous signals are processed in isolation rather than fused into a shared representation. Scalable knowledge grounding fails when domain knowledge is baked into the system architecture rather than held in a modular substrate that can grow. PROMPTS instantiates all five, and the evaluation results are legible as evidence for each.

5.2.1 Decoding

The Decoding principle requires that a system convert low-level representations—raw signals, execution traces, system logs—into structured intermediate artifacts at a level of abstraction the human can work with directly. In ML system optimization, the raw material is the profiling trace: a dense record of execution-level events, memory bandwidth measurements, and collective operation timings that requires years of domain experience to read correctly. The trace is technically complete—everything relevant to the optimization is encoded in it—but it is not cognitively accessible. It does not say which operation is causing the bottleneck, what type of bottleneck it is, or what class of configuration change would address it. An engineer reading a trace must perform that translation herself.

PROMPTS's Analyzer Agent performs this decoding step. It ingests three data types—key

performance indicators (accelerator duty cycle, step time, communication overhead), HLO operation profiles (execution time, memory bandwidth utilization, roofline efficiency per operation), and on-device roofline data—and produces a structured bottleneck report that names the dominant inefficiency, categorizes it as compute-bound, HBM-bound, or communication-bound, identifies the specific operations responsible, and provides a recommendation for optimization direction. The report has the same section structure as the analysis an expert engineer would write by hand: KPI health check, HLO bottleneck identification, correlation of the two, structured diagnosis. This is not a summary of the trace; it is a translation of the trace into the vocabulary of performance engineering decisions.

5.2.2 Embodied Feedback Coupling

The Embodied feedback coupling principle, as established in Chap. 2, describes mediation that works because feedback enters the user's sensorimotor loop directly rather than requiring deliberate interpretation. That mode of mediation is not available in ML system optimization, where the 'action' is submitting an experiment and the 'feedback' arrives hours later as a profiling trace. But the principle survives in a weaker, cognitive-layer form: feedback should be coupled to the specific action that generated it, at the granularity where the engineer can trace the connection.

PROMPTS instantiates this by structuring its output around the causal chain from configuration choice to performance consequence. The Proposal Agent's justification for each candidate configuration specifies which bottleneck the configuration addresses, how it modifies the current sharding in the relevant dimension, what performance impact is expected from that modification, and what trade-offs it introduces. This structure couples the proposal—the recommended action—to the diagnosis—the evidence that motivated it. An engineer reviewing a proposal can follow the reasoning from trace

observation to configuration recommendation in a single document. The coupling is not pre-reflective, but it is tight: the feedback is specific enough to the action that the engineer does not need to reconstruct the reasoning chain herself.

5.2.3 eXposed Reasoning Transparency

The eXposed reasoning transparency principle requires that a system's intermediate reasoning be available for inspection, not merely its final output. This is the principle that most directly distinguishes PROMPTS from black-box optimization: a black-box optimizer returns a configuration, and an engineer can evaluate whether that configuration works, but she cannot evaluate whether the optimizer's reasoning was sound. A configuration that works for the wrong reasons is a configuration she cannot trust to generalize.

PROMPTS exposes its reasoning at two levels. The Analyzer Agent's bottleneck report is the first level: it surfaces the evidence chain from raw metrics to diagnosis, making explicit which observations support which conclusions. The Proposal Agent's justification dialogue is the second: when an engineer challenges the agent's top recommendation, the agent defends it by reasoning back to the profiling evidence. It does not produce a confidence score from an opaque model; it cites the device duty cycle, the top HLO operations and their bottleneck classifications, and the relationship between the current sharding configuration and the observed inefficiency. It distinguishes between what the data shows and what remains uncertain, and it rates its confidence in terms of the strength of the evidence chain.

Sharding Memory—the persistent log of all tool calls, intermediate agent responses, and user inputs—extends this transparency across the session. An engineer can review not just the agent's final

proposal but the sequence in which its hypotheses developed, the tools it called to gather evidence, and the points at which it updated its diagnosis. This audit trail is what allows the zero-knowledge stress test to be informative: the performance engineer who annotated the agent's output could identify exactly where the reasoning was correct, where it applied a valid principle to the wrong case, and where it lacked the context to reach the right conclusion. The gap was legible because the reasoning was visible.

5.2.4 Adaptive Input Unification

The Adaptive input unification principle requires that a system fuse heterogeneous input signals into a shared representation that can be reasoned over, rather than processing them in isolation. In In-Situ Presenter, the relevant signals were speech, gesture, and slide structure—each encoding a different aspect of the presenter's intent, none sufficient alone. In PROMPTS, the relevant signals are KPI metrics, HLO operation profiles, and roofline data—each encoding a different aspect of the workload's performance characteristics, none sufficient alone.

A KPI metric like device duty cycle produces an initial hypothesis: the workload appears compute-saturated. HLO profiles either confirm or complicate that hypothesis: in Case 2, a high duty cycle masked an HBM bottleneck that only became visible when the top operations were individually examined and classified by roofline analysis. Roofline data provides the third layer: it distinguishes between operations that are compute-bound and those that are memory-bandwidth-limited, at the per-operation level where the diagnosis becomes actionable. The Analyzer Agent does not report these three data sources separately; it synthesizes them into a single diagnosis that neither source could support alone. The bottleneck classification—compute-bound, HBM-bound, communication-bound—

is an emergent property of the fusion, not of any individual input.

5.2.5 Scalable Knowledge Grounding

The Scalable knowledge grounding principle requires that a system's domain knowledge be held in a modular substrate that can be extended without requiring changes to the reasoning engine. This is the architectural property that makes PROMPTS extensible in a practically meaningful sense: not that it handles every workload correctly out of the box, but that when it does not, the fix is tractable.

PROMPTS's knowledge base consists of two sources: a structured database of past engineer-submitted optimization cases, and a document knowledge base of expert-written technical guides covering foundational principles and hardware-specific strategies. The Proposal Agent queries both via retrieval-augmented generation, with the underlying LLM formulating semantic queries based on the bottleneck report rather than keyword matching. The agents themselves contain reasoning procedures, not domain facts. When a new hardware platform is added, the relevant documentation is added to the knowledge base; the agents are unchanged. When a new model architecture introduces unfamiliar computational patterns, a technical guide describing those patterns is added; the agents are unchanged.

The zero-knowledge stress test establishes the boundary of this property empirically. In the zero-knowledge condition, the agent's reasoning procedure remained sound: it correctly identified the dominant bottleneck operation and produced technically coherent proposals. Its errors were knowledge errors, not reasoning errors—a missing root-cause explanation for a tensor relayout operation, a general fusion recommendation that the hardware's memory architecture made inapplicable, an analysis of an all-gather operation that missed its mandatory function in the specific workload. Each error maps to a named document that would have corrected it. The knowledge base grows; the agents remain stable.

5.3 Evaluation

Table 5.1: Summary of experimental configurations for extensibility testing using real production workloads. We separate the **ML Task** from the **Agent's Task**. System versions are annotated with their network topology. Proprietary model names have been anonymized.

Case	TPU System	Topology	Model Name / Type	ML Task	Agent's Task
1	`v5p (3D Torus)`	8x8x8	Proprietary Large Dense Model	Reward Modeling (Post-training)	Optimize `ici_mesh` for existing topology
2	`v6e (2D Torus)`	4x4	Proprietary Small Dense Model	SFT (Post-training)	Optimize `ici_mesh` for existing topology
3	`v5e (2D Torus)`	16x16	Proprietary Medium Dense Model	SFT (Post-training)	Optimize `ici_mesh` for existing topology
4	`v6e (2D Torus)`	16x16 → 8x16	Proprietary Medium Dense Model	SFT (Post-training)	**Generate:** Adapt `ici_mesh` for new topology
5	`v6e (2D Torus)`	16x16	Proprietary MoE Model	Reward Modeling (Post-training)	Optimize `ici_mesh` for existing topology
6	`tpu7x (3D Torus)`	- (2 chips)	Qwen 32B (Open Source Dense)	Inference	**Generate:** Propose `ici_mesh` for new model
7	`tpu7x (3D Torus)`	- (256 chips)	DeepSeek (Open Source MoE)	Pre-training	Optimize `ici_mesh` for existing topology
8	`v5p (3D Torus)`	4x8x16	Proprietary Audio Generation Model	Audio Generation (Post-training)	Optimize `ici_mesh` for existing topology

Table 5.2: Summary of Experimental Results. The agent consistently identifies high-performance configurations with minimal search effort compared to the total size of the valid search space.

Case	% Perf. Improvement[a]	Agent Search Effort[b]	Blackbox Search Effort[c]	# Agent Invocations[d]	Agent Batch Size[e]	Compilability[f]	Top Pick Adopted?[g]
1	40.83%	1	55	1	3	100%	Yes
2	42.59%	1	35	1	3	100%	Yes
3	79.15%	1	165	1	3	66.67%	Yes
4	14.29%	2	45	1	3	33.33%	No
5	434.75%	1	45	1	2	50%	Yes
6	182.86%	1	2	1	1	100%	Yes
7	104.16%	1	9	1	3	66.67%	Yes
8	26.25%	1	55	1	3	33.33%	Yes
					Average	*69%*	*87.5%*

[a] **% Perf. Improvement**: The percentage increase in throughput (e.g., steps/sec), calculated from the reduction in end-to-end step time.
[b] **Agent Search Effort**: The number of agent-suggested configurations evaluated until the desired one was found.
[c] **Blackbox Search Effort**: The total number of configurations in the search space, equivalent to the evaluations needed in an exhaustive (brute-force) search to find the optimal configuration.
[d] **# Agent Invocations**: The number of times the agent was executed to generate configuration suggestions.
[e] **Agent Batch Size**: The number of `ici_mesh` configurations provided by the agent per invocation.
[f] **Compilability**: The percentage of suggested configurations in the agent's batch that successfully compile.
[g] **Top Pick Adopted?**: Indicates if the agent's top-suggested configuration was the one ultimately adopted by the engineer.

PROMPTS was evaluated across eight real-world production workloads, drawn from engineer-submitted optimization cases at Google. The workloads span four TPU generations (v5p, v5e, v6e, tpu7x), 2D and 3D Torus network topologies, computation scales from 2 to 2,048 chips, dense and Mixture-of-Experts model architectures, and the full ML lifecycle—pre-training, supervised fine-

tuning, reward modeling, and inference serving. The evaluation was structured to test five properties: whether the system compresses the search space, whether its solutions are technically correct and match expert-validated outcomes, whether its reasoning is legible and inspectable, whether it extends to new hardware and model types via knowledge injection, and whether it reconfigures the human role toward higher-level judgment.

5.3.1 Search-Space Compression

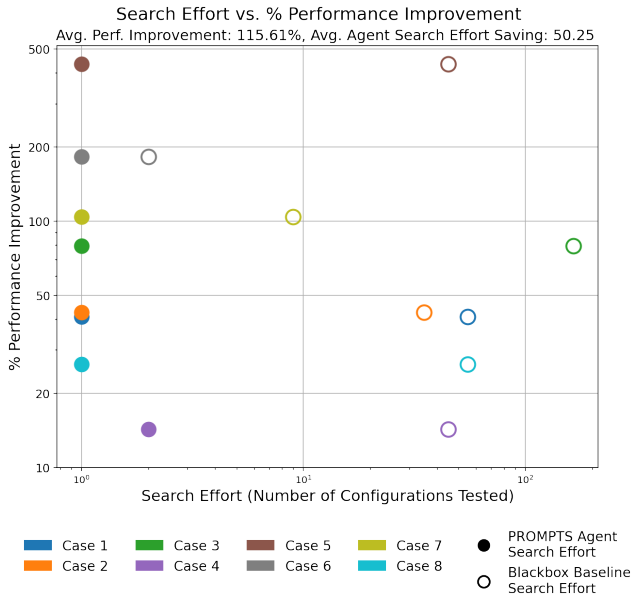


Figure 5.2: PROMPTS Agent vs. Blackbox Baseline search effort for a given performance improvement across eight experiments. For each experiment (unique color), the line connects the PROMPTS Agent's search effort (solid dot) to the Blackbox baseline search effort (hollow dot). Search effort, plotted on a log scale, is defined as the number of configurations that were actually tested until the expert-validated configuration was identified. While both methods yield the same average performance improvement (115.61%), the agent's search effort is consistently and significantly lower, saving an average of 50.25 experimental runs per task.

The most direct evidence for the system's value is the reduction in search effort it achieves relative to black-box baseline. Search effort is defined as the number of configurations actually

evaluated before finding the expert-validated solution. Across all eight cases, PROMPTS required a single invocation and, in seven of eight cases, identified the expert-validated configuration as its first suggestion. The black-box baseline required between 2 and 165 evaluations to find the same configuration [105, 106]. The average search-effort saving was 50.25 experimental runs per task, representing a reduction of roughly two orders of magnitude in the worst case.

This compression is not achieved by accident. It is the direct result of applying domain knowledge at the point where it can most dramatically reduce the search space: before any configuration is evaluated. The agent's first-pass diagnosis—identifying whether the workload is compute-bound, HBM-bound, or communication-bound—eliminates entire regions of the configuration space from consideration. A memory-bound workload will not benefit from configurations that increase model parallelism without sharding the model; a communication-bound workload will not benefit from configurations that further increase collective operation frequency. Black-box methods cannot reason about these constraints without evaluating their consequences; PROMPTS identifies them from the profiling trace.

The throughput improvements achieved by the agent's top recommendations range from 14.29% to 434.75%, with an average of 115.61%—identical to the black-box baseline average. The agent does not sacrifice solution quality for search efficiency. It finds solutions of the same quality as exhaustive search but reaches them through a single informed step rather than through iterative sampling.

5.3.2 Solution Correctness Under Mediation

A reasoning system that produces fast but wrong answers is of no practical value in a production optimization context. PROMPTS validates its solutions at two levels. The first is compilation: invalid sharding configurations—those that are memory-infeasible, topology-incompatible, or otherwise malformed—

are rejected by the XLA compiler before execution, providing a deterministic correctness check at zero additional instrumentation cost. The agent's compilability rate averages 69% across its batch of three proposed configurations per invocation. This rate is not perfect, but it is high enough that the agent consistently includes a valid configuration in its top three, and in 87.5% of cases the top-ranked proposal is both compilable and matches the production-validated configuration.

The second level is matching: whether the agent's top-ranked configuration is the configuration that engineers actually adopted after extensive manual analysis or automated search. In seven of eight cases, it is. This metric captures something that compilation alone cannot: not just whether the configuration works, but whether it is the right configuration—the one that best balances performance, memory footprint, and communication overhead for the specific workload. An 87.5% match rate on first-ranked proposals, across diverse workloads and hardware, with a single invocation per task, is strong evidence that the agent is reasoning about the domain rather than retrieving cached answers.

5.3.3 Reasoning Legibility

Each agent response in the evaluation follows a consistent structure: diagnosis, proposal, comparison to baseline, expected impact, trade-offs. This structure is not cosmetic. It is what allows an engineer to evaluate the agent's reasoning rather than merely its conclusion. In the three detailed qualitative case studies documented in the appendices, a senior performance engineer reviewed the agent's output and found no factual errors in its diagnoses or reasoning chains for Cases 1 through 3. The agent correctly identified compute-bound, HBM-bound, and communication-bound workloads respectively, and its proposals in each case directly addressed the diagnosed bottleneck through mechanisms it articulated and defended.

The legibility of the system's reasoning is most visible in the justification dialogues. When challenged—'Are you really sure? How confident are you?'—the agent does not produce a confidence score from an opaque model. It reasons back to the profiling evidence: the device duty cycle, the top HLO operations and their bottleneck classifications, the relationship between the current sharding configuration and the observed inefficiency. It distinguishes between what the data shows and what remains uncertain. It rates its own confidence in terms of the strength of the evidence chain, not in terms of a prior calibration. This is the behavior of a system that understands its reasoning, not merely a system that produces structured output.

5.3.4 Extensibility via Knowledge Injection

Six of the eight evaluation cases involve workloads not present in the historical optimization database: the agent must reason from first principles, using only the general optimization guides and the bottleneck report, to propose configurations for models and hardware it has not seen before. In all six cases, it succeeds. Cases 4 and 6 go further: neither has a baseline configuration to optimize from. The agent must generate an ICI mesh from scratch for a new topology (Case 4) and a new open-source model on a new hardware platform (Case 6). In both cases it produces valid, performant proposals in a single invocation.

The zero-knowledge stress test, discussed in detail in Sec. 5.4.3, establishes the specific boundary of this extensibility. In the zero-knowledge condition, the agent's high-level reasoning procedure remains sound: it correctly identifies the dominant bottleneck operation and produces technically coherent proposals. What it lacks is the platform-specific and model-specific knowledge that would allow it to identify the root cause of the bottleneck at a finer grain, recognize when a general optimization

principle does not apply on this specific hardware, or identify sequential anti-patterns that require global analysis of the execution graph. These are factual knowledge gaps, not reasoning gaps. Each one is addressable by adding the relevant documentation to the knowledge base.

This distinction—between reasoning procedure and domain knowledge, held in separate factored components—is what makes the system extensible in a practically meaningful sense. A new hardware generation does not require rebuilding the agent. It requires adding documentation about the hardware's specific architectural constraints and optimization patterns. A new model architecture does not require new agent training. It requires adding documentation about the model's computational profile and known optimization strategies. The knowledge base grows; the agents remain stable.

5.3.5 Human-Role Reconfiguration

The 87.5% adoption rate for the agent's top recommendation is the most direct evidence of human-role reconfiguration. Engineers who review the agent's proposals and find their own preferred configuration among them—as the first-ranked choice—in seven out of eight cases are engineers whose primary work has shifted from configuration generation to configuration review. They are no longer constructing the candidate; they are evaluating whether the candidate the agent produced is the one they would have produced.

This shift is qualitatively different from automation that removes the human from the loop. The agent does not decide which configuration to deploy. It proposes, justifies, and ranks. The engineer reviews, questions, and adopts or rejects. The qualitative dialogues make this visible: engineers who question the agent's recommendation receive a reasoned defense that they can evaluate on the merits. Engineers who disagree with the recommendation can ask for alternatives. The interaction is not a

handoff from human to machine; it is a reassignment of which steps in the reasoning process each party is responsible for. The human remains at the judgment layer. The agent occupies the labor-intensive translation layer between profiling data and configuration hypothesis.

5.4 Cases in Depth

5.4.1 Three Archetypal Bottleneck Cases

Cases 1, 2, and 3 each instantiate a different fundamental bottleneck type—compute-bound, HBM-bound, and communication-bound, respectively—and together demonstrate that PROMPTS's diagnostic procedure generalizes across the major categories of performance inefficiency in distributed LLM workloads. Case 1, a reward modeling workload on TPU v5p-512, presented a near-perfect device duty cycle (99.7%) with low communication overhead (2.6%). The Analyzer correctly diagnosed a compute-saturated workload and recommended increasing data parallelism at the expense of sequence parallelism, reasoning that this would better saturate compute units without introducing communication bottlenecks. The recommendation matched the expert-validated production configuration exactly.

Case 2, a supervised fine-tuning workload on TPU v6e-16, presented a more subtle failure mode. The high device duty cycle was consistent with a compute-bound interpretation, but HLO analysis revealed that the top operations were all memory-bandwidth-limited: the device was busy, but busy waiting for data from HBM rather than performing computation. The Analyzer correctly identified this anti-pattern—high duty cycle masking a memory bottleneck—and diagnosed the cause: the current configuration replicated the full model on each chip, generating HBM pressure that was not visible in the KPI summary. The proposal introduced 4-way model parallelism to shard the model and directly reduce per-chip memory footprint. This optimization was not present in the historical database; the

agent reasoned from first principles.

Case 3, a fine-tuning workload on TPU v5e-256, showed a workload where the top five most time-consuming operations were all collective communication operations. Despite a near-perfect duty cycle, the workload was communication-bound: the 4-way model parallelism in the baseline configuration was generating excessive all-reduce traffic. The Analyzer correctly identified this pattern and the Proposal Agent recommended halving model parallelism while doubling data parallelism, reducing the frequency and size of collective operations. The resulting configuration matched the production baseline. In the justification dialogue, the agent explained the trade-off between collective operation frequency and per-device memory footprint with enough specificity that the engineer could evaluate whether the reasoning was sound for her particular workload.

5.4.2 Case 4: Boundary Conditions and Compiler Validation

Case 4 is the exception that clarifies the rule. The task was to propose an ICI mesh for a workload migrating from a 16×16 (256-chip) to an 8×16 (128-chip) topology—a hardware topology change that invalidated the original configuration. The Analyzer correctly identified the primary bottleneck and the Proposal Agent generated three candidate configurations. Its top recommendation, however, failed: it prioritized step-time optimization without adequately accounting for the memory constraint that the smaller topology imposed. Halving sequence parallelism doubled activation memory per chip, triggering an out-of-memory error at compilation. The second proposal, which the agent had ranked lower, succeeded: it maintained high sequence parallelism, keeping activation memory within bounds, and achieved a 14.29% performance improvement.

This case demonstrates two things. First, the system's reasoning has a boundary: when hardware

constraints interact in ways not well-represented in the knowledge base, the agent can optimize for the wrong objective. This is a factual knowledge gap—the agent lacked explicit knowledge that for constrained topologies, memory capacity is a hard primary constraint that supersedes step-time optimization. Second, compiler-level cross-compilation provides a fail-fast mechanism that catches this kind of error before hardware is allocated. The compilation failure is not a system failure in any meaningful sense; it is the validation layer working as designed, filtering infeasible proposals before they consume engineering time. The engineer adopted Proposal 2, and the system identified the correct configuration within its batch of three.

5.4.3 Zero-Knowledge Stress Test

The zero-knowledge stress test used the same experimental setup as Case 6—an open-source Qwen-32B model on a new tpu7x platform—but deliberately withheld all documentation about the model architecture and hardware platform from the knowledge base. A senior performance engineer then reviewed and annotated the agent's output post-hoc, producing a line-by-line comparison between the agent's analysis and an expert's analysis of the same profiling data.

The high-level verdict is that the agent's reasoning procedure was sound but its conclusions were bounded by what it knew. It correctly identified the dominant bottleneck operation (copy.2700) as severely HBM-bound. It correctly flagged the all-gather operation as inefficient. It proposed fusing memory-bound operations to hide latency. The expert's annotation confirmed the bottleneck identification as 'technically correct' while noting that it lacked root-cause depth: the copy operation was not a simple data transfer but a tensor relayout resulting from mismatched input and output shapes, a distinction that changes what the correct fix is. The expert rejected the fusion suggestion

as inapplicable on tpu7x, where shared HBM bandwidth saturation makes latency hiding via fusion ineffective, and recommended Sparse Core offloading instead—a platform-specific optimization the agent could not have known without documentation about tpu7x's architectural constraints.

The annotation also identified a case where the agent analyzed an all-gather operation in isolation and suggested removing it if possible. The expert's note was precise: this particular all-gather is mandatory because it shards the embedding table, and removing it would break the model. The general principle—that unnecessary all-gathers should be eliminated—is correct. Its application to this specific operation required knowledge that the operation's function was non-optional in this workload, which the agent lacked.

What the stress test establishes is the structure of the gap between the agent and the expert. It is not a reasoning gap—the agent's analytical procedure is coherent and its general principles are sound. It is a knowledge gap, specific enough to be mapped to named facts: the architectural function of a particular operation, the memory management constraints of a specific hardware platform, the conditions under which a general optimization principle does not apply. Each gap corresponds to a document that could be added to the knowledge base. This is extensibility in the technically meaningful sense: the path from current performance to expert-level performance is not architectural redesign, but knowledge acquisition.

5.5 PROMPTS and the Dissertation's Argument

PROMPTS is not the dissertation's thesis. It is the thesis demonstrated at scale, in a domain where the human middleware problem is acute and the cost of failing to resolve it is high. The experts who work in ML system optimization are doing exactly what the introduction described: translating

between high-level intent—make this workload run faster, fit this model on this hardware—and low-level system representations, in the form of profiling traces that require years of expertise to read and configuration spaces that require hundreds of evaluations to search. The burden of that translation falls entirely on the engineer, and it is the rate-limiting step in the optimization workflow.

What PROMPTS demonstrates is that this translation is offloadable without being replaceable. The system does not determine which workload to optimize, set the performance target, or decide whether the agent's top recommendation is the right one for this deployment context. It performs the labor-intensive middle steps: reading the trace, identifying the bottleneck, retrieving relevant precedents, generating candidate configurations, and articulating the reasoning that connects evidence to proposal. The engineer remains responsible for the judgment that requires understanding what the optimization is for. The agent is responsible for the reasoning that requires understanding what the profiling data says.

This division maps directly onto the dissertation's definition of effective mediation: a system that reduces the mental work required to translate between what a human is trying to do and what the system needs to execute, without removing human agency. PROMPTS reduces the mental work of profiling trace analysis and configuration generation to the work of reviewing and evaluating a structured proposal. It does not remove the engineer from the process; it reassigns which steps she performs. The throughput gains—up to 434%—are a consequence of finding better configurations faster. The 87.5% adoption rate is evidence that the configurations the agent proposes are the configurations engineers would have eventually reached on their own, reached in under a minute rather than through hours of iterative search.

The five DEXAS principles map onto the system's architecture cleanly, and their presence explains why the system works where black-box methods do not. Decoding converts profiling traces

into structured diagnoses at the level of abstraction where engineering decisions are made. Embodied feedback coupling—adapted to the cognitive layer—ties each proposal tightly to the evidence that motivated it, so the engineer can evaluate the reasoning rather than just the conclusion. eXposed reasoning transparency makes the agent's intermediate steps inspectable, through the bottleneck report, the justification dialogue, and the Sharding Memory log. Adaptive input unification fuses KPI metrics, HLO profiles, and roofline data into a single diagnosis that no individual source could support. Scalable knowledge grounding separates the reasoning procedure from the domain knowledge, so the system extends to new hardware and models by knowledge addition rather than architectural change.

The zero-knowledge stress test makes this last point most sharply. The gap between a junior analyst and an expert is not primarily a gap in reasoning ability; it is a gap in accumulated knowledge about the domain. PROMPTS's architecture makes that gap explicit, locatable, and fillable. A system that fails opaquely teaches nothing. A system that fails at a specifiable knowledge boundary teaches exactly what to add next. This is what it means for a reasoning layer to be extensible in a practically meaningful sense: not that it handles every case correctly, but that when it does not, the failure is understandable and the fix is tractable.

Chapter 6: Discussion

Each of the five projects in this dissertation was motivated by a different problem. Surface I/O asked whether a passive surface could encode its own interaction vocabulary. Motionless Movement asked how feedback crosses the perceptual gap between intended and actual motion. Sustainflatable asked why a physically correct autonomous system required a simulation tool to be deployable. In-Situ Presenter asked what internal representation allows a system to act on inferred intent. PROMPTS asked what a reasoning agent can do with a profiling trace that an engineer would otherwise spend hours interpreting manually. The problems share a structure that only became visible in retrospect: in each case, the bottleneck was not computation or sensing or actuation, but translation — the gap between the representation a system operates on and the representation a human can reason with.

DEXAS names five ways that gap can be addressed, derived from patterns that recurred across the projects each time mediation succeeded or failed. The principles are a diagnostic vocabulary rather than a design checklist. They describe what was missing when mediation broke down, and what was present when it worked. This chapter examines what the five projects establish in common, where their findings diverge, and what they leave unresolved. Section 6.1 revisits the mediation spectrum from Chapter 1 with the full arc of the dissertation in view. Section 6.2 draws out three findings that cut across all five projects. Section 6.3 addresses the limits of the current work. Section 6.4 closes.

6.1 The Mediation Spectrum Revisited

Chapter 1 introduced the spectrum of mediation as a way of organizing the dissertation's projects: from physical interaction, where the body absorbs mediation below the threshold of conscious interpretation, to cognitive partnership, where a system must make its reasoning legible to a human who evaluates it deliberately. That framing was organizational. With the full arc of projects in view, the spectrum looks

different — less like a line and more like two independent dimensions that the projects helped identify.

At the physical end, the relevant representation is the action space: geometry, vibrotactile signature, the structural properties that make certain interactions feel natural and others feel wrong. Surface I/O's affordance layers are not visible to the user as a design artifact; they are felt as the path of least resistance when her finger moves. Motionless Movement's vibrotactile feedback does not get interpreted as a signal about joint angle; it arrives in the sensorimotor loop as the sense of movement itself, before interpretation begins. Mediation at this end works best when the representation dissolves into use — when there is nothing left for the user to read.

At the cognitive end, the relevant representation is the reasoning process: the sequence of steps by which the system moves from input to output, the evidence it considered, the inferences it drew. A designer configuring a Sustainflatable system needs to see this reasoning as temporal pressure curves and actuation previews. An engineer reviewing a PROMPTS recommendation needs to see it as a bottleneck report with an explicit justification chain. The interface that disappears into use at the physical end would, at the cognitive end, be a black box — a system acting on a representation its operator cannot inspect.

In-Situ Presenter sits at the seam between these two poles in a way the spectrum does not cleanly capture. The system's internal representation — the semantic alignment between script segments and slide elements — is never shown to the presenter during her talk. She sees what the system does with it: her video feed moves. When the movement is correct, the representation is invisible and that is fine. When the system misfires, she has no way to diagnose why, because the representation that drove the behavior is not surfaced. The failure is not at the physical end of the spectrum, where the representation was never meant to be seen, nor at the cognitive end, where it would be fully exposed. It is the consequence of a design choice about how much inference to surface, made without a clear

account of what the presenter needs to contest when things go wrong.

The more useful frame that emerges from the full arc of projects is not a single spectrum but two questions asked together: how explicitly must the system's representation be surfaced for the human to use the system effectively, and how much of the system's reasoning must the human be able to contest in order to maintain meaningful agency? Surface I/O and Motionless Movement sit low on both. PROMPTS sits high on both. Sustainflatable and In-Situ Presenter sit high on the first and reveal, through their failure modes, that the second cannot be ignored.

6.2 Three Cross-Cutting Findings

6.2.1 Intermediate Artifacts Are the Unit of Mediation

None of the five systems in this dissertation mediates by taking input and producing output. Each mediates by producing something in between — an artifact that sits between a representation the system can compute and a representation the human can use. Surface I/O's affordance layers encode the interaction pattern in geometry, sitting between the passive surface and the user's action. Motionless Movement's vibrotactile signal sits between the force sensor's measurement and the proprioceptive sense of movement, bridging a gap that neither could cross alone. The Sustainflatable design tool's simulation output sits between the pneumatic equations and the designer's judgment about what the configuration will do. PROMPTS's bottleneck report sits between the profiling trace and the configuration decision, converting dense execution data into something an engineer can act on.

What makes an intermediate artifact effective is not accuracy in isolation but fitness for the judgment it is meant to support. A simulation that outputs only a final actuation timestamp is technically accurate but practically useless, because the designer's judgment operates on the shape of the pressure

curve across time, not on its endpoint. A bottleneck classifier that outputs only a configuration recommendation is similarly impoverished — the engineer's judgment operates on whether the diagnosis is correct, and without the diagnosis she has no basis for that evaluation. The artifact's job is to make the system's reasoning legible at the level where human judgment is most productive, which is rarely the level of final outputs.

This observation cuts against a dominant tendency in AI system design, where optimizing for end-to-end task performance is the primary objective and the legibility of intermediate steps is a secondary concern at best. End-to-end training rewards correct final outputs regardless of whether the path to those outputs is inspectable; it can produce systems that are highly accurate and completely opaque in the same stroke. A system that achieves 90% accuracy through an inaccessible route may be less useful in practice than one that achieves 85% accuracy through a process the human can follow, correct, and learn from. Evaluation frameworks that do not measure the quality of intermediate artifacts will systematically undervalue the design choices that make reasoning layers work.

6.2.2 Opacity Is a System Failure, Not a Design Trade-off

In each of the five projects, the point at which mediation failed was the point at which the system's reasoning became inaccessible. Before the Sustainflatable design tool existed, the system's behavior was physically determined but epistemically invisible: a designer could assemble a correct configuration and have no way to predict when it would actuate or how sensitive it was to environmental variation. In In-Situ Presenter, misalignment between the semantic representation and the presenter's intent was observable in the video feed's behavior but not diagnosable, because the representation driving that behavior was never surfaced. In Case 4 of the PROMPTS evaluation, the agent optimized

for the wrong objective because its constraint reasoning about memory capacity under the new topology was not exposed at the level where the engineer could catch it; the compiler's cross-compilation step caught it instead, before hardware was wasted.

These failures share a structure. The system acts on a representation the human cannot see, and the human has no way to judge whether that action was correct until after its consequences arrive. Sarter and Woods documented this pattern in the Airbus A-320 context and called it automation surprise: pilots caught off guard by systems that had taken actions they could not anticipate because the systems' internal reasoning was opaque. The same structure recurs across domains wherever autonomous systems act on computed or inferred internal state, and the lesson from aviation — that opacity produces not just usability problems but safety failures — holds in attenuated form well outside cockpits.

Opacity is often treated as a design trade-off: transparency costs complexity, and some users prefer a cleaner interface even at the price of inspectability. But from the perspective of the human who must operate the system, opacity does not simplify anything. It transfers the complexity from the interface into the user's mental model: she must now maintain a representation of an opaque process whose behavior she cannot observe or predict. The cost of transparency falls on the system designer, once, at design time. The cost of opacity falls on every user, at every interaction, and compounds whenever the system does something unexpected. Redistributing that cost toward users is not a design trade-off; it is a design failure whose consequences are deferred.

6.2.3 The Human Role That Reasoning Layers Preserve

Each project reassigns the labor distribution between human and system without reassigning the judgment. Surface I/O reduces the labor of learning an input vocabulary; the user still decides what to do. Motionless Movement reduces the perceptual labor of interpreting force feedback; the user still controls the force application. The Sustainflatable design tool reduces the computational labor of reasoning about pneumatic physics; the designer still selects the configuration. In-Situ Presenter reduces the manual labor of repositioning the video feed during a live talk; the presenter still controls the content and pacing. PROMPTS reduces the analytical labor of trace reading and configuration generation; the engineer still decides which configuration to deploy and why.

This consistent division reflects something real about where value resides in each workflow. The work that gets offloaded to the system is work that is hard for humans because it requires processing representations humans are poorly suited to process — physical equations, dense execution traces, multimodal signal streams. The work that stays with the human is work that is hard for systems because it requires understanding what the task is for, what counts as good enough in this context, and what downstream consequences a given decision will have. That is not an arbitrary split; it reflects genuine comparative advantage.

What is less obvious is that the division is fragile. A system that automates more of the pipeline without keeping the intermediate artifacts legible does not just shift the labor distribution — it degrades the human's capacity to exercise the judgment that remains. An engineer who has never read a profiling trace because PROMPTS has always decoded it for her loses the ability to tell when PROMPTS's diagnosis is wrong. A presenter who has never manually positioned her video feed cannot recover gracefully when In-Situ Presenter misidentifies her intent. Preserving meaningful human agency in a

reasoning-layer architecture requires more than leaving the human a formal role in the pipeline. The intermediate artifacts must remain legible enough that she can exercise genuine judgment — including judgment about whether to trust the system's reasoning at all.

6.3 Limitations and Future Directions

6.3.1 Evaluation and Alignment of Agent Output

The evaluation frameworks across this dissertation are heterogeneous by necessity. Motionless Movement uses controlled psychophysical experiments; the Sustainflatable design tool is evaluated through design walkthroughs; In-Situ Presenter is evaluated against the ITalk-57 dataset; PROMPTS is evaluated through production workload matching and expert annotation. Each method is appropriate for its domain. None of them measures the legibility of intermediate artifacts directly, and the dissertation does not provide a measurement procedure for legibility despite arguing for its importance. That gap limits how far the claims can be generalized.

For agentic systems the problem is sharper. The 87.5% adoption rate for PROMPTS's top recommendation is evidence that its proposals are expert-aligned, but it says nothing about whether the reasoning that led to those proposals was sound, or whether it would remain sound in the cases where the proposal is wrong. An agent that reaches good conclusions through flawed reasoning is less trustworthy than its accuracy rate suggests. Reinforcement learning from human feedback offers a path toward more systematic alignment, but defining the reward signal for a diagnostic reasoning task is not straightforward. Reasoning faithfulness, evidence grounding, and calibrated confidence are all relevant properties, but none has a standard operationalization for open-ended diagnostic tasks. What a smooth reward function for this kind of agent would look like is genuinely unclear.

PROMPTS's knowledge base grows through manual addition of documents and cases but does not update from the agent's own operational experience. An agent that successfully diagnoses a novel bottleneck type should be able to record that diagnosis as a reusable precedent. One whose proposal fails compilation should update its constraint model for the topology involved. Engineers do this as a matter of course: each optimization attempt updates their mental model of the configuration space. PROMPTS does not close that loop automatically, and closing it raises questions that are not obvious to answer. A failure case on a specific model-hardware combination may reflect a genuine general constraint or an idiosyncratic interaction that should not propagate as a heuristic. An agent that updates indiscriminately from its own outputs accumulates noise alongside signal. Distinguishing between a case where the reasoning was correct and the proposal failed for unrelated reasons, a case where the reasoning was wrong in a way the compiler caught, and a case where the reasoning was wrong in a way that only became visible after deployment — each requires a different updating rule, and designing those rules is non-trivial.

The longer-term question is knowledge base maintenance over time: as hardware generations change the constraint landscape, as new model architectures introduce unfamiliar computational patterns, and as old optimizations become outdated precedents, how should the knowledge base evolve? A static base grows stale. One that grows without curation grows noisy. The right architecture for this problem in a fast-moving production domain is open, and will matter more as these systems accumulate months or years of deployment history.

6.3.2 From Sharding to General Optimization

PROMPTS's current scope is ICI-mesh sharding for TPU workloads, with batch size, rematerialization, and compiler flags held fixed. That narrowing was a deliberate design choice: isolating one variable makes the evaluation clean and the results interpretable. The framework is modular enough that extending it to joint optimization requires adding knowledge modules rather than rebuilding the agents, but the extension is not trivial. As the number of jointly optimized variables grows, so do the interactions among them, and those interactions place demands on the knowledge base that the current document-based representation may not support well.

Reasoning about sharding and batch size together requires the knowledge base to encode the dependency between them: optimal batch size given a sharding configuration depends on the per-device HBM capacity that sharding determines. Reasoning about rematerialization and compute-bound-ness together requires encoding the trade-off between recomputation cost and memory savings as a function of the current bottleneck type. These interactions are not intractable — experienced engineers reason about them routinely as part of their mental model of the optimization space — but encoding them in a form an agent can query reliably may require moving from a document knowledge base toward a structured knowledge graph, where constraints and trade-offs are represented as relationships rather than as passages in technical guides.

6.3.3 The Scope of DEXAS

DEXAS emerged from five projects in three domains, and its five principles describe requirements that proved necessary across those domains. The dissertation does not claim they are sufficient for all reasoning-layer applications, or equally relevant in every context. Embodied feedback coupling is most

directly applicable to systems that interact with the sensorimotor loop; its extension to the cognitive layer in PROMPTS is a meaningful adaptation, but it is an extension. Decoding is most relevant when the system's inputs are dense, domain-specific representations that require expert translation; it is less central when users are already working at a high level of abstraction.

More broadly, DEXAS describes properties of the mediating layer, not of the full human-system relationship over time. It says nothing about trust calibration: how a user develops appropriate confidence in a reasoning layer's outputs, and how that confidence should update when the system fails in ways the user did not anticipate. It says nothing about skill maintenance: whether users who rely on reasoning layers for translation tasks retain the ability to perform those tasks without assistance, and under what conditions that dependency becomes a vulnerability. It says nothing about accountability: who is responsible when a reasoning layer's recommendation leads to a bad outcome in a high-stakes setting. These are not peripheral concerns. They are the questions that will determine whether reasoning layers are deployed well or poorly as they become more capable and more consequential, and the current work was not positioned to answer them.

6.4 Closing

The problem this dissertation started from — that experts are consumed by the translation between what they are trying to do and what their systems need from them — is not new. It appears whenever the representations a system operates on and the representations a human reasons with are mismatched in scale, vocabulary, or level of abstraction. It appeared in early HCI, where command-line interfaces required users to learn formal syntax with no natural relationship to their goals. It appeared in the automation surprise literature, where pilots found that the systems they trusted were

acting on internal state they could not observe. It appears in contemporary ML engineering, where the gap between a researcher's intent and the configuration decisions that implement it has become a significant rate limiter on what the field can build.

What has changed is not the problem but the available tools. Language models that can synthesize domain knowledge into structured diagnoses, multimodal architectures that can unify heterogeneous signals into shared semantic representations, simulation environments that can render physical dynamics inspectable at the level of a designer's judgment — these make reasoning layers buildable in ways they were not a decade ago. The question has shifted from whether to build them to how: with what architecture, against what evaluation criteria, and with what account of the human role they are meant to support rather than replace.

DEXAS is a first attempt at a systematic answer to that how question. Its five principles name requirements that proved necessary across the projects in this dissertation, derived empirically from cases where mediation worked and cases where it did not. PROMPTS is a demonstration that all five can be instantiated simultaneously in a domain where the cost of not doing so is measurable in engineering hours and hardware utilization rates. Neither DEXAS nor PROMPTS is the last word. DEXAS is a framework built from five cases, and five cases do not settle what an adequate general theory would require. PROMPTS solves one well-scoped problem in a fast-moving domain, and the solutions that work for ICI-mesh sharding on TPUs will not translate unchanged to the next problem.

What the dissertation does establish is a way of framing the problem that makes it harder to avoid. A reasoning layer is not a solution to the human middleware problem. It is an acknowledgment that the problem is real, and a commitment to building systems that address it — that surface their intermediate reasoning rather than hiding it, that couple their outputs to the human's judgment rather than bypassing it, and that extend their capabilities through knowledge rather than through architectural

opacity. Whether that commitment holds as the systems become more capable, the reasoning layers thicker, and the human's window into them narrower — that is the question the work is designed to make unavoidable.

Bibliography

[1] Pattie Maes, Ben Shneiderman, and Jim Miller. Intelligent software agents vs. user-controlled direct manipulation: a debate. In *CHI'97 Extended Abstracts on Human Factors in Computing Systems*, pages 105–106. 1997.

[2] Yuran Ding, Nihar Sabnis, and Paul Strohmeier. Motionless movement: Towards vibrotactile kinesthetic displays. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pages 1–16, 2024.

[3] Yuran Ding, Craig Shultz, and Chris Harrison. Surface i/o: Creating devices with functional surface geometry for haptics and user input. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–22, 2023.

[4] Susan J Lederman and Roberta L Klatzky. Extracting object properties through haptic exploration. *Acta psychologica*, 84(1):29–40, 1993.

[5] Susan J Lederman and Roberta L Klatzky. Hand movements: A window into haptic object recognition. *Cognitive Psychology*, 19(3):342–368, July 1987.

[6] S. J. Lederman and R. L. Klatzky. Haptic perception: A tutorial. *Attention, Perception, & Psychophysics*, 71(7):1439–1459, October 2009.

[7] Don Norman. *The design of everyday things: Revised and expanded edition*. Basic books, 2013.

[8] Peter-Paul Verbeek. *What things do: Philosophical reflections on technology, agency, and design*. Penn State Press, 2005.

[9] Don Ihde. Philosophy of technology. In *Philosophical problems today: World and worldhood*, pages 91–108. Springer, 2004.

[10] J. Kevin O'Regan and Alva Noë. A sensorimotor account of vision and visual consciousness. *Behavioral and Brain Sciences*, 24(5):939–973, October 2001.

[11] Don Ihde. *Technology and the lifeworld*. Indiana Series in the Philosophy of Technology. Indiana University Press, Bloomington, MN, May 1990.

[12] Hiroshi Imamizu. Prediction of sensorimotor feedback from the efference copy of motor commands: A review of behavioral and functional neuroimaging studies. *Japanese Psychological Research*, 52(2):107–120, 2010.

[13] Pedro Lopes, Doăa Yüksel, François Guimbretière, and Patrick Baudisch. Muscle-plotter: An interactive system based on electrical muscle stimulation that produces spatial output. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 207–217, 2016.

[14] Pedro Lopes and Patrick Baudisch. Interactive systems based on electrical muscle stimulation. In *ACM SIGGRAPH 2017 Studio*, pages 1–2. 2017.

[15] Mahdi Azmandian, Mark Hancock, Hrvoje Benko, Eyal Ofek, and Andrew D Wilson. Haptic retargeting: Dynamic repurposing of passive haptics for enhanced virtual reality experiences. In *Proceedings of the 2016 chi conference on human factors in computing systems*, pages 1968–1979, 2016.

[16] Sousuke Nakamura. Challenges for Immersive Virtual Reality and Initiatives in the Motion-Less VR. *Journal of Japan Society for Fuzzy Theory and Intelligent Informatics*, 33(3):89–96, 2021.

[17] Richard Schmidt. *Motor control and learning 6th edition with web resource*. Human Kinetics, Champaign, IL, 6 edition, February 2018.

[18] A. Lecuyer, S. Coquillart, A. Kheddar, P. Richard, and P. Coiffet. Pseudo-haptic feedback: can isometric input devices simulate force feedback? In *Proceedings IEEE Virtual Reality 2000 (Cat. No.00CB37048)*, pages 83–90, 2000.

[19] Johan Kildal. Kooboh: Variable Tangible Properties in a Handheld Haptic-Illusion Box. In Poika Isokoski and Jukka Springare, editors, *Haptics: Perception, Devices, Mobility, and Communication*, Lecture Notes in Computer Science, pages 191–194, Berlin, Heidelberg, 2012. Springer.

[20] Johan Kildal. 3d-press: haptic illusion of compliance when pressing on a rigid surface. In *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction*, pages 1–8, 2010.

[21] Paul Strohmeier, Jesse Burstyn, Juan Pablo Carrascal, Vincent Levesque, and Roel Vertegaal. ReFlex: A Flexible Smartphone with Active Haptic Feedback for Bend Input. In *Proceedings of the TEI '16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '16, pages 185–192, New York, NY, USA, February 2016. Association for Computing Machinery.

[22] Paul Strohmeier, Seref Güngör, Luis Herres, Dennis Gudea, Bruno Fruchard, and Jürgen Steimle. bARefoot: Generating Virtual Materials using Motion Coupled Vibration in Shoes. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, pages 579–593, Virtual Event USA, October 2020. ACM.

[23] Nihar Sabnis, Dennis Wittchen, Courtney N Reed, Narjes Pourjafarian, Jürgen Steimle, and Paul Strohmeier. Haptic servos: Self-contained vibrotactile rendering system for creating or augmenting material experiences. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–17, 2023.

[24] Seongkook Heo, Jaeyeon Lee, and Daniel Wigdor. PseudoBend: Producing Haptic Illusions of Stretching, Bending, and Twisting Using Grain Vibrations. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 803–813, New York, NY, USA, October 2019. Association for Computing Machinery.

[25] Paul Strohmeier and Kasper Hornbæk. Generating haptic textures with a vibrotactile actuator. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 4994–5005, 2017.

[26] Geoff Cumming. The new statistics: Why and how. *Psychological Science*, 25(1):7–29, November 2013.

[27] Pierre Dragicevic. Fair statistical communication in hci. 2016.

[28] Mark D Weiser. Ubiquitous computing. In *ACM Conference on Computer Science*, volume 418, pages 197530–197680, 1994.

[29] Mark Weiser. What is pervasive computing? 2001.

[30] Mark Weiser. The computer for the 21st century. *ACM SIGMOBILE Mobile Computing and Communications Review*, 3(3):3–11, July 1999.

[31] Ivan Poupyrev, Nan-Wei Gong, Shiho Fukuhara, Mustafa Emre Karagozler, Carsten Schwesig, and Karen E. Robinson. Project Jacquard: Interactive Digital Textiles at Scale. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 4216–4227, New York, NY, USA, May 2016. Association for Computing Machinery.

[32] Sliman Bensmaïa and Mark Hollins. Pacinian representations of fine surface texture. *Perception & Psychophysics*, 67(5):842–854, July 2005.

[33] David T Blake, Steven S Hsiao, and Kenneth O Johnson. Neural coding mechanisms in tactile pattern recognition: the relative contributions of slowly and rapidly adapting mechanoreceptors to perceived roughness. *Journal of Neuroscience*, 17(19):7480–7489, 1997.

[34] David Arthur Burns, Roberta L. Klatzky, Michael A. Peshkin, and J. Edward Colgate. Spatial perception of textures depends on length-scale. In *2021 IEEE World Haptics Conference (WHC)*, pages 415–420, July 2021.

[35] Louise R. Manfredi, Hannes P. Saal, Kyler J. Brown, Mark C. Zielinski, John F. Dammann, Vicky S. Polashock, and Sliman J. Bensmaia. Natural scenes in tactile texture. *Journal of Neurophysiology*, 111(9):1792–1802, May 2014. Publisher: American Physiological Society.

[36] S Bolanowski, George Gescheider, R Verrillo, and C Checkosky. Four channels mediate the mechanical aspects of touch. *The Journal of the Acoustical Society of America*, 84:1680–94, December 1988.

[37] Craig Shultz, Daehwa Kim, Karan Ahuja, and Chris Harrison. TriboTouch: Micro-Patterned Surfaces for Low Latency Touchscreens. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, pages 1–13, New York, NY, USA, April 2022. Association for Computing Machinery.

[38] Michael Wiertlewski, José Lozada, and Vincent Hayward. The Spatial Spectrum of Tangential Skin Displacement Can Encode Tactual Texture. *IEEE Transactions on Robotics*, 27(3):461–472, June 2011. Conference Name: IEEE Transactions on Robotics.

[39] Qiuyu Lu, Tianyu Yu, Semina Yi, Yuran Ding, Haipeng Mi, and Lining Yao. Sustainflatable: harvesting, storing and utilizing ambient energy for pneumatic morphing interfaces. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–20, 2023.

[40] Joseph E Mercado, Michael A Rupp, Jessie YC Chen, Michael J Barnes, Daniel Barber, and Katelyn Procci. Intelligent agent transparency in human–agent teaming for multi-uxv management. *Human factors*, 58(3):401–415, 2016.

[41] Joseph B Lyons. Being transparent about transparency. In *AAAI Spring Symposium*, pages 48–53. AAAI Press Palo Alto, California, USA, 2013.

[42] Raja Parasuraman and Victor Riley. Humans and automation: Use, misuse, disuse, abuse. *Human factors*, 39(2):230–253, 1997.

[43] Nadine B Sarter and David D Woods. Team play with a powerful and independent agent: Operational experiences and automation surprises on the airbus a-320. *Human factors*, 39(4):553–569, 1997.

[44] Jennifer Mankoff. HCI and sustainability: a tale of two motivations. *Interactions*, 19(3):16–19, May 2012.

[45] Lining Yao, Ryuma Niiyama, Jifei Ou, Sean Follmer, Clark Della Silva, and Hiroshi Ishii. PneUI: pneumatically actuated soft composite materials for shape changing interfaces. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*, UIST '13, pages 13–22, New York, NY, USA, October 2013. Association for Computing Machinery.

[46] Jifei Ou, Mélina Skouras, Nikolaos Vlavianos, Felix Heibeck, Chin-Yi Cheng, Jannik Peters, and Hiroshi Ishii. aeroMorph - Heat-sealing Inflatable Shape-change Materials for Interaction Design. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 121–132, New York, NY, USA, October 2016. Association for Computing Machinery.

[47] Qiuyu Lu, Jifei Ou, João Wilbert, André Haben, Haipeng Mi, and Hiroshi Ishii. milliMorph – Fluid-Driven Thin Film Shape-Change Materials for Interaction Design. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 663–672, New York, NY, USA, October 2019. Association for Computing Machinery.

[48] Harpreet Sareen, Udayan Umapathi, Patrick Shin, Yasuaki Kakehi, Jifei Ou, Hiroshi Ishii, and Pattie Maes. Printflatables: Printing Human-Scale, Functional and Dynamic Inflatable Objects. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 3669–3680, New York, NY, USA, May 2017. Association for Computing Machinery.

[49] Junichi Yamaoka, Ryuma Niiyama, and Yasuaki Kakehi. BlowFab: Rapid Prototyping for Rigid and Reusable Objects using Inflation of Laser-cut Surfaces. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pages 461–469, New York, NY, USA, October 2017. Association for Computing Machinery.

[50] Junichi Yamaoka, Kazunori Nozawa, Shion Asada, Ryuma Niiyama, Yoshihiro Kawahara, and Yasuaki Kakehi. AccordionFab: Fabricating Inflatable 3D Objects by Laser Cutting and Welding Multi-Layered Sheets. In *Adjunct Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, UIST '18 Adjunct, pages 160–162, New York, NY, USA, October 2018. Association for Computing Machinery.

[51] Yiyue Luo, Kui Wu, Andrew Spielberg, Michael Foshey, Daniela Rus, Tomás Palacios, and Wojciech Matusik. Digital Fabrication of Pneumatic Actuators with Integrated Sensing by Machine Knitting. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, pages 1–13, New York, NY, USA, April 2022. Association for Computing Machinery.

[52] Saya Suzunaga, Yuichi Itoh, Yuki Inoue, Kazuyuki Fujita, and Takao Onoye. TuVe: A Shape-changeable Display using Fluids in a Tube. In *Proceedings of the International Conference on Advanced Visual Interfaces*, AVI '20, pages 1–9, New York, NY, USA, October 2020. Association for Computing Machinery.

[53] Carlos E. Tejada, Raf Ramakers, Sebastian Boring, and Daniel Ashbrook. AirTouch: 3D-printed Touch-Sensitive Objects Using Pneumatic Sensing. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, pages 1–10, New York, NY, USA, April 2020. Association for Computing Machinery.

[54] Liang He, Gierad Laput, Eric Brockmeyer, and Jon E. Froehlich. SqueezaPulse: Adding Interactive Input to Fabricated Objects Using Corrugated Tubes and Air Pulses. In *Proceedings of the Eleventh International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '17, pages 341–350, New York, NY, USA, March 2017. Association for Computing Machinery.

[55] Marynel Vázquez, Eric Brockmeyer, Ruta Desai, Chris Harrison, and Scott E. Hudson. 3D Printing Pneumatic Device Controls with Variable Activation Force Capabilities. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1295–1304, New York, NY, USA, April 2015. Association for Computing Machinery.

[56] Juri Fujii, Satoshi Nakamaru, and Yasuaki Kakehi. LayerPump: Rapid Prototyping of Functional 3D Objects with Built-in Electrohydrodynamics Pumps Based on Layered Plates. In *Proceedings of the Fifteenth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '21, pages 1–7, New York, NY, USA, February 2021. Association for Computing Machinery.

[57] Takafumi Morita, Yu Kuwajima, Ayato Minaminosono, Shingo Maeda, and Yasuaki Kakehi. HydroMod : Constructive Modules for Prototyping Hydraulic Physical Interfaces. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, pages 1–14, New York, NY, USA, April 2022. Association for Computing Machinery.

[58] Teng Han, Fraser Anderson, Pourang Irani, and Tovi Grossman. HydroRing: Supporting Mixed Reality Haptics Using Liquid Flow. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, UIST '18, pages 913–925, New York, NY, USA, October 2018. Association for Computing Machinery.

[59] Shan-Yuan Teng, Tzu-Sheng Kuo, Chi Wang, Chi-huan Chiang, Da-Yuan Huang, Liwei Chan, and Bing-Yu Chen. PuPoP: Pop-up Prop on Palm for Virtual Reality. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, UIST '18, pages 5–17, New York, NY, USA, October 2018. Association for Computing Machinery.

[60] Jasmine Lu, Ziwei Liu, Jas Brooks, and Pedro Lopes. Chemical Haptics: Rendering Haptic Sensations via Topical Stimulants. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, UIST '21, pages 239–257, New York, NY, USA, October 2021. Association for Computing Machinery.

[61] Kazunori Ogawa, Tomohiro Ikeda, and Yuichi Kurita. Unplugged Powered Suit for Superhuman Tennis. In *2018 12th France-Japan and 10th Europe-Asia Congress on Mechatronics*, pages 361–364, September 2018.

[62] Kazunori Ogawa, Chetan Thakur, Tomohiro Ikeda, Toshio Tsuji, and Yuichi Kurita. Development of a pneumatic artificial muscle driven by low pressure and its application to the unplugged powered suit. *Advanced Robotics*, 31(21):1135–1143, November 2017.

[63] Rachel A. Shveda, Anoop Rajappan, Te Faye Yap, Zhen Liu, Marquise D. Bell, Barclay Jumet, Vanessa Sanchez, and Daniel J. Preston. A wearable textile-based pneumatic energy harvesting system for assistive robotics. *Science Advances*, 8(34):eabo2418, August 2022.

[64] Philipp Rothemund, Alar Ainla, Lee Belding, Daniel J. Preston, Sarah Kurihara, Zhigang Suo, and George M. Whitesides. A soft, bistable valve for autonomous control of soft actuators. *Science Robotics*, 3(16):eaar7986, March 2018.

[65] Matteo Taffetani, Xin Jiang, Douglas P. Holmes, and Dominic Vella. Static bistability of spherical caps. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2213):20170910, May 2018.

[66] Lishuai Jin, Yueying Yang, Bryan O. Torres Maldonado, Sebastian David Lee, Nadia Figueroa, Robert J. Full, and Shu Yang. Ultrafast, Programmable, and Electronics-Free Soft Robots Enabled by Snapping Metacaps. *Advanced Intelligent Systems*, n/a(n/a):2300039, February 2023.

[67] Valkyrie Savage, Carlos Tejada, Mengyu Zhong, Raf Ramakers, Daniel Ashbrook, and Hyunyoung Kim. AirLogic: Embedding Pneumatic Computation and I/O in 3D Models to Fabricate Electronics-Free Interactive Objects. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, UIST '22, pages 1–12, New York, NY, USA, October 2022. Association for Computing Machinery.

[68] Qiuyu Lu, Haiqing Xu, Yijie Guo, Joey Yu Wang, and Lining Yao. Fluidic Computation Kit: Towards Electronic-free Shape-changing Interfaces. In *Proceedings of the 2023 CHI*

Conference on Human Factors in Computing Systems, CHI '23, pages 1–21, New York, NY, USA, April 2023. Association for Computing Machinery.

[69] Anoop Rajappan, Barclay Jumet, and Daniel J. Preston. Pneumatic soft robots take a step toward autonomy. *Science Robotics*, 6(51):eabg6994, February 2021.

[70] Daniel J. Preston, Philipp Rothemund, Haihui Joy Jiang, Markus P. Nemitz, Jeff Rawson, Zhigang Suo, and George M. Whitesides. Digital logic for soft devices. *Proceedings of the National Academy of Sciences*, 116(16):7750–7759, April 2019.

[71] Janet Beavin Bavelas, Nicole Chovil, Linda Coates, and Lori Roe. Gestures Specialized for Dialogue. *Personality and Social Psychology Bulletin*, 21(4):394–405, 1995.

[72] Seokmin Kang, Barbara Tversky, and John B Black. Coordinating Gesture, Word, and Diagram: Explanations for Experts and Novices. *Spatial Cognition & Computation*, 15(1):1–26, 2015.

[73] Randi A Engle. Not Channels But Composite Signals: Speech, Gesture, Diagrams and Object Demonstrations Are Integrated in Multimodal Explanations. In *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society*, pages 321–326. Routledge, Routledge, 2022.

[74] Susan Goldin-Meadow and Martha Wagner Alibali. Gesture's Role in Speaking, Learning, and Creating Language. *Annual Review of Psychology*, 64(1):257–283, 2013.

[75] Cornelia Müller, Alan Cienki, Ellen Fricke, Silva Ladewig, David McNeill, and Sedinha Teßendorf. Body-Language-Communication. *An International Handbook on Multimodality in Human Interaction*, 1(1):131–232, 2013.

[76] Paul Chandler and John Sweller. The split-attention effect as a factor in the design of instruction. *British Journal of Educational Psychology*, 62(2):233–246, 1992.

[77] Margot Van Wermeskerken, Susanna Ravensbergen, and Tamara Van Gog. Effects of instructor presence in video modeling examples on attention and learning. *Computers in Human Behavior*, 89:430–438, 2018.

[78] Richard E Mayer and Roxana Moreno. Nine ways to reduce cognitive load in multimedia learning. *Educational psychologist*, 38(1):43–52, 2003.

[79] Jaeho Jeon, Seongyong Lee, and Hohsung Choe. Enhancing efl pre-service teachers' affordance noticing and utilizing with the synthesis of qualitative evidence strategies: An exploratory study of a customizable virtual environment platform. *Computers & Education*, 190:104620, 2022.

[80] Philip J. Guo, Juho Kim, and Rob Rubin. How video production affects student engagement: an empirical study of mooc videos. In *Proceedings of the First ACM Conference on Learning @ Scale Conference*, L@S '14, page 41–50, New York, NY, USA, 2014. Association for Computing Machinery.

[81] Zoom Video Communications, Inc. Sharing Slides As a Virtual Background. https://support.zoom.us/hc/en-us/articles/360046912351-Sharing-slides-as-a-Virtual-Background, August 2022. Feature introduced in Zoom version 5.2.0.

[82] Microsoft Support. Presenting With Cameo, 2023. Accessed: 2024-09-08.

[83] John Sweller. Cognitive load theory, learning difficulty, and instructional design. *Learning and instruction*, 4(4):295–312, 1994.

[84] Michael E Ellis. Perceived proxemic distance and instructional videoconferencing: Impact on student performance and attitude. 1992.

[85] Jian Liao, Adnan Karim, Shivesh Singh Jadon, Rubaiat Habib Kazi, and Ryo Suzuki. Realitytalk: Real-Time Speech-Driven Augmented Presentation for Ar Live Storytelling. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, pages 1–12, 2022.

[86] Ken Perlin, Zhenyi He, and Karl Rosenberg. Chalktalk: A Visualization and Communication Language-As a Tool in the Domain of Computer Science Education. *ArXiv Preprint ArXiv:1809.07166*, 2018.

[87] Yi Zhang, Ke Xu, Zhongling Pi, and Jiumin Yang. Instructor's Position Affects Learning From Video Lectures in Chinese Context: An Eye-Tracking Study. *Behaviour & Information Technology*, 41(9):1988–1997, 2022.

[88] Ray Smith and Google. Tesseract OCR: An Open-Source Optical Character Recognition Engine, 2005. Accessed: 2024-09-09.

[89] OpenAI. GPT-4: OpenAI's Multimodal Large Language Model, 2023. Accessed: 2024-09-09.

[90] Google. MediaPipe. https://developers.google.com/mediapipe, 2023. Accessed: 2023.

[91] Leah Findlater and Joanna McGrenere. A comparison of static, adaptive, and adaptable menus. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 89–96, 2004.

[92] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.

[93] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[94] Eric Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 159–166. ACM, 1999.

[95] James E Allen, Curry I Guinn, and Eric Horvitz. Mixed-initiative interaction. *IEEE Intelligent Systems and their Applications*, 14(5):14–23, 1999.

[96] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

[97] Xuanlei Zhao, Bin Jia, Haotian Zhou, Ziming Liu, Shenggan Cheng, and Yang You. Hetegen: Efficient heterogeneous parallel inference for large language models on resource-constrained devices. *Proceedings of Machine Learning and Systems*, 6:162–172, 2024.

[98] Shenggui Li, Hongxin Liu, Zhengda Bian, Jiarui Fang, Haichen Huang, Yuliang Liu, Boxiang Wang, and Yang You. Colossal-ai: A unified deep learning system for large-scale parallel training. In *Proceedings of the 52nd International Conference on Parallel Processing*, pages 766–775, 2023.

[99] Lianmin Zheng, Zhuohan Li, Hao Zhang, Yonghao Zhuang, Zhifeng Chen, Yanping Huang, Yida Wang, Yuanzhong Xu, Danyang Zhuo, Eric P Xing, et al. Alpa: Automating inter-and {Intra-Operator} parallelism for distributed deep learning. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 559–578, 2022.

[100] Yuanzhong Xu, HyoukJoong Lee, Dehao Chen, Blake Hechtman, Yanping Huang, Rahul Joshi, Maxim Krikun, Dmitry Lepikhin, Andy Ly, Marcello Maggioni, et al. Gspmd: general and scalable parallelization for ml computation graphs. *arXiv preprint arXiv:2105.04663*, 2021.

[101] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, 2020.

[102] Sami Alabed, Daniel Belov, Bart Chrzaszcz, Juliana Franco, Dominik Grewe, Dougal Maclaurin, James Molloy, Tom Natan, Tamara Norman, Xiaoyue Pan, et al. Partir: Composing spmd partitioning strategies for machine learning. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*, pages 794–810, 2025.

[103] Zhihao Jia, Matei Zaharia, and Alex Aiken. Beyond data and model parallelism for deep neural networks. *Proceedings of Machine Learning and Systems*, 1:1–13, 2019.

[104] Xupeng Miao, Yujie Wang, Youhe Jiang, Chunan Shi, Xiaonan Nie, Hailin Zhang, and Bin Cui. Galvatron: Efficient transformer training over multiple gpus using automatic parallelism. *arXiv preprint arXiv:2211.13878*, 2022.

[105] Xingyou Song, Qiuyi Zhang, Chansoo Lee, Emily Fertig, Tzu-Kuo Huang, Lior Belenki, Greg Kochanski, Setareh Ariafar, Srinivas Vasudevan, Sagi Perel, et al. The vizier gaussian process bandit algorithm. *arXiv preprint arXiv:2408.11527*, 2024.

[106] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Elliot Karro, and D. Sculley, editors. *Google Vizier: A Service for Black-Box Optimization*, 2017.

[107] Taegeon Um, Byungsoo Oh, Minyoung Kang, Woo-Yeon Lee, Goeun Kim, Dongseob Kim, Youngtaek Kim, Mohd Muzzammil, and Myeongjae Jeon. Metis: Fast automatic distributed

training on heterogeneous {GPUs}. In *2024 USENIX Annual Technical Conference (USENIX ATC 24)*, pages 563–578, 2024.

[108] Foteini Strati, Zhendong Zhang, George Manos, Ixeia Sánchez Périz, Qinghao Hu, Tiancheng Chen, Berk Buzcu, Song Han, Pamela Delgado, and Ana Klimovic. Sailor: Automating distributed training over dynamic, heterogeneous, and geo-distributed clusters. In *Proceedings of the ACM SIGOPS 31st Symposium on Operating Systems Principles*, pages 204–220, 2025.

[109] Wei Wen, Quanyu Zhu, Weiwei Chu, Wen-Yen Chen, and Jiyan Yang. Cubicml: Automated ml for large ml systems co-design with ml prediction of performance. *arXiv preprint arXiv:2409.04585*, 2024.

[110] JooHyoung Cha, Munyoung Lee, Jinse Kwon, Jubin Lee, Jemin Lee, and Yongin Kwon. ML^2 Tuner: Efficient code tuning via multi-level machine learning models. *arXiv preprint arXiv:2411.10764*, 2024.

[111] Sana Damani, Siva Kumar Sastry Hari, Mark Stephenson, and Christos Kozyrakis. Warpdrive: An agentic workflow for ninja gpu transformations. In *NeurIPS Mach. Learn. Syst. Workshop*, 2024.

[112] Charles Hong, Sahil Bhatia, Alvin Cheung, and Yakun Sophia Shao. Autocomp: Llm-driven code optimization for tensor accelerators. *arXiv preprint arXiv:2505.18574*, 2025.

[113] Shubham Gandhi, Manasi Patwardhan, Lovekesh Vig, and Gautam Shroff. Budgetmlagent: A cost-effective llm multi-agent system for automating machine learning tasks. In *Proceedings of the 4th International Conference on AI-ML Systems*, AIMLSystems '24, New York, NY, USA, 2025. Association for Computing Machinery.

[114] Zexi Liu, Jingyi Chai, Xinyu Zhu, Shuo Tang, Rui Ye, Bo Zhang, Lei Bai, and Siheng Chen. Ml-agent: Reinforcing llm agents for autonomous machine learning engineering. *arXiv preprint arXiv:2505.23723*, 2025.

[115] Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

[116] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

[117] Paras Jain, Ajay Jain, Aniruddha Nrusimha, Amir Gholami, Pieter Abbeel, Joseph Gonzalez, Kurt Keutzer, and Ion Stoica. Checkmate: Breaking the memory wall with optimal tensor rematerialization. *Proceedings of Machine Learning and Systems*, 2:497–511, 2020.

[118] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

[119] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, et al. {TVM}: An automated {End-to-End} optimizing compiler for deep learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 578–594, 2018.

[120] Philippe Tillet, Hsiang-Tsung Kung, and David Cox. Triton: an intermediate language and compiler for tiled neural network computations. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, pages 10–19, 2019.

[121] Amit Sabne. Xla: Compiling machine learning for peak performance, 2020.