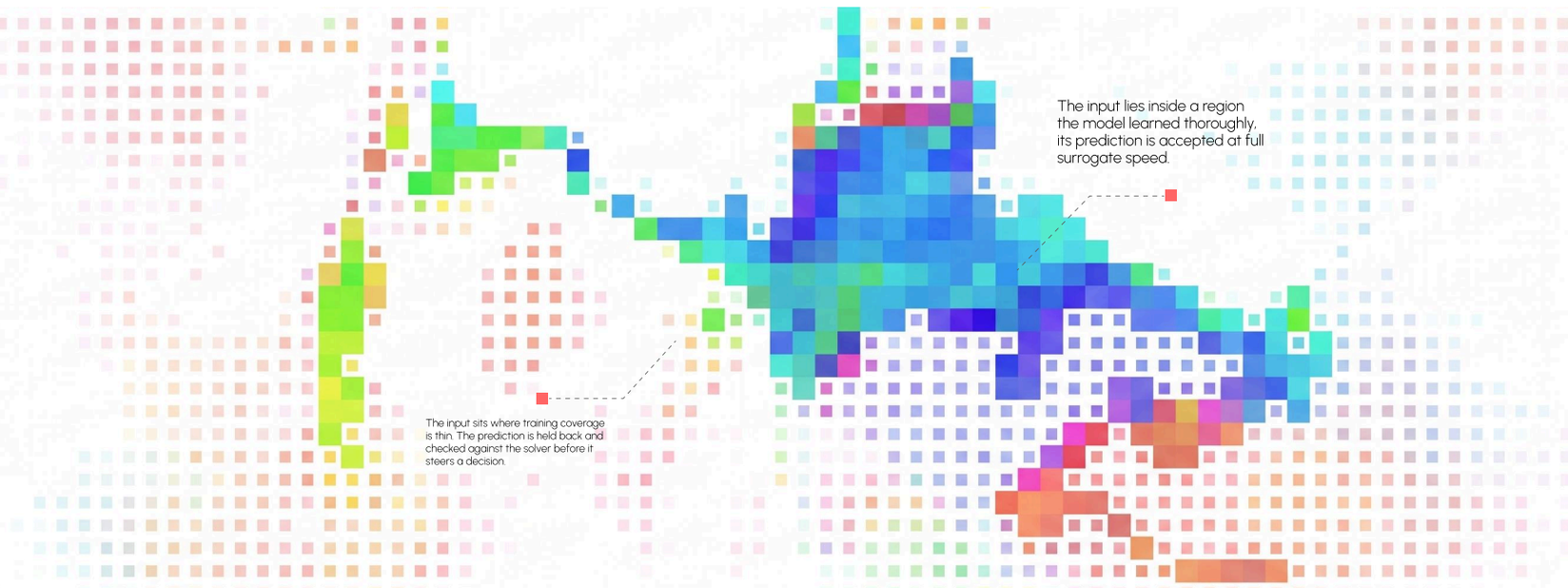
Case Study - Software & Models

WHEN A DESIGN SEARCH PUSHES A SURROGATE MODEL PAST WHAT IT LEARNED



A surrogate model is measured on held-out cases and posts errors of a few percent, comfortably inside tolerance. On that result it goes into a design workflow.

An optimization loop then does what it was built to do and searches configurations the training never covered. There the surrogate's error climbs into the tens of percent, while it keeps answering in the same form it used where it was accurate.

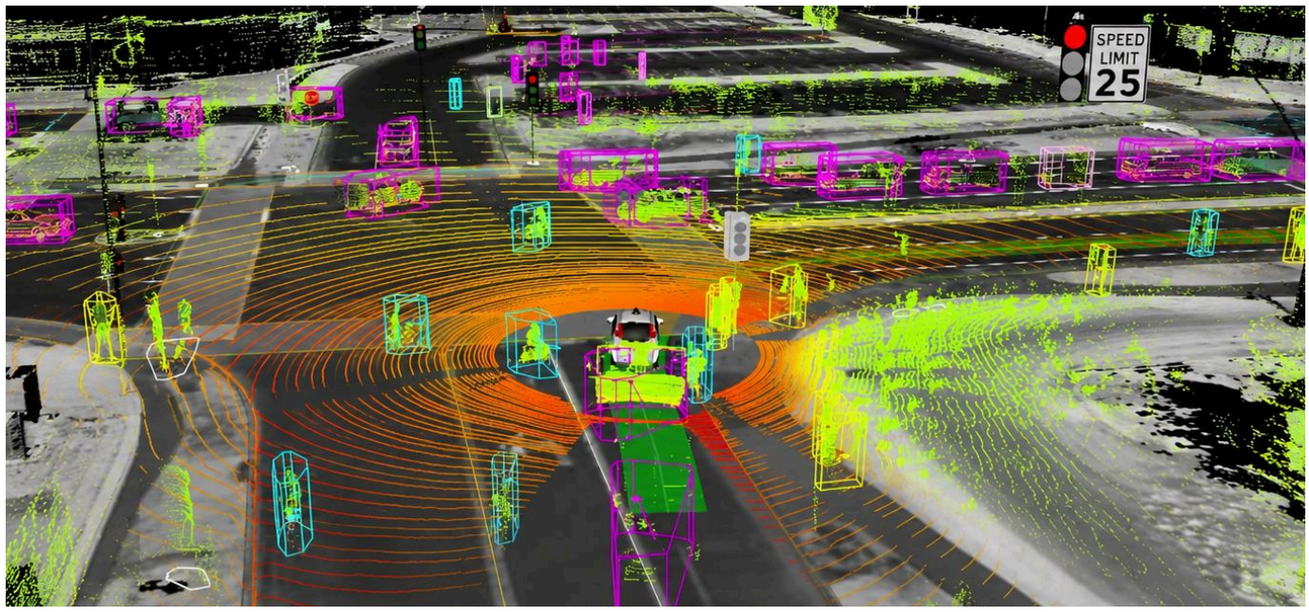
The model is most reliable where it was tested and least reliable where a design search spends its time, and its output gives no sign of which is which. That inversion is the subject of this case study.

WHY ENGINEERING TEAMS ADOPT SURROGATES

A high-fidelity simulation of fluid flow, structural stress or thermal behavior can take hours or days for a single configuration. That cost rules out whole categories of work. You cannot sweep thousands of variants, optimize a geometry, or run a fast control loop when each evaluation takes a day.

A surrogate removes the barrier. It learns to approximate the simulation from the solver's own outputs, then reproduces them in milliseconds. A physics-informed surrogate goes further, folding the governing equations into training so its answers are pulled toward physical validity rather than fitted only to points.

An infeasible design sweep becomes an afternoon's work, and decisions follow from it: this geometry over that one, this operating point judged safe.
The value of the tool is that it lets engineers explore widely and cheaply. Wide exploration is also what carries the surrogate out of the territory where it can be believed.



WHERE THE APPROXIMATION RUNS OUT

A surrogate learns the simulation across the conditions its training data represents. Inside that region it interpolates among cases it effectively saw, and it can be excellent.

Outside it, nothing anchors the approximation. Studies of surrogate models report relative errors of a few percent inside the training regime rising into the twenty to thirty percent range in extrapolation. Embedded physics helps less than expected. Research finds the extrapolation error of a physics-informed surrogate often remains about as high as a plain network's once the input leaves the sampled region.

A design optimization is an extrapolation engine by nature. Its job is to search past familiar configurations toward novel ones, which draws it systematically toward the regions the training data did not cover.

The surrogate does not decline the unfamiliar query. It returns an answer with the same form and apparent authority as one from the middle of its training region. An optimum the search converges on can be an artifact of surrogate error in a region where the model was never reliable.

WHY CHECKING EVERY ANSWER DEFEATS THE PURPOSE

The obvious defense is to verify against the real simulation, and it collides with the reason the surrogate exists.

It was built because the solver is too slow to run at the scale required. Validating every evaluation against the solver surrenders the speedup and restores the cost the surrogate was meant to remove. At the scale where surrogates pay off, meaning thousands or millions of evaluations inside a sweep, per-point validation is impractical by construction.

Inspecting the output does not help either. Researchers working with surrogate fields note that a prediction can look entirely plausible and still be physically wrong. An extrapolation error presents as a smooth, reasonable-looking answer that happens to be off.

So the boundary between dependable interpolation and unreliable extrapolation decides whether a design decision is sound, and it is invisible in anything the surrogate reports.

WHAT THE SAMPLING PLAN COVERED

There are two separate questions here, and the first one is answerable before anyone looks at the model's behavior.

A surrogate's training set is a sampling plan. Someone chose which parameter combinations to run through the expensive solver, and that choice defines the regime the surrogate will be able to approximate. Squint Vision Studio reports the composition of a dataset, including how samples are distributed across it, which turns the sampling plan into something a team can examine rather than assume.

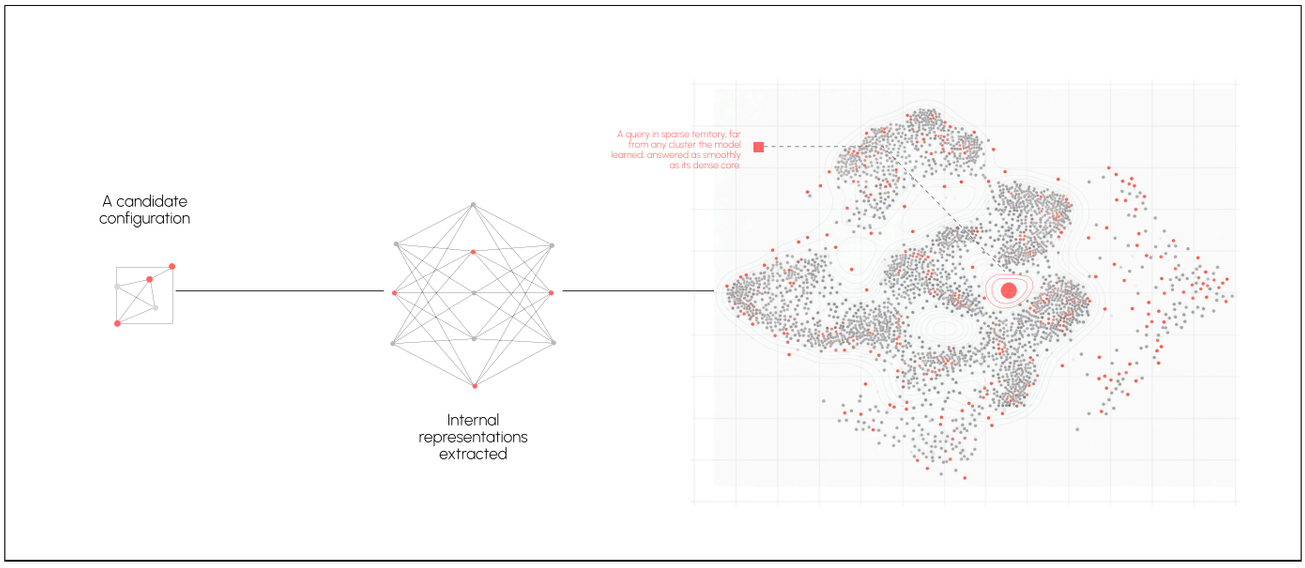
The useful comparison is against the space the optimizer will actually search. Where the design space extends well beyond the sampled regime, the mismatch is visible before a single evaluation runs, and the response is to commission more solver cases rather than to filter results afterward. That is a cheaper fix than discovering the gap through a bad design decision.

READING THE MODEL'S INTERNAL STRUCTURE

The second question concerns how the model organized what it learned, and it needs the model rather than the dataset.

Vision Studio extracts the internal representations the network builds as it processes an input, then clusters them to show how the training data sits inside the model. Squint's published work demonstrates this on image classifiers, where clusters form around classes and regions whose prediction accuracy falls below a chosen threshold are treated as ambiguous.
A surrogate predicts a continuous quantity rather than choosing between classes, so there are no class clusters to separate. What the representation space can still show is which inputs the model has dense, consistent experience of and which sit in sparse territory where nothing constrains its output.

This matters because coverage and reliability are not the same thing. A sampling plan can look adequate while the model has still failed to learn a region well, and the internal view catches that where a dataset summary cannot. The characterization comes from the structure of what the model learned rather than from any confidence value it reports, which is what allows it to identify an extrapolation the surrogate would have answered smoothly.



FILTERING THE OPTIMIZATION LOOP

Vision Studio exports the analysis as a watchdog, a lightweight decision layer that runs alongside the surrogate rather than replacing it. Its job is to decide when the base model can be trusted and when further processing is worth the cost.

Each query takes one of three paths.

A point inside the well-supported region takes the fast path. The evaluation is accepted at full surrogate speed, and the great majority of queries in a typical sweep fall here.

A point in sparse or overlapping territory is not accepted at face value. In Squint's published pipeline this path escalates to a heavier model or to human review. In a design workflow the equivalent moves are available: route that point to the real solver for confirmation, or exclude it from the decision.

A point unlike anything in the training data takes the review path. It is retained for analysis rather than acted on, because it marks a region the surrogate was never trained to cover and where additional solver runs may be worth commissioning.

The speedup is preserved where the surrogate is sound. The consequential extrapolations, the few that would otherwise steer the result, are the ones stopped and checked.

THE FAILURE THIS LEAVES OPEN

Mapping the input boundary catches a surrogate being used outside the conditions it learned. It does not catch a surrogate that is wrong inside its training region because the physics it was given was misspecified. That is a distinct failure with its own literature, in which a low training loss on correct-looking inputs does not guarantee a correct solution.

The map speaks to where the model is being asked to operate, not to the correctness of the equations it was taught.

The map is also only as good as the training data behind it. If the sampled regime is unrepresentative of the design space being searched, regions that appear well supported may not be.

WHAT CHANGES FOR THE DESIGN TEAM

A surrogate governed only by its test-set accuracy is trusting a number measured where the model is strong to vouch for its behavior where a search will actually take it.

Mapping where the approximation holds turns an invisible boundary into a filter. The surrogate keeps the speed that justifies its existence on the evaluations it can support. The extrapolations it cannot support are stopped before they harden into engineering decisions. For a tool whose purpose is to be pushed cheaply into the unexplored, knowing the line past which it can no longer be trusted is what makes it safe to push at all.

This case study describes a representative scenario. The interpolation and extrapolation error behavior, the limited extrapolation benefit of embedded physics, and the impracticality of full solver validation at scale are drawn from published research on physics-informed and data-driven surrogate models. Squint Vision Studio's capabilities, including embedding extraction, clustering-based region mapping and exportable watchdog modules, are described as documented by Squint Cognition rather than measured from this scenario.