THE DECISION VALUE OF PERCEPTION COMPUTE

Anonymous authors
Paper under double-blind review

ABSTRACT

Adaptive perception spends extra computation on inputs where perception is expected to improve. When perception feeds a downstream decision system, a better perception output need not produce a better decision. We define the *decision value* of perception compute as the change in downstream loss from escalating an input from a cheap to an expensive perception mode. Because this value can be negative, the allocation of perception compute should be judged against a budget-constrained decision oracle, with uniform full-fidelity inference as a baseline rather than an upper bound. We introduce DEEP (**Decision Evaluation for Escalated Perception**), a benchmark that scores pre-escalation allocators against this oracle under selection, latency and energy budgets, charging each allocator for its own computation. With deployed monocular geometry on KITTI and nuScenes, we find that 34–54% of the escalations that change downstream loss make it worse; harmful escalations also occur for the published PDM-Closed planner, evaluated open-loop on nuPlan with real detector outcomes. On nuScenes, perception-level gain frequently disagrees in sign with decision value. This mismatch has practical consequences: choosing among fixed deployable signals by missed-object perception gain rather than by decision value reduces realized test decision gain by 7.4% of the all-cheap loss on average. Learned allocators recover part of the oracle's value by finding beneficial escalations but select nearly as much harm as random, and once their own computation is charged at a 20% latency budget, only the lightweight routers, at about 3.5% of a full detector pass, still beat random.

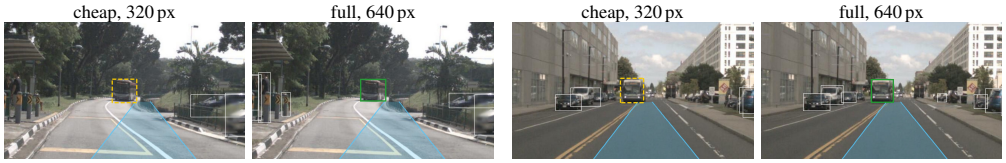
1 INTRODUCTION

Adaptive perception is motivated by a simple trade-off: more computation can improve perception, but it need not be spent on every input. Existing methods therefore try to identify where additional computation is worthwhile, using signals such as uncertainty, difficulty, criticality, or expected perception gain (Weiss et al., 2013; Heo et al., 2022; Kang et al., 2022; Liu et al., 2022). Selective routing between weak and strong detectors takes this idea further, sending an input to the expensive model only when it is expected to improve prediction quality, while recognizing that the stronger model need not be better on every input (Qiu et al., 2024; Geng et al., 2026). In these approaches, the value of additional perception compute is evaluated through the perception improvement it provides.

This criterion is incomplete when perception feeds a downstream decision system. A more expensive detector may recover a missed object without changing the downstream action, recover one that changes a safety-critical action, or add a false positive that triggers an unnecessary response. Figure 1 shows two nuScenes frames in which the full mode recovers the same kind of missed object, a bus: the braking controller decelerates correctly in one frame and brakes hard for a bus outside its path in the other. What ultimately determines whether additional compute is worthwhile is therefore its effect on the downstream decision, for which perception gain is only an indirect signal. We call this the *decision value* of perception compute: the change in downstream loss from escalating an input from a cheap to an expensive perception mode, positive when the loss decreases.

Such cases are not isolated. Across KITTI (Geiger et al., 2012) and nuScenes (Caesar et al., 2020), a substantial fraction of the escalations that change downstream loss make it worse, also for a published planner, and different downstream systems can assign opposite values to the same perception change. Decision value is therefore not determined by perception improvement alone, nor is it intrinsic to the perception model. Because it can be negative, uniform full-fidelity inference, which absorbs harmful escalations together with beneficial ones, is a practical baseline when the budget admits it,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) escalation harms: keep speed → hard brake **(b)** escalation helps: keep speed → decelerate

Figure 1: **Better detection need not improve the decision.** Two nuScenes frames under the same braking controller and monocular geometry. In both, the full mode recovers a bus that the cheap mode misses (green box; dashed yellow: the missed reference object); the shaded band is the controller's driving corridor on the road. **(a)** The recovered bus lies outside the corridor, but the monocular lift places it inside, and the controller brakes hard. **(b)** The recovered bus lies inside the corridor, and the controller decelerates. Both improve detection, yet their decision values have opposite signs.

not an upper bound on allocation; the reference for allocation quality is a *budget-constrained decision oracle* that spends the budget on the most valuable escalations and skips harmful ones. This oracle is not deployable, because decision value is known only after both modes have run and their decisions have been evaluated against the scene: a deployable allocator must decide before escalation from the input, the cheap output and available context, and pay for whatever computation it uses.

We introduce DEEP (**Decision Evaluation for Escalated Perception**), a benchmark for this pre-escalation allocation problem. DEEP scores allocators by the decision value they realize relative to the decision oracle under selection, latency, and energy budgets, while charging each allocator for its own computation. Its core track uses detection-based driving pipelines on KITTI and nuScenes, and its external track uses published planners on nuPlan. This paper makes three contributions.

- We define the decision value of perception compute and, because it is signed, measure allocation quality against a budget-constrained decision oracle, with uniform full fidelity as a baseline rather than an upper bound.
- We introduce DEEP, which evaluates pre-escalation allocation against the decision oracle under selection and measured compute budgets, including the allocator's own computation cost. Because perception outputs, decision values and costs are cached, evaluating a new allocator requires only its pre-escalation scores and, for measured budgets, its own measured cost, not rerunning the perception models.
- We find that decision value is sign-varying, consumer-dependent and not determined by perception-level improvement alone, that ranking allocators by perception gain can select worse ones, and that learned allocators capture part of the beneficial value but retain their advantage under measured budgets only at low overhead relative to a full detector pass.

2 Related Work

Adaptive inference and task-aware computation. Adaptive inference varies computation across inputs. Dynamic model selection trades accuracy against computation per input (Weiss et al., 2013), and real-time detectors adapt image scale to each image's sensitivity to downscaling, prioritize safety-critical regions (Heo et al., 2022; Kang et al., 2022; Liu et al., 2022), or route LiDAR frames through shallow or deep processing by scene complexity (Kim & Park, 2026). Robot policies likewise condition computation on the task state (Li et al., 2026; Pei et al., 2026; Chun et al., 2025). These methods are evaluated by retained perception accuracy or task success at reduced cost; DEEP instead values an escalation by its marginal effect on the loss of a separate decision system.

Selective offloading and routing. Structurally, the closest work routes inputs between a weak and a strong detector. Adaptive Feeding classifies each image as easy or hard and sends it to a fast or an accurate detector (Zhou et al., 2017). ORIC defines a per-image offloading reward as the gain in contextual mAP from replacing a weak detector's output with a strong detector's, estimates it from the weak detector's results alone at a cost of about 0.5 ms, and offloads images under a budget (Qiu et al., 2024). Budget-adaptive routing predicts from raw pixels whether swapping in the strong detector's output on a frame raises dataset-wide AP, so that the weak pass can be skipped, and charges the estimator to the compute budget (Geng et al., 2026). Both works find that the strong detector is

not better on every input and that reward-guided routing can exceed the strong detector's mAP; within a single network, additional depth can likewise turn a correct prediction into an error (Kaya et al., 2019). DEEP shares this allocation structure: a signed per-input value, estimated before escalation and selected under a budget. What differs is the value. In these detector-routing methods it is the change in detection quality; in DEEP it is the change in the loss of a downstream decision system. The two are not interchangeable: on nuScenes, four per-input perception-loss gains disagree in sign with decision value on 25–51% of the frames where both are non-zero, and the same perception change can have opposite values for two downstream systems. DEEP also measures whether perception- and decision-level evaluation select different allocators, how much of the oracle each recovers, and whether its advantage survives its own cost. Routing between language models (Ong et al., 2025) and selective prediction (Geifman & El-Yaniv, 2017; Galil et al., 2023) make per-input choices judged by the quality of a model's own output, and cost-sensitive offloading (Moothedath et al., 2026) and learning to defer (Mozannar & Sontag, 2020) trade prediction error against offloading or deferral cost; in each, utility is defined on the prediction rather than by the loss of a downstream decision system.

Planning-aware perception evaluation. Another line of work evaluates perception by its downstream consequences rather than by task-agnostic accuracy. PKL scores a detector by the divergence between a learned planner's outputs given reference and given detected objects (Philion et al., 2020), planning-aware metrics evaluate detection and prediction by their effect on downstream planning (Ivanovic & Pavone, 2022), TIP models the planner as an expected-utility maximizer and analyzes how perception noise affects its objective (Li & Yang, 2023), and task-aware risk estimation assesses the risk a detected perception failure poses to the motion plan (Antonante et al., 2023). DEEP shares this premise but asks a different question. These methods score a perception output or failure once it is available; DEEP asks for the marginal downstream value of one additional computation on one input, under a budget, and requires the allocator to act before it runs.

Value of information and computation. Valuing information or computation by its effect on decisions is classical. Information value theory prices the resolution of uncertainty by the decisions it improves (Howard, 1966), rational metareasoning defines the value of a computation as the expected improvement in decision quality it yields, net of its cost (Russell & Wefald, 1991; Lieder & Griffiths, 2017), active perception chooses sensing actions for the task at hand (Bajcsy, 1988), including perceptual attention in robot driving (Reece & Shafer, 1995), and selective perception runs the perceptual analyses whose expected value of information justifies their cost (Oliver & Horvitz, 2005). These ideas motivate our formulation, which operationalizes them for cheap-to-full escalations of learned perception on a fixed input, with the realized change in downstream loss as the value, a budget-constrained oracle as the reference and an allocator that commits before escalation. A negative value does not contradict the non-negative expected value of information: V_i^q is realized on a single input, the full output replaces rather than supplements the cheap one, and it is consumed by a fixed decision rule rather than by a decision maker that optimally combines both.

3 Decision Value of Perception Compute

Let $\mathbf{x}_i$ be an input with reference scene state $\mathbf{s}_i$. The cheap and full perception modes produce $\mathbf{z}_i^{\mathsf{c}} = f_{\mathsf{c}}(\mathbf{x}_i)$ and $\mathbf{z}_i^{\mathsf{f}} = f_{\mathsf{f}}(\mathbf{x}_i)$, respectively. A downstream system q consists of a decision rule π_q and a downstream loss $\mathcal{L}_q$. We define the *decision value* of escalating input i as

$$V_i^q = \mathcal{L}_q(\pi_q(\mathbf{z}_i^{\mathsf{c}}), \mathbf{s}_i) - \mathcal{L}_q\big(\pi_q(\mathbf{z}_i^{\mathsf{f}}), \mathbf{s}_i\big)\,, \tag{1}$$

which is positive when the full mode reduces downstream loss, zero when it has no effect, and negative when it increases the loss. V_i^q is a gross value; the cost of computation enters through the budget below. A per-input perception loss $\mathcal{E}$ analogously induces the perception gain $G_i^{\mathcal{E}} = \mathcal{E}(\mathbf{z}_i^{\mathsf{c}}, \mathbf{s}_i) - \mathcal{E}(\mathbf{z}_i^{\mathsf{f}}, \mathbf{s}_i)$, while a planning-aware metric $\mathcal{M}_r$, oriented as a loss, induces $H_i^r = \mathcal{M}_r(\mathbf{z}_i^{\mathsf{c}}, \mathbf{s}_i) - \mathcal{M}_r(\mathbf{z}_i^{\mathsf{f}}, \mathbf{s}_i)$. Both are positive when the full mode scores better, but the allocation target is V^q, not $G^{\mathcal{E}}$ or H^r.

The oracle reference. With equal escalation costs, a selection budget k permits at most k escalations. The decision oracle selects

$$\mathcal{A}_q^*(k) = \arg\max_{\mathcal{A}:\,|\mathcal{A}|\le k} \sum_{i\in\mathcal{A}} V_i^q. \tag{2}$$

The budget is a capacity, not a quota: because V_i^q may be negative, the oracle selects only beneficial escalations and may leave capacity unused. DEEP evaluates each input counterfactually against a fixed reference scene state and defines the gain of an allocation as the sum of these per-input values; where a decision rule or loss depends on the previous action, both branches use the action of all-cheap operation (Appendix J). In sequential control, V_i^q therefore measures the counterfactual value of one escalation for the decision it informs, with the reference scene and history fixed, rather than its closed-loop value. We define $\mathcal{P}_q = \{i : V_i^q > 0\}$, and $\mathcal{N}_q^- = \{i : V_i^q < 0\}$. Then

$$\underbrace{\sum_{i\in\mathcal{P}_q} V_i^q + \sum_{i\in\mathcal{N}_q^-} V_i^q}_{\text{all-full}} < \underbrace{\sum_{i\in\mathcal{A}_q^*(k)} V_i^q}_{\text{oracle at budget } k} \quad \text{whenever } |\mathcal{P}_q| \le k \text{ and } \mathcal{N}_q^- \neq \varnothing. \tag{3}$$

Thus, whenever all beneficial escalations fit within the budget and at least one escalation is harmful, the oracle's decision gain strictly exceeds that of all-full inference. The oracle thus measures the headroom of allocation, while all-full inference remains a practical baseline when it fits the budget.

What a single score must encode. Eq. (2) also determines what an allocator must predict. Let S_i denote its score for input i, and let $\mathcal{A}_S(k)$ contain the k highest-scoring inputs. An optimal score must do more than identify promising inputs: it must order beneficial escalations by their decision value and separate them from non-beneficial ones. We obtain the following characterization.

Proposition 1 (Optimal ranking under equal costs)**.** Assume equal escalation costs and let $K_q = |\{i : V_i^q \ge 0\}|$. A score attains the optimum of Eq. (2) for every budget $k \le K_q$ if and only if: (i) all inputs with $V_i^q > 0$ are ranked ahead of those with $V_i^q \le 0$, with positive values ordered non-increasingly; and (ii) all inputs with $V_i^q = 0$ are ranked ahead of those with $V_i^q < 0$.

The proof is in Appendix A, which also shows that no single strict ranking is optimal for two downstream systems with a ranking reversal among beneficial escalations. Allocation quality therefore depends on the ordering of beneficial inputs and on the boundary between useful and harmful escalations, not on correlation with V^q alone.

Multiple fidelities and measured cost. In deployment, perception modes differ in cost and the allocator itself is not free. Let $\mathcal{F}$ denote the available fidelity modes, with cheap mode $\mathsf{c} \in \mathcal{F}$. For each mode $m \in \mathcal{F}$, define

$$V_i^{q,m} = \mathcal{L}_q(\pi_q(\mathbf{z}_i^{\mathsf{c}}), \mathbf{s}_i) - \mathcal{L}_q(\pi_q(\mathbf{z}_i^{m}), \mathbf{s}_i)\,, \tag{4}$$

and let ΔC_m be the additional compute cost of selecting mode m, with $\Delta C_{\mathsf{c}} = 0$. In our cascade, higher-fidelity modes are run from scratch, so $\Delta C_m = C_m$. If the cheap mode and the allocator run on every input at cost $C_{\mathsf{c}} + C_S$, a per-input budget $b \ge C_{\mathsf{c}} + C_S$ leaves $B = N(b - C_{\mathsf{c}} - C_S)$ for escalation. Allocation then becomes

$$\max_{\{m_i\in\mathcal{F}\}} \sum_i V_i^{q,m_i} \qquad \text{s.t.} \qquad \sum_i \Delta C_{m_i} \le B, \tag{5}$$

a multiple-choice knapsack that reduces to Eq. (2) for two modes with equal escalation cost. The budget is an average compute budget over inputs, not a per-frame deadline.

4 DEEP: Decision Evaluation for Escalated Perception

DEEP evaluates allocators, not perception models: a track fixes the perception modes, the downstream system, the data split and the cost profile, and a submission scores each input before escalation. Its design follows four principles (Figure 2). *Decision alignment*: additional computation is valued by its marginal effect on a fixed downstream loss, V_i^q in Eq. (1), rather than by perception quality. *Pre-escalation validity*: an allocator commits before the expensive mode runs and may use only the input, the cheap output, previous frames, calibration and ego proprioception; such signals are *deployable*. Signals derived from the full output or the reference state are *diagnostic*: they never enter an allocator's input at test time, and when used directly as scores they only quantify what unavailable information would contribute. On the non-test units, a supervised allocator may use decision values, full outputs and reference states as supervision. *Budget-matched reference*: allocators are compared with the decision oracle at the same budget, and uniform full-fidelity inference is reported as a baseline, not as an upper bound. *Resource accounting*: under latency and energy budgets, the allocator's own computation is charged on every input.

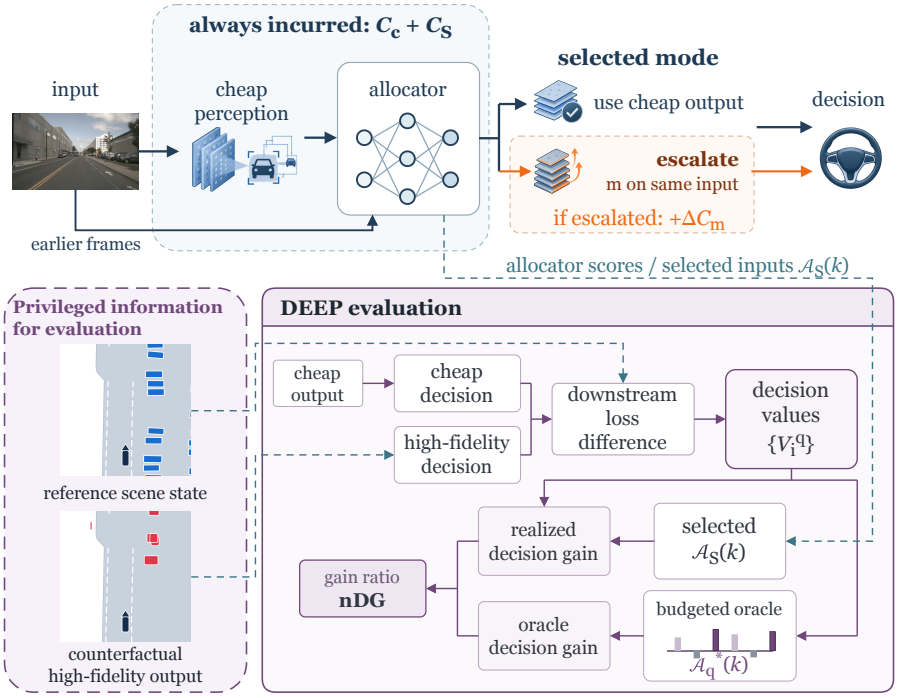


Figure 2: **The DEEP task.** Top, deployment: every input pays for the cheap mode and the allocator, which scores it from pre-escalation information only; an escalated input also pays for its higher-fidelity mode. Bottom, offline evaluation, unavailable to the allocator: the decisions obtained from the cached cheap and high-fidelity outputs are scored on the same reference scene state to give V_i^q, inputs are selected from the scores under the budget, and nDG = realized gain / oracle gain.

4.1 Tracks and downstream systems

In every escalation pair the full mode is a detector at 640 and the cheap mode is the same detector at lower resolution: YOLOv8s (Jocher et al., 2023) from 320 on nuScenes and nuPlan, and on KITTI additionally YOLOv8s from 384 and 512 and RT-DETR-l (Zhao et al., 2024) from 320 and 480. The cached outputs of both modes enter Eq. (1) counterfactually, on the same reference scene state. A cell is one combination of dataset, detector pair, geometry and downstream system. Each core cell uses either deployed monocular geometry or an oracle-geometry control in which matched detections inherit reference 3D geometry and unmatched ones keep monocular geometry; the control changes only the geometry passed to the downstream system, and allocators receive identical inputs in both.

The core track contains three downstream systems. The braking controller q_{brake}, on nuScenes and KITTI, is evaluated with a braking loss over safety shortfall and unnecessary braking. The receding-horizon controller q_{traj}, used on KITTI, selects longitudinal and lateral actions from perceived geometry and evaluates them against reference obstacles. The published learned planner q_{plan} on nuScenes is evaluated by the displacement of its argmax path from the logged future ego trajectory on the 2,655 frames that have one (Appendix J); it is included because PKL is built on it, and its loss measures deviation from the logged trajectory rather than safety. The external track runs PDM-Closed and IDM (Dauner et al., 2023) at their released configurations on nuPlan (Caesar et al., 2021), with trajectories scored against logged tracks by geometric losses. Each mode's observation is built from real detections on the nuPlan front camera: logged tracks inside the image are retained only when detected by that mode, while unmatched detections enter as false positives lifted onto the road plane. Each state is evaluated in open loop with a fresh planner instance per branch (Appendix F).

4.2 Splits, statistics and metrics

Each track uses one deterministic split, fixed before any baseline was evaluated: by scene on nuScenes, by sequence on KITTI, and by log on nuPlan, since same-log scenarios overlap in time. The test splits

contain 24 scenes, 6 sequences, and 9 logs, respectively. Allocators are reported on these held-out units; learned allocators are fit on the remaining units and evaluated once on the test split.

Setting-level quantities, namely the harm rate, the share of affected inputs ($V_i^q \neq 0$) with $V_i^q < 0$, and the harm ratio, are reported over all units because they characterize the perception–decision pair rather than a learned allocator. Confidence intervals use 1,000 bootstrap resamples of scenes, sequences, or logs, never frames, and comparisons with random are paired within each draw. Selections are evaluated in exact expectation over tie-breaks, which matters for discrete signals (Appendix G).

Normalized decision gain (nDG) is the fraction of oracle decision value realized by an allocator at the same budget, and **harm ratio** ρ_q characterizes the perception–decision setting:

$$\mathrm{nDG}_q(S,k) = \frac{\sum_{i\in\mathcal{A}_S(k)} V_i^q}{\sum_{i\in\mathcal{A}_q^*(k)} V_i^q}, \qquad \rho_q = \frac{\sum_{i:V_i^q<0} |V_i^q|}{\sum_{i:V_i^q>0} V_i^q}. \tag{6}$$

Since the oracle maximizes the numerator, $\mathrm{nDG} \in (-\infty, 1]$: $\mathrm{nDG} = 1$ matches the oracle, $\mathrm{nDG} = 0$ realizes no net decision gain, and $\mathrm{nDG} < 0$ indicates that the selected escalations cause net harm. When all beneficial inputs fit within the budget, uniform full-fidelity inference retains a fraction $1 - \rho_q$ of the oracle value; thus $\rho_q = 0$ is the monotone-benefit case in which full-fidelity inference absorbs no harmful value. Both metrics are reported only when their denominators are positive. The oracle may leave capacity unused, whereas a ranking escalates exactly its k highest-scoring inputs; under measured budgets, nDG divides by the zero-overhead oracle at the same budget, so an allocator's own cost reduces the capacity it can use. Because nDG normalizes by oracle value, it is unstable when that value sits in few inputs; Appendix E therefore also reports realized decision gain.

4.3 Budgets and baselines

The measured-budget track implements Eq. (5) as a cascade: the cheap pipeline and allocator run on every input, and an escalated input additionally runs its selected higher-fidelity mode. Per-input budgets $b = C_\mathrm{c} + \alpha C_\mathrm{f}$ are expressed in milliseconds or millijoules at levels where a zero-cost allocator could escalate a fraction $\alpha \in \{0.1, 0.2, 0.3, 0.5\}$ of inputs. Detectors run in FP16 TensorRT on the target device; measured budgets charge each perception pipeline and the allocator's preprocessing and inference, with the timing boundary of each track in Appendix J. Selection budgets measure ranking quality by ranking each test split; causal allocation at a fixed rate is reported in Appendix E.

We include random routing and ego speed as simple controls; cheap-detection uncertainty and cheap-side criticality as hand-designed signals; ridge and gradient-boosted gates over cheap-side features; a detection-list router in the style of ORIC over the 25 most confident cheap detections; and a 0.15 GFLOPs pixel router on the downscaled camera frame. The diagnostic signals comprise criticality from reference geometry, eight fixed perception-loss gains, including the false-negative gain (the reduction in missed objects) and the risk-weighted $\mathcal{E}_\mathrm{risk}$, and the PKL and TIP gains.

5 Empirical Findings

The experiments cover 3,376 nuScenes and 8,008 KITTI frames with three downstream systems on the core track and 1,440 nuPlan states with two published planners on the external track. Allocators are reported on the frozen test splits at a 20% budget unless stated otherwise. We ask, in turn, whether escalation has signed decision value (Section 5.1), whether this holds for published planners under real detector outcomes (Section 5.2), whether decision value can be predicted before escalation (Section 5.3), whether judging allocators by perception gain selects the right ones (Section 5.4), and whether an allocator's advantage survives its measured cost (Section 5.5).

5.1 Expensive perception has sign-varying downstream value

Table 1 reports decision value across our settings. With monocular geometry, escalation harms 34–54% of affected inputs in every setting (for KITTI braking, Appendix C). On nuScenes under q_brake with oracle geometry, helpful escalations reduce the all-cheap loss by 24.1% and harmful ones increase it by 10.1%, so $\rho_q = 0.42$ and uniform escalation reduces the loss by a net 14.0%. Under monocular geometry, uniform escalation reduces the trajectory controller's loss by at most 12.8%, and at 512→640 it raises the loss by 0.59% ($\rho_q = 1.11$, interval $[0.45, 3.82]$).

Table 1: Sign-varying decision value across datasets, detector families, fidelity gaps and downstream systems, over all units. Affected: inputs with non-zero decision value, of 3,376 nuScenes frames (2,655 for q_{plan}), 8,008 KITTI frames and 1,440 nuPlan states. Harmed: share of affected inputs made worse by the full mode. All-full and Oracle@20: loss reduction relative to all-cheap inference, for escalating every input (all-full, a compute-agnostic baseline) and for the oracle escalating up to 20% of inputs; † marks budget-binding cells. Rows show both geometries on nuScenes and deployed geometry on KITTI; the KITTI oracle-geometry controls are in Appendix C. nuPlan rows use real detector outcomes on nuPlan images. See also Appendices C and F.

Dataset, system, geometry	Transition / loss	Affected	Harmed	ρ_q	All-full	Oracle@20
nuScenes, q_{brake}, oracle	YOLOv8s 320→640	141	35.5%	0.42	14.01%	24.07%
nuScenes, q_{brake}, mono	YOLOv8s 320→640	273	39.6%	0.41	16.47%	27.77%
nuScenes, q_{plan}, oracle†	YOLOv8s 320→640	1,179	50.9%	0.70	1.31%	4.41%
nuScenes, q_{plan}, mono†	YOLOv8s 320→640	1,612	49.3%	0.71	1.59%	5.31%
KITTI, q_{traj}, mono	YOLOv8s 320→640	678	48.4%	0.28	12.78%	17.81%
KITTI, q_{traj}, mono	YOLOv8s 384→640	458	48.7%	0.46	5.97%	11.03%
KITTI, q_{traj}, mono	YOLOv8s 512→640	266	53.8%	1.11	−0.59%	5.16%
KITTI, q_{traj}, mono	RT-DETR-l 480→640	295	43.4%	0.58	2.94%	6.98%
nuPlan, PDM-Closed	safety loss	45	44.4%	0.33	9.81%	14.74%
nuPlan, IDM	safety loss	5	40.0%	0.06	1.42%	1.50%

For the braking controller on nuScenes under oracle geometry, the mechanism is an exchange between recovered misses and added false positives among the objects the controller considers: on harmed frames the full mode adds a false positive that the controller reacts to in 76% of cases and recovers a miss it needs in 6%, against 29% and 62% on helped frames. Across braking settings, 73–96% of the harm is unnecessary braking, not a safety shortfall. Higher resolution raises recall and, at a fixed threshold, surfaces more low-quality detections, so perception improvement does not fix the sign of decision value: in this setting, across four perception-level gains, the signs disagree on 25–33% of braking and 50–51% of planner frames where both are non-zero. The downstream system also matters: of the 79 frames to which both systems respond, 40 are helped under one and harmed under the other (Appendix H).

Sign variation also persists when each fidelity gets its own operating point, chosen on the training units from precision and recall only. With equal box counts on nuScenes braking under monocular geometry, the full mode is more precise than the cheap mode (0.76 against 0.65), yet 36.2% of affected frames are harmed; under each of three such schemes, every nuScenes and moderate-gap KITTI cell keeps harm of at least 34% and $\rho_q \geq 0.33$ (Appendix D).

5.2 The pattern holds for a published planner under real detector outcomes

PDM-Closed corroborates the core finding under real detector outcomes on nuPlan's front camera, with both published planners at their released configurations (Table 1): escalation harms 44% of the states whose safety loss changes and 38% under the scalar loss, with ρ_q 0.33 and 0.35, and an oracle at 20% capacity reduces the safety loss by 14.7% against 9.8% for uniform escalation. IDM changes its safety loss on only 5 of 1,440 states, consistent with equal recall of the two modes within 10 m, so the same detections matter to one published planner and barely to another. Decision-changing events are sparse, so this track corroborates the core track rather than replacing it (Appendix F).

Construct validity. The controls of Sections 5.1 and 5.2 test whether signed decision value is an artifact of the benchmark's construction. Range error of matched objects does not produce it: the braking controller still harms 25–48% of affected inputs when matched detections inherit reference geometry, although the trajectory controller's harm ratio then falls below 0.06 on four of five fidelity pairs. It persists under per-mode operating points. Transient detections do not produce it, since requiring detections to persist for three frames moves harmed shares by at most 6.8 points (Appendix C). Nor do loss weights, corridor width, one detector or one decision rule: varying the weights and width keeps the harm rate at 33–55% in every monocular setting (Appendix J), and it appears for both detector families, every fidelity gap we test and a published planner. These controls do not establish that the magnitudes transfer to other detectors, planners or closed-loop operation.

Table 2: Official held-out results on the core track (YOLOv8s 320→640): nDG of the deployable baselines at a 20% selection budget on the frozen test splits. Bold (significance, not rank): the paired 95% interval of realized decision gain against random, over every bootstrap draw, lies above zero; *: below zero. Planner rows use q_{plan} with average displacement; the final-displacement variants, which with these rows form the ten core cells, are in Appendix E. The detection-list router appears in its four pre-specified variants, fitted to V or $\mathbf{1}[V > 0]$.

	Controls		Heuristic		Gate		Detection-list router			
Cell	Rand.	Ego sp.	Unc.	Crit.	ridge	GBM	MLP, V	MLP, $V>0$	GBM, V	GBM, $V>0$
nuScenes oracle, q_{brake}	0.148	0.208	0.015	0.470	0.254	0.155	0.354	0.354	0.196	0.421
nuScenes mono, q_{brake}	0.132	0.148	0.088	0.407	0.293	**0.482**	**0.366**	0.255	**0.383**	0.369
nuScenes oracle, q_{plan}	-0.006	-0.021	-0.117	-0.272^*	-0.150	-0.141	-0.079	-0.074	-0.083	0.031
nuScenes mono, q_{plan}	0.092	-0.015	0.013	-0.060	0.063	0.042	0.250	0.102	-0.022^*	0.090
KITTI oracle, q_{brake}	0.139	**0.584**	-0.010^*	-0.013^*	0.238	0.226	0.255	**0.313**	0.293	**0.360**
KITTI mono, q_{brake}	0.079	**0.415**	0.035	0.074	0.156	0.158	**0.178**	**0.172**	**0.201**	**0.215**
KITTI oracle, q_{traj}	0.199	0.795	0.031^*	0.015^*	0.455	**0.538**	0.356	**0.422**	**0.516**	**0.484**
KITTI mono, q_{traj}	-0.091	-0.691	0.148	0.178	0.298	0.184	0.107	0.127	0.125	0.059

5.3 Learned allocators find pre-escalation signal

Table 2 reports the official held-out results for pre-escalation scores. Neither heuristic significantly outperforms random in any cell at the 20% selection budget; cheap-side criticality is significantly worse on three cells and uncertainty on two. Gains from learned allocators appear under oracle geometry on both KITTI controllers, where the gradient-boosted gate reaches 0.538 on the trajectory controller. Under monocular geometry, the gradient-boosted gate and both regression routers beat random on nuScenes braking, the gate at 0.482 against 0.132 for random (paired gain 0.087 $[0.014, 0.154]$ of the all-cheap loss), and all four router variants do so on KITTI braking.

The nuScenes wins also hold under the filtered nDG interval, and each keeps an nDG of at least 0.33 when any single test scene is removed; the gradient-boosted fits do not depend on the training seed, whereas the MLP router's win holds for two of six seeds. On nuPlan the wins concentrate in one test log, so the core track carries our conclusions (Appendix E).

Ego speed is a strong trivial baseline on KITTI braking, where it has the best point estimate of any deployable signal, but not on nuScenes at 20%. Trained instead on the published routing objectives of ORIC and budget-adaptive routing, the same five router architectures beat random in 12 and 15 of 200 architecture, cell and budget combinations, against 40 for decision value, although the direction depends on the architecture and partly on the target transform (Appendix E).

5.4 Ranking allocators by perception gain can select worse ones

Consider the target-free deployable signals, namely random, uncertainty, cheap-side criticality and ego speed, none fitted to either objective. Scored directly on the test splits, the perception and decision objectives share no optimum in 43 of 52 cell–budget pairs. When each objective instead selects its winner on the validation units, choosing by the false-negative gain lowers realized test decision gain by 0.074 $[0.010, 0.108]$ of the all-cheap loss at a 20% budget, a mean over eleven cells, most of it on KITTI under oracle geometry (Figure 3c); under the risk-weighted gain the reduction is not resolved. The perception-evaluation signals themselves carry information but are diagnostic rather than deployable: the risk-weighted and false-negative gains each separate from random on two of the ten core cells, and the per-input PKL and TIP gains on none of the six nuScenes cells (Appendix E). All of them require the full output or the reference state, and even computed exactly they recover only part of the oracle's value, with the best of them changing with the downstream system (Appendix B).

5.5 Measured cost reorders the allocators

A selection budget treats the allocator as free; charging its measured cost reorders the allocators (Figure 3b). The feature-based gates spend 18–19% of a full pass on feature extraction, so at a 20% latency budget the ridge gate cannot run and the batched gradient-boosted gate escalates at most 1.7% of inputs; neither beats random. The MLP detection-list routers cost 0.65 ms, about 3.5% of a full pass. They still escalate 16.5% of inputs at the 20% budget, and are the only learned allocators that

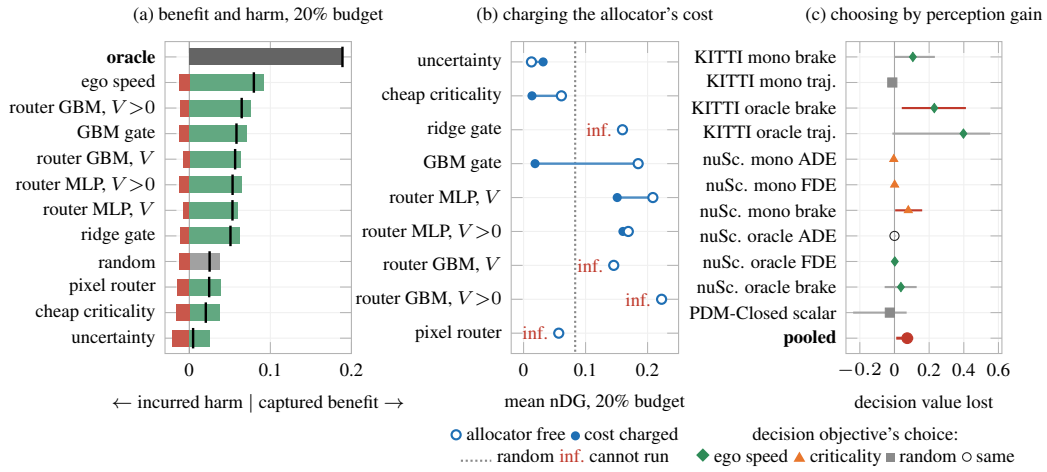


Figure 3: **Where decision value is lost**, in shares of the all-cheap loss unless stated. **(a)** Benefit captured (green), harm selected (red) and net gain (black) of each allocator at a 20% selection budget, mean over the ten core cells. **(b)** nDG with the allocator free and with its measured latency charged, 20% budget, mean over the ten core cells; inf.: the allocator's cost exceeds the budget. **(c)** Test gain of the signal chosen on validation by decision value minus that of the signal chosen by false-negative gain, among the target-free signals of Section 5.4; markers: the decision objective's choice (open: same choice); paired 95% intervals, red if excluding zero; pooled: their equal-weight mean.

beat random there. A useful summary is an allocator's *break-even overhead*, the largest per-input cost at which its ranking still realizes random's gain: at this budget its median over the core cells is 1.6–3.0 ms for the learned allocators, which the MLP routers meet in seven or eight of ten cells and the gradient-boosted routers and single-call gate exceed 6- to 14-fold. A 0.15 GFLOPs pixel router costs 26–50% of a full pass and beats random nowhere. Uniform full fidelity fits only once α reaches 29–34% on the core track and 39% on nuPlan; above that, with overhead charged and causal enforcement, it leads the best feasible allocator in eight of ten core cells at a 50% latency budget. Choosing among fidelities also matters: on KITTI at a 20% budget, the best allocation restricted to the 640 px mode reaches 0.73 of the multi-fidelity optimum on braking and 0.57 on the trajectory controller, and latency and energy rank the intermediate modes differently, with 384 px costing 0.78 of the 640 px mode in milliseconds but 0.57 in millijoules.

Allocators fall short of the oracle by missing benefit. Learned allocators beat random by capturing more benefit but select nearly as much harm (Figure 3a); ego speed has the largest net gain on average, carried by the KITTI cells. For learned allocators that fit a measured budget, ranking, not overhead or causal enforcement, accounts for most of the gap (Appendix E).

6 Conclusion

Additional perception compute does not have monotone downstream value: in every monocular-geometry setting, 34–54% of the escalations that change the downstream loss make it worse, and on nuScenes perception-level gain frequently disagrees in sign with decision value, so ranking allocators by perception gain can select worse ones. Measured against the decision oracle, learned allocators show that pre-escalation signal exists: at a 20% selection budget they capture on average about a third of the oracle's beneficial value and select nearly as much harm as random, and under measured budgets only low-overhead ones keep an advantage over random. A new allocator should improve realized decision gain from pre-escalation information only, also once its own cost is charged, and be reported per downstream system, with nDG beside realized gain and the failure modes of Figure 3 separated: missed benefit, selected harm, and computation spent on the ranking. DEEP measures single-step counterfactual value; closed-loop evaluation from camera input needs sensor simulation.

Reproducibility statement

Code, cached outputs and evaluation scripts are available at `https://github.com/anonymous221206/perception-value`. A track fixes the perception modes, the downstream system, the frozen split and the cost profile. An allocator supplies one pre-escalation score per test input, or one per budget level, computed without the full-mode output or the reference state; the released full-mode outputs, reference states and test decision values serve evaluation and must not enter an allocator's input. The intended protocol selects allocators and their hyperparameters on the non-test units and scores each allocator once on the frozen test split. The scorer returns nDG, realized decision gain and paired bootstrap intervals against random under selection budgets and, given the allocator's measured cost, under latency and energy budgets. To make this possible without rerunning perception, the release contains cached detections at every fidelity used, of YOLOv8s on the 11,384 KITTI and nuScenes frames and of RT-DETR-l on KITTI, and of both YOLOv8s modes on the 12,921 nuPlan front-camera images inside the benchmark's scenario windows; per-input decision values for every downstream system; the frozen splits, fixed before any baseline was evaluated; the downstream evaluators; the latency and energy profiles of every perception mode and allocator on the target device; and pinned versions of PKL, TIP, PDM-Closed and IDM. Every benchmark number in the paper is produced from these cached outputs by the released scripts.

AI use statement

This paper was prepared with the assistance of generative AI tools, and we disclose that use here. The authors formulated the research questions, the problem formulation, and the mathematical statements and their proofs. No data set used in this paper was generated by a generative AI tool.

Generative AI tools assisted the authors in implementing, running and debugging experiment code, including the benchmark pipelines, the evaluation and statistics scripts and the scripts that render the figures, and in specifying experiment protocols under the authors' direction. They assisted in drafting and editing the manuscript and in searching for related work; every citation was checked against its original source. The authors also used a large language model to obtain critical feedback on drafts of this manuscript and decided which of that feedback to act on.

The authors retained responsibility for the research questions, the experimental methodology and the interpretation of the reported results, and are responsible for the final text, claims and artifacts. Reported results were checked against the saved experiment outputs; the released verification scripts check reported numbers against those outputs, and agreement with them does not by itself validate the experimental protocol. Qualitative or thematic data analysis was not applicable to this work.

References

Pasquale Antonante, Sushant Veer, Karen Leung, Xinshuo Weng, Luca Carlone, and Marco Pavone. Task-aware risk estimation of perception failures for autonomous vehicles. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, 2023. doi: 10.15607/RSS.2023.XIX.100.

Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988. doi: 10.1109/5.5968.

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11618–11628, 2020. doi: 10.1109/CVPR42600.2020.01164.

Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles. In *CVPR ADP3 Workshop*, 2021. arXiv:2106.11810.

Inkook Chun, Seungjae Lee, Michael Samuel Albergo, Saining Xie, and Eric Vanden-Eijnden. Dynamic test-time compute scaling in control policy: Difficulty-aware stochastic interpolant policy. In *Advances in Neural Information Processing Systems*, volume 38, pp. 57249–57270, 2025. doi: 10.52202/085713-1714.

Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with misconceptions about learning-based vehicle motion planning. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1268–1281, 2023.

Ido Galil, Mohammed Dabbah, and Ran El-Yaniv. What can we learn from the selective prediction and uncertainty estimation performance of 523 ImageNet classifiers? In *International Conference on Learning Representations*, 2023.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 30, pp. 4878–4887, 2017.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012. doi: 10.1109/CVPR.2012.6248074.

Wei Geng, Nitinder Mohan, and Jörg Ott. Budget-adaptive routing: Skipping the weak when the strong answers anyway. In *Proceedings of the Workshop on Networks for AI Computing (NAIC '26)*, pp. 2001–2009, 2026. doi: 10.1145/3789240.3828740.

Seonyeong Heo, Shinnung Jeong, and Hanjun Kim. RTScale: Sensitivity-aware adaptive image scaling for real-time object detection. In *34th Euromicro Conference on Real-Time Systems*, volume 231 of *Leibniz International Proceedings in Informatics*, pp. 2:1–2:22, 2022. doi: 10.4230/LIPIcs.ECRTS.2022.2.

Ronald A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26, 1966. doi: 10.1109/TSSC.1966.300074.

Boris Ivanovic and Marco Pavone. Injecting planning-awareness into prediction and detection evaluation. In *2022 IEEE Intelligent Vehicles Symposium*, pp. 821–828, 2022. doi: 10.1109/IV51971.2022.9827101.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLOv8. `https://github.com/ultralytics/ultralytics`, 2023. Version 8.0.0, software.

Woosung Kang, Siwoo Chung, Jeremy Yuhyun Kim, Youngmoon Lee, Kilho Lee, Jinkyu Lee, Kang G. Shin, and Hoon Sung Chwa. DNN-SAM: Split-and-merge DNN execution for real-time object detection. In *2022 IEEE 28th Real-Time and Embedded Technology and Applications Symposium*, pp. 160–172, 2022. doi: 10.1109/RTAS54340.2022.00021.

Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In *International Conference on Machine Learning*, volume 97, pp. 3301–3310, 2019.

Donghyun Kim and Jaehyoung Park. Think as needed: Geometry-driven adaptive perception for autonomous driving. *arXiv preprint arXiv:2605.10117*, 2026.

Weixin Li and Xiaodong Yang. Transcendental idealism of planner: Evaluating perception from planning perspective for autonomous driving. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 20253–20275. PMLR, 2023.

Ye Li, Yuan Meng, Zewen Sun, Kangye Ji, Chen Tang, Jiajun Fan, Xinzhu Ma, Shu-Tao Xia, Zhi Wang, and Wenwu Zhu. SP-VLA: A joint model scheduling and token pruning approach for VLA model acceleration. In *International Conference on Learning Representations*, 2026.

Falk Lieder and Thomas L. Griffiths. Strategy selection as rational metareasoning. *Psychological Review*, 124(6):762–794, 2017. doi: 10.1037/rev0000075.

Shengzhong Liu, Xinzhe Fu, Maggie Wigness, Philip David, Shuochao Yao, Lui Sha, and Tarek Abdelzaher. Self-cueing real-time attention scheduling in criticality-aware visual machine perception. In *2022 IEEE 28th Real-Time and Embedded Technology and Applications Symposium*, pp. 173–186, 2022. doi: 10.1109/RTAS54340.2022.00022.

Vishnu Narayanan Moothedath, Umang Agarwal, Umeshraja N, James Richard Gross, Jaya Prakash Champati, and Sharayu Moharir. Inference offloading for cost-sensitive binary classification at the edge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 24449–24457, 2026. doi: 10.1609/aaai.v40i29.39627.

Hussein Mozannar and David Sontag. Consistent estimators for learning to defer to an expert. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7076–7087. PMLR, 2020.

Nuria Oliver and Eric Horvitz. Selective perception policies for guiding sensing and computation in multimodal systems: A comparative analysis. *Computer Vision and Image Understanding*, 100 (1–2):198–224, 2005. doi: 10.1016/j.cviu.2004.12.004.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *International Conference on Learning Representations*, 2025.

Xiaohuan Pei, Yuxing Chen, Yuheng Shi, Siyu Xu, Yunke Wang, and Chang Xu. Action-aware dynamic pruning for efficient vision-language-action manipulation. In *International Conference on Learning Representations*, 2026.

Jonah Philion, Amlan Kar, and Sanja Fidler. Learning to evaluate perception models using planner-centric metrics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14052–14061, 2020. doi: 10.1109/CVPR42600.2020.01407.

Jiaming Qiu, Ruiqi Wang, Brooks Hu, Roch Guérin, and Chenyang Lu. Optimizing edge offloading decisions for object detection. In *Proceedings of the IEEE/ACM Symposium on Edge Computing (SEC)*, pp. 164–177, 2024. doi: 10.1109/SEC62691.2024.00021.

Douglas A. Reece and Steve A. Shafer. Control of perceptual attention in robot driving. *Artificial Intelligence*, 78(1–2):397–430, 1995. doi: 10.1016/0004-3702(95)00029-1.

Stuart Russell and Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1–3):361–395, 1991. doi: 10.1016/0004-3702(91)90015-C.

David Weiss, Benjamin Sapp, and Ben Taskar. Dynamic structured model selection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2656–2663, 2013. doi: 10.1109/ICCV.2013.330.

Yian Zhao, Wenyu Lv, Shangliang Xu, Jinman Wei, Guanzhong Wang, Qingqing Dang, Yi Liu, and Jie Chen. DETRs beat YOLOs on real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16965–16974, 2024. doi: 10.1109/CVPR52733.2024.01605.

Hong-Yu Zhou, Bin-Bin Gao, and Jianxin Wu. Adaptive feeding: Achieving fast and accurate detections by adaptively combining object detectors. In *IEEE International Conference on Computer Vision*, pp. 3525–3533, 2017. doi: 10.1109/ICCV.2017.379.

APPENDIX

A PROOFS

A.1 PROOF OF PROPOSITION 1

Write $\mathcal{P}_q = \{i : V_i^q > 0\}$, $\mathcal{Z}_q = \{i : V_i^q = 0\}$ and $\mathcal{N}_q^- = \{i : V_i^q < 0\}$, so that $K_q = |\mathcal{P}_q| + |\mathcal{Z}_q|$. At budget k a score selects its k top-ranked inputs. With equal escalation costs every $\mathcal{A}$ with $|\mathcal{A}| \le k$ is feasible, and removing a non-positive element never decreases a sum, so the optimum of Eq. (2) is

$$\mathrm{OPT}(k) = \sum_{r=1}^{\min(k,|\mathcal{P}_q|)} V_{(r)}^q, \tag{7}$$

where $V_{(r)}^q$ is the r-th largest positive value; for $k \ge |\mathcal{P}_q|$ it equals $\sum_{i\in\mathcal{P}_q} V_i^q$.

Sufficiency. Under (i) and (ii), for $k \le |\mathcal{P}_q|$ the top k inputs are positive inputs carrying the k largest values, with ties in any order, and for $|\mathcal{P}_q| < k \le K_q$ they are $\mathcal{P}_q$ together with $k - |\mathcal{P}_q|$ zero-valued inputs. Both attain $\mathrm{OPT}(k)$ whatever the order of the zero-valued inputs.

Necessity of (ii). If some input in $\mathcal{N}_q^-$ is ranked above some input in $\mathcal{Z}_q$, the top K_q inputs cannot all be non-negative, so at budget K_q the selection contains some $j \in \mathcal{N}_q^-$. Its value is then at most $\sum_{i\in\mathcal{P}_q} V_i^q + V_j^q < \mathrm{OPT}(K_q)$.

Necessity of (i). Suppose some $i \in \mathcal{P}_q$ is ranked below an input j with $V_j^q < V_i^q$, and let k be the position of j. If $k \le K_q$, the selection at budget k contains j but not i, and exchanging j for i gives a feasible set with strictly larger value, so the score falls below $\mathrm{OPT}(k)$. If $k > K_q$, then i is not among the top K_q inputs, and the value at budget K_q is at most $\sum_{i'\in\mathcal{P}_q\setminus\{i\}} V_{i'}^q < \mathrm{OPT}(K_q)$. □

Abstention. If an allocator may also leave capacity unused, through an abstention set whose members are never selected, the same argument with (ii) replaced by abstention on every input in $\mathcal{N}_q^-$, and every input in $\mathcal{P}_q$ left non-abstained, gives optimality at every budget, including $k > K_q$.

Remark. Both conditions use equal escalation costs. With per-input costs ΔC_{m_i} and a budget B, Eq. (5) is a knapsack whose optimum depends on the pairs $(V_i^{q,m_i}, \Delta C_{m_i})$ jointly; no ordering of V^q alone is sufficient, and the abstention rule generalizes to a threshold on $V_i^{q,m_i}/\Delta C_{m_i}$ only in the fractional relaxation.

A.2 NO SHARED OPTIMUM ACROSS DOWNSTREAM SYSTEMS

Corollary 1 (No shared optimum under a ranking reversal)**.** If two inputs i, j have positive value under downstream systems q_1 and q_2, but $V_i^{q_1} > V_j^{q_1}$ while $V_i^{q_2} < V_j^{q_2}$, then no single strict ranking is optimal at every budget for both systems.

Proof. Since i and j are positive under both systems, condition (i) of Proposition 1 requires an optimal ranking under q_1 to place i above j and one under q_2 to do the reverse, which no strict ranking does.

A.3 REGRET FROM A TOP-k MISMATCH

For $k \le K_q$, take an oracle selection $\mathcal{A}_q^*(k)$ of size k whose zero-valued members are drawn from $\mathcal{A}_S(k)$ first, so that the missed-oracle set $\mathcal{U} = \mathcal{A}_q^*(k) \setminus \mathcal{A}_S(k)$ and the wrongly selected set $\mathcal{W} = \mathcal{A}_S(k) \setminus \mathcal{A}_q^*(k)$ have equal size u and never both contain a zero-valued input. If every missed oracle input exceeds every wrongly selected input by at least $\delta_V > 0$, the regret satisfies

$$\mathcal{R}_q(S,k) = \sum_{i\in\mathcal{A}_q^*(k)} V_i^q - \sum_{i\in\mathcal{A}_S(k)} V_i^q = \sum_{r=1}^{u} \big(V_{\mathcal{U}_r}^q - V_{\mathcal{W}_r}^q\big) \ge u\,\delta_V. \tag{8}$$

Top-k disagreement therefore matters most when the decision-value margin at the budget boundary is large, which is why aggregate correlation is a poor summary of allocation quality.

B Cross-Target Diagnostic Analysis

This appendix evaluates each score as one fixed function against several downstream targets on the same nuScenes frames, transition and geometry, over all units. It is an analysis of what the diagnostic signals know, not a benchmark result; the official held-out results are in Table 2.

Self-reference gap. For a planning-aware metric $\mathcal{M}_r$ computed through the learned planner, with gain H^r, define $\Gamma_{\text{self}}(r) = 1 - \text{nDG}_{q_{\text{plan}}}(H^r, k)/\text{nDG}_{q_{\text{plan}}^{\text{self}}}(H^r, k)$, which compares scoring the planner against the logged trajectory with scoring it against its own reference-conditioned output; a descriptive contrast that is unbounded above. On the 2,655 nuScenes frames with a logged future and a 20% budget, with both nDG values computed on those frames (0.893 against the self-reference for PKL), $\Gamma_{\text{self}}(\text{PKL}) = 0.60$ $[0.40, 0.84]$ and $\Gamma_{\text{self}}(\text{TIP}) = 0.56$ under oracle geometry, and 0.47 and 0.46 under monocular geometry. PKL shares model, weights and forward pass with $q_{\text{plan}}^{\text{self}}$, so a frame whose predicted heatmap moves affects both quantities by construction; measured against the logged trajectory the remaining signal is 0.36, against 0.06 for random. PKL is designed to measure consequences through its planner; Γ_{self} contrasts its allocation value against that planner's own output with its value against the logged outcome.

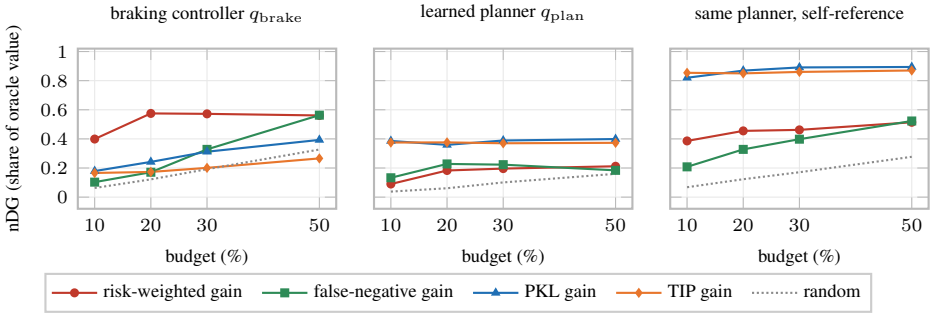


Figure 4: **Perception-level signals recover part of the decision value, and the best one depends on the downstream system.** nDG on nuScenes (YOLOv8s 320→640, oracle geometry, all units) when escalation is ranked by each signal, computed exactly after escalation. Right: the planner scored against its own reference-conditioned output.

Table 3: Cross-target diagnostic analysis: nDG at a 20% budget on nuScenes, oracle geometry, all units. Only the first three rows are deployable; the gate is fit out of fold by scene. The $q_{\text{plan}}^{\text{self}}$ column uses all 3,376 frames, the q_{plan} column the 2,655 frames with a logged future; the self-reference gap in the text recomputes $q_{\text{plan}}^{\text{self}}$ on those 2,655 frames. Bold: best diagnostic score in each column.

		Braking	Published learned planner	
Allocation score	class	q_{brake}	q_{plan}	$q_{\text{plan}}^{\text{self}}$
Random (16 seeds)	deployable	0.122	0.061	0.123
Cheap-detection uncertainty	deployable	0.005	0.134	0.096
Pre-escalation gate, GBM	deployable	0.309	0.079	—
Criticality (reference geometry)	diagnostic	0.182	0.016	0.219
False-negative gain $G^{\mathcal{E}_{\text{FN}}}$	diagnostic	0.171	0.228	0.328
Risk-weighted gain $G^{\mathcal{E}_{\text{risk}}}$	diagnostic	**0.575**	0.183	0.455
PKL gain H^{PKL}	diagnostic	0.242	0.359	**0.869**
TIP gain H^{TIP}	diagnostic	0.173	**0.376**	0.850
Target decision oracle	label	1.000	1.000	1.000

On q_{brake} and the logged-trajectory q_{plan}, no diagnostic approaches the oracle: the best are the risk-weighted gain at 0.575 $[0.32, 0.75]$ on q_{brake} and TIP at 0.376 $[0.12, 0.54]$ on q_{plan}. The best score also changes with the downstream system (Figure 4): $\mathcal{E}_{\text{risk}}$ leads on q_{brake}, against PKL's 0.242, TIP leads on the logged-trajectory target and PKL on the self-referenced one. Under deployed monocular geometry the risk-weighted gain falls to 0.227 on the braking target, since escalation also

changes the estimated range of what it detects, which a detection-set score cannot see; PKL and TIP, which read the lifted detections through a planner, reach 0.289 and 0.271 against 0.135 for random.

B.1 Longitudinal braking controller

On the full 3,376-frame set at a 20% budget, random routing reaches 0.122, uncertainty 0.005, criticality 0.182, false-negative gain 0.171, the risk-weighted gain 0.575, and PKL and TIP 0.242 $[0.01, 0.43]$ and 0.173 $[-0.14, 0.41]$. The oracle reduces the all-cheap loss by 24.07% against 14.01% for all-full inference; of the 141 frames with non-zero decision value, 91 are helped and 50 harmed.

B.2 Learned planner against the logged trajectory

On 2,655 frames, q_{plan} has non-zero decision value on 1,179, of which 600 (50.9%) are harmed by escalation. The oracle with a 20% budget reduces the all-cheap loss by 4.41% against 1.31% for all-full inference. PKL reaches $\mathrm{nDG} = 0.359$ $[0.13, 0.54]$ and TIP 0.376 $[0.12, 0.54]$, against 0.228 for the best perception-based score and 0.061 for random.

B.3 Learned planner against its own reference-conditioned output

The self-referenced control has non-zero decision value on 1,413 of 3,376 frames, of which 578 (40.9%) are harmed. The oracle with a 20% budget reduces the loss by 13.67% against 7.49% for all-full inference. PKL reaches $\mathrm{nDG} = 0.869$ $[0.82, 0.91]$ and TIP 0.850 $[0.79, 0.89]$. This condition serves the self-reference gap above and is not a downstream system of independent interest.

B.4 Lateral corridor controller

The lateral corridor controller enters only the robustness sweep of Appendix I; on nuScenes its decision value is non-zero on 23 frames, too few for scene-level intervals.

B.5 TIP score direction

PKL is higher when worse, so its gain is $H^{\mathrm{PKL}} = \mathrm{PKL}_{\mathsf{c}} - \mathrm{PKL}_{\mathsf{f}}$. The released TIP implementation computes a non-positive preference-gap quantity, with more negative values indicating worse agreement, so, following the sign convention of the released implementation, its gain is $H^{\mathrm{TIP}} = \mathrm{TIP}_{\mathsf{f}} - \mathrm{TIP}_{\mathsf{c}}$. Under this convention TIP gain correlates positively with perception gain on the wiring-validation subset, as PKL gain does, which confirms the direction.

C Sign Variation Across Configurations

Table 4: Sign-varying decision value and harm ratio across detector families, fidelity gaps and geometries. Loss reductions are relative to all-cheap inference; Oracle@20 has capacity up to 20% of inputs. Every KITTI row is budget-slack, so $\rho_q = 1 -$ all-full/Oracle@20 there. ρ_q is smallest for the largest fidelity gap under oracle geometry (row 5), where escalation is almost always beneficial.

Downstream system	Pair	Geometry	Affected	Harmed	ρ_q	All-full / Oracle@20
nuScenes, q_{brake}	Y8 320→640	oracle	141	35.5%	0.42	14.01 / 24.07%
nuScenes, q_{plan}	Y8 320→640	oracle	1,179	50.9%	0.70	1.31 / 4.41%
nuScenes, $q_{\mathrm{plan}}^{\mathrm{self}}$	Y8 320→640	oracle	1,413	40.9%	0.46	7.49 / 13.67%
KITTI, q_{traj}	Y8 320→640	mono	678	48.4%	0.28	12.78 / 17.81%
KITTI, q_{traj}	Y8 320→640	oracle	463	3.0%	0.003	68.34 / 68.58%
KITTI, q_{traj}	Y8 384→640	mono	458	48.7%	0.46	5.97 / 11.03%
KITTI, q_{traj}	Y8 512→640	mono	266	53.8%	1.11	−0.59 / 5.16%
KITTI, q_{traj}	RT 480→640	mono	295	43.4%	0.58	2.94 / 6.98%

Persistence and oracle geometry. Two controls on KITTI test the sensitivity of the sign variation to transient detections and to monocular lifting. Passing a detection to the controller only if a detection

of the same class with image IoU of at least 0.3 appears in each of the two previous frames of the same mode adds 0.2 s of latency at 10 Hz, drops 26–32% of detections, and moves the harmed share by at most 6.8 points across the twelve settings; only the braking controller at YOLOv8s 320→640 sees ρ_q fall by a third or more, from 0.17 to 0.11 under monocular and from 0.08 to 0.05 under oracle geometry. Table 5 extends oracle geometry to every fidelity pair. The braking controller's harm persists when matched detections carry reference geometry, at 25–48% of affected inputs. The trajectory controller's harm ratio collapses on four of five pairs while 3–37% of its affected inputs stay harmed, so range error on real objects drives the size of its harm more than its frequency. Oracle geometry replaces the geometry of matched detections only: 73% of KITTI and 71% of nuScenes frames contain an unmatched detection, which keeps its monocular geometry.

Table 5: Harm with deployed monocular and oracle geometry on every KITTI fidelity pair, over all units. Harmed: share of affected inputs made worse by the full mode (%); ρ_q: harm ratio.

	q_{brake}				q_{traj}			
	Harmed		ρ_q		Harmed		ρ_q	
Pair	mono	oracle	mono	oracle	mono	oracle	mono	oracle
YOLOv8s 320→640	34.2	24.6	0.168	0.080	48.4	3.0	0.282	0.003
YOLOv8s 384→640	39.8	35.3	0.473	0.295	48.7	5.1	0.459	0.002
YOLOv8s 512→640	42.9	39.7	0.615	0.467	53.8	17.0	1.114	0.051
RT-DETR-l 320→640	44.2	47.5	0.701	0.691	49.7	19.9	0.762	0.023
RT-DETR-l 480→640	42.2	47.1	0.720	0.884	43.4	36.8	0.579	0.141

D Per-Mode Operating Points

All other results use one detection threshold, 0.25, for both modes. Here each fidelity has its own operating point, selected on the training units of each track from precision and recall only, never from decision value. **S1** sets the full-mode threshold so that it emits as many boxes per frame as the cheap mode; **S2** places each mode at its F1 optimum; **S3** uses the lowest full-mode threshold whose precision is at least that of the cheap mode. Under S1 on nuScenes, the full mode goes from 4.18 to 2.59 boxes per frame and from precision 0.58 to 0.76, above the cheap mode's 0.65. Table 6 gives harm rate and ρ_q over all units. Under each scheme, every nuScenes cell and every moderate-gap KITTI cell keeps a harm rate of at least 34% and $\rho_q \geq 0.33$. Among the KITTI 320→640 cells, which Table 6 omits, the braking and oracle-geometry cells keep ρ_q below 0.18 under every scheme, while the monocular trajectory cell has ρ_q of 0.28–0.35. Across all 25 threshold pairs in $\{0.15, \ldots, 0.55\}^2$ the lowest nuScenes harm rate is 24.4%, and thresholds tuned on the training and validation units to each mode's own downstream loss give harm rates of 33–49% and ρ_q of 0.44–0.84 in the nuScenes cells and the moderate-gap KITTI braking cells.

Table 6: Harm rate and harm ratio ρ_q with a shared threshold (S0) and with per-mode operating points (S1–S3), over all units. KITTI rows use monocular geometry.

Cell	S0 (shared)	S1 (equal boxes)	S2 (F1)	S3 (precision)
nuScenes oracle, q_{brake}	35.5% / 0.42	38.5% / 0.54	34.5% / 0.39	34.1% / 0.41
nuScenes oracle, q_{plan}	50.9% / 0.70	50.2% / 0.78	50.1% / 0.71	50.2% / 0.72
nuScenes mono, q_{brake}	39.6% / 0.41	36.2% / 0.34	37.0% / 0.34	37.5% / 0.37
nuScenes mono, q_{plan}	49.3% / 0.71	49.6% / 0.83	49.3% / 0.73	49.2% / 0.71
KITTI Y8 384→640, q_{traj}	48.7% / 0.46	48.1% / 0.48	48.2% / 0.47	48.6% / 0.47
KITTI Y8 384→640, q_{brake}	39.8% / 0.47	34.9% / 0.44	37.1% / 0.45	38.8% / 0.47
KITTI Y8 512→640, q_{traj}	53.8% / 1.11	53.6% / 1.15	52.9% / 1.06	53.6% / 1.11
KITTI Y8 512→640, q_{brake}	42.9% / 0.62	41.6% / 0.62	41.2% / 0.60	42.8% / 0.62
KITTI RT 480→640, q_{traj}	43.4% / 0.58	43.4% / 0.58	46.6% / 0.52	43.4% / 0.58
KITTI RT 480→640, q_{brake}	42.2% / 0.72	41.9% / 0.74	41.9% / 0.84	42.2% / 0.72

E BENCHMARK REFERENCE RESULTS

Held-out allocation. Table 8 gives every deployable baseline at a 20% budget on all fourteen held-out cells, including the final-displacement variant of q_{plan}. Out of fold over all units the same gates look stronger — 0.510 on nuScenes braking under monocular geometry, for instance — and those numbers are released but not scored, since the gates were fit on those units. Across all four budgets, on realized decision gain with every bootstrap draw kept, the ridge and gradient-boosted gates beat random in 4 and 10 rows, the detection-list router variants in 9–11 each, mostly on KITTI, and the pixel router in none (Table 7). Among the untrained signals, ego speed beats random in ten rows, on both KITTI braking cells at every budget from 20% up, on KITTI's trajectory controller under oracle geometry at 30 and 50%, and on nuScenes braking under oracle geometry and PDM-Closed's scalar loss at 50%; cheap-side criticality and uncertainty do so once each, on nuScenes braking and on KITTI's trajectory controller under monocular geometry at 30%.

The detection-list router is a two-layer MLP with 64 units per layer, or the benchmark's gradient-boosted model, on the 25 most confident cheap detections (the 25 nearest cheap tracks on nuPlan), with V or $\mathbf{1}[V > 0]$ as target. The pixel router is a MobileNetV2 of 0.15 GFLOPs on a 128×128 frame, trained from scratch on $\mathbf{1}[V > 0]$ and served with TensorRT FP16. Both are adaptations of published detector routers rather than reimplementations, and differ from them in ways that matter here. ORIC estimates a per-image detection reward (Qiu et al., 2024); our detection-list router estimates decision value in its regression variants and $P(V > 0)$ in its classification variants, refitted per downstream system; the published objectives themselves are evaluated on these architectures in the paragraph on published routing objectives below. Budget-adaptive routing skips the weak pass when the strong model answers anyway (Geng et al., 2026); our cascade pays the cheap mode on every input, so the pixel router only decides whether the full mode is added, and its measured cost excludes the image decoding that the cheap pass already pays. Charged instead as a skipping router, where an escalated input runs the full mode in place of the cheap one, it can escalate 83.5% of KITTI inputs at the 50% latency budget against 23.9% under the cascade, and is still rarely better than random: over the 8 latency and 30 module-energy cell and budget combinations where that share is positive, it beats random once in each, on KITTI's braking controller under oracle geometry at the 50% budget, and loses in none and two, respectively. On nuPlan the detection-list router reads the cheap branch's object list rather than a detector's output, and most of its slots hold objects that escalation cannot change.

Table 7: Router variants, nDG at a 20% budget on the held-out test cells. Bold: the paired 95% interval of realized decision gain against random, over every bootstrap draw, lies above zero; asterisk: below zero. nuPlan rows use real detector outcomes; the pixel router does not apply to nuPlan.

	Detection-list router				
Cell	MLP, V	MLP, $V{>}0$	GBM, V	GBM, $V{>}0$	Pixel router
nuScenes oracle, q_{brake}	0.354	0.354	0.196	0.421	−0.048
nuScenes oracle, planner ADE	−0.079	−0.074	−0.083	0.031	−0.163
nuScenes oracle, planner FDE	0.037	−0.047	−0.093	0.065	−0.034
nuScenes mono, q_{brake}	**0.366**	0.255	**0.383**	0.369	0.215
nuScenes mono, planner ADE	0.250	0.102	−0.022*	0.090	0.082
nuScenes mono, planner FDE	0.260	0.065	−0.062	0.131	0.066
KITTI oracle, q_{brake}	0.255	**0.313**	0.293	**0.360**	0.177
KITTI oracle, q_{traj}	0.356	**0.422**	**0.516**	**0.484**	0.199
KITTI mono, q_{brake}	**0.178**	**0.172**	**0.201**	**0.215**	0.093
KITTI mono, q_{traj}	0.107	0.127	0.125	0.059	−0.021
nuPlan PDM-Closed, safety	0.143	−0.067	0.072	0.069	—
nuPlan PDM-Closed, scalar	0.141	0.278	0.065	0.072	—
nuPlan IDM, safety	too few affected test states (4)				
nuPlan IDM, scalar	0.031	0.938	0.000	0.059	—

On nuPlan, all pre-escalation inputs are rebuilt from the cheap branch with real detections, including its false positives. The test split contains 27 and 48 affected states for PDM-Closed under the safety and scalar losses, and 4 and 10 for IDM. Most of PDM-Closed's test prize lies in one log and one scenario, in which the ego moves faster (8.5 against 3.9 m/s) among many objects; the gates rank that

Table 8: Deployable baselines, nDG at a 20% budget on all held-out test cells. Bold: the paired 95% interval of realized decision gain against random, over every bootstrap draw, lies above zero; asterisk: below zero. nuPlan rows use the real-perception track; †no uncertainty signal is defined on it. Planner rows are q_{plan} with average (ADE) and final (FDE) displacement.

Cell	Random	Ego speed	Uncertainty	Criticality	Gate, ridge	Gate, GBM
nuScenes oracle, q_{brake}	0.148	0.208	0.015	0.470	0.254	0.155
nuScenes oracle, planner ADE	−0.006	−0.021	−0.117	−0.272*	−0.150	−0.141
nuScenes oracle, planner FDE	0.059	0.024	−0.016	−0.142	0.030	0.125
nuScenes mono, q_{brake}	0.132	0.148	0.088	0.407	0.293	**0.482**
nuScenes mono, planner ADE	0.092	−0.015	0.013	−0.060	0.063	0.042
nuScenes mono, planner FDE	0.080	0.007	−0.059	−0.048	−0.043*	0.079
KITTI oracle, q_{brake}	0.139	**0.584**	−0.010*	−0.013*	0.238	0.226
KITTI oracle, q_{traj}	0.199	0.795	0.031*	0.015*	0.455	**0.538**
KITTI mono, q_{brake}	0.079	**0.415**	0.035	0.074	0.156	0.158
KITTI mono, q_{traj}	−0.091	−0.691	0.148	0.178	0.298	0.184
nuPlan PDM-Closed, safety	0.128	0.205	—†	0.070	0.992	0.992
nuPlan PDM-Closed, scalar	0.125	0.201	—†	0.066	**0.988**	0.989
nuPlan IDM, safety	too few affected test states (4)					
nuPlan IDM, scalar	0.185	−0.076	—†	0.008	0.056	0.000

scenario's states first, which is what their near-perfect nDG reflects, and bootstrap draws that omit that log carry little prize.

Table 9: Diagnostic signals, nDG at a 20% budget on the held-out test cells. Each column is one fixed signal, not a per-cell selection. Bold: the paired 95% interval of realized decision gain against random, over every bootstrap draw, lies above zero; asterisk: below zero. These signals require the full output or the reference state. PKL and TIP are defined only on nuScenes. On the real-perception nuPlan track the false-negative gain counts missed tracks; neither it nor the risk-weighted gain sees the false positives that drive most of PDM-Closed's decision value.

Cell	Criticality (reference)	$G^{\mathcal{E}_{\text{FN}}}$	$G^{\mathcal{E}_{\text{risk}}}$	PKL	TIP
nuScenes oracle, q_{brake}	0.287	0.258	**0.687**	0.461	0.365
nuScenes oracle, planner ADE	−0.195	0.123	−0.085	−0.132	−0.181
nuScenes oracle, planner FDE	−0.005	0.121	−0.036	0.124	−0.035
nuScenes mono, q_{brake}	0.323	0.210	0.330	0.215	0.110
nuScenes mono, planner ADE	−0.051*	0.350	0.404	0.126	0.117
nuScenes mono, planner FDE	−0.037	0.264	0.246	0.077	0.168
KITTI oracle, q_{brake}	0.072	**0.269**	0.178	—	—
KITTI oracle, q_{traj}	0.015*	0.534	0.057	—	—
KITTI mono, q_{brake}	0.084	0.098	0.056	—	—
KITTI mono, q_{traj}	**0.404**	**0.473**	**0.400**	—	—
nuPlan PDM-Closed, safety	0.070	0.105	0.140	—	—
nuPlan PDM-Closed, scalar	0.068	0.096	0.136	—	—
nuPlan IDM, safety	too few affected test states (4)				
nuPlan IDM, scalar	0.008	0.001	0.062	—	—

Typical uncertainty per cell. Table 10 gives the median half-width of the paired interval on realized decision value at a 20% budget, as a share of the all-cheap loss. It describes the typical uncertainty of a comparison with random in each cell, not a minimum detectable effect, and whether a particular allocator beats random is decided by its own interval. Typical half-widths are below 2% in the planner cells, 4–7% in the braking cells, and 14–18% in the three cells that carry the largest reported wins, KITTI's trajectory controller under oracle geometry and the two PDM-Closed losses. The gradient-boosted gate's advantage on KITTI's trajectory controller is 17.6% of the all-cheap loss, and its own interval excludes zero; those of PDM-Closed's safety gates do not (below).

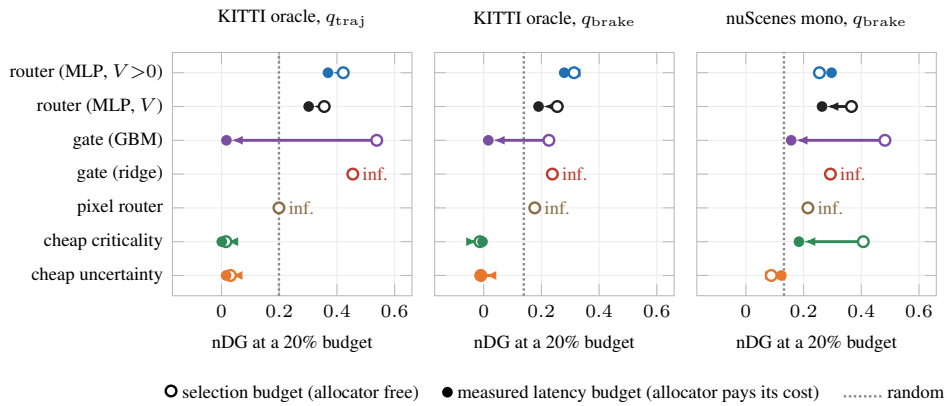


Figure 5: **Charging allocator cost reorders the allocators.** nDG on the held-out split at a 20% budget, when the allocator is free (open) and when its measured latency on the device is charged (filled; the gradient-boosted gate with batched inference). The feature-based gates lose their advantage because feature extraction consumes the budget (inf.: the allocator cannot run within it at all), while both MLP variants of the detection-list router, at about 3.5% of a full pass, keep most of it; its gradient-boosted variants cannot run within any measured budget. Figure 6 covers every budget level.

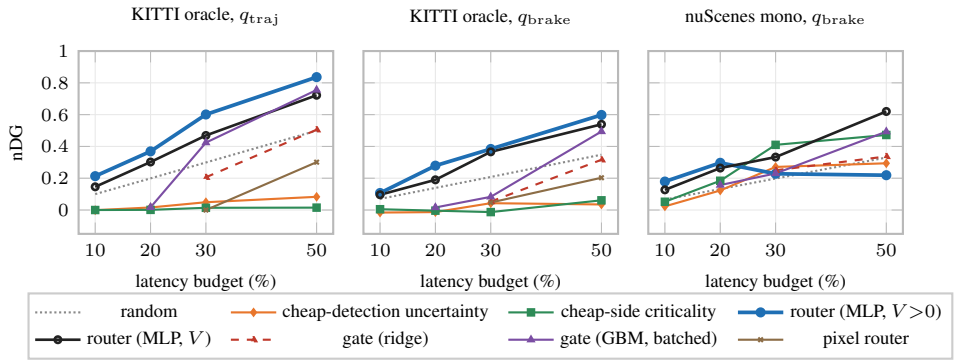


Figure 6: **nDG across measured latency budgets.** Held-out split, the three core cells of Figure 5, every budget level. Missing points are infeasible: the ridge gate cannot run within the 10 and 20% budgets, the batched gradient-boosted gate within 10%, and the pixel router within 10 and 20% on KITTI and any budget on nuScenes. Both MLP variants of the detection-list router, at about 3.5% of a full pass, keep their ranking skill.

Table 10: Typical uncertainty of paired comparisons with random at a 20% budget: the median half-width, over all scored signals, of the paired 95% interval on realized decision value, as a share of the all-cheap loss, with every bootstrap draw kept.

Cell	Half-width	Cell	Half-width
nuScenes oracle, planner ADE	0.7%	KITTI mono, q_{brake}	3.7%
nuPlan IDM, scalar	1.0%	nuScenes mono, q_{brake}	5.8%
nuScenes oracle, planner FDE	1.1%	nuScenes oracle, q_{brake}	6.0%
nuScenes mono, planner ADE	1.1%	KITTI oracle, q_{brake}	6.5%
nuPlan IDM, safety	1.1%	nuPlan PDM-Closed, scalar	14.0%
nuScenes mono, planner FDE	1.7%	nuPlan PDM-Closed, safety	16.0%
KITTI mono, q_{traj}	2.3%	KITTI oracle, q_{traj}	17.6%

Streaming allocation. The budget in Table 2 is filled by ranking a whole test split, which no deployed allocator can do. Over the twelve pooled cells and six learned signals at a 20% rate, with every bootstrap draw kept, a score threshold calibrated on the validation units and frozen before the test stream differs from that ranking in nDG by 0.00 $[-0.05, +0.09]$, while 49 of the 72 rows miss the rate by more than five points, and KITTI escalates a median of 29–33% of inputs at a 20% target.

Enforcing the rate causally, with at most $\lfloor 1 + \alpha t \rfloor$ escalations after t inputs of a unit, lowers nDG by 0.09 $[-0.17, +0.00]$, because escalations arrive in bursts that the cap truncates; the interval reaches zero, so the size of this cost is not resolved. Rate controllers tuned on validation units do not recover it under the pre-specified selection rule: an adaptive threshold that tracks the rate online cuts the rows missing it from 54 under the cap to 14 and still lowers nDG by 0.07 $[-0.15, +0.02]$, and the best token bucket has unbounded capacity and so reduces to the cap. On random scores the cap costs about 0.01 and the adaptive threshold nothing, so what loss there is belongs to the learned rankings under a causal budget. What the selection track reports is therefore the ranking quality of a signal; turning a ranking into a causal allocation at a fixed rate remains an open part of the problem.

Ranking allocators by perception gain. DEEP scores a signal by the decision value it realizes; a perception-first evaluation would score the same signals by the perception gain they realize instead. Holding the protocol and the tie expectation fixed and changing only that objective, we compare the sets of optimal signals, with ties within 10^{-9}, on the 52 cell–budget pairs whose decision value is defined on test. On the target-free deployable pool — random, uncertainty, cheap-side criticality and ego speed, none of them fitted to either objective — the two objectives share no optimum in 43 of the 52 pairs when the gain is the reduction in missed objects and in 44 when it is the risk-weighted gain, with a median Kendall τ of -0.33 between the two rankings. Adding the learned deployable signals gives 50 and 50; the pool with the perception diagnostics gives 45 and 42, where the perception objective's winner is partly definitional, since the signal equal to the gain being scored attains its optimum. These counts are descriptive.

To estimate what choosing by perception gain costs, each objective selects its winner from the target-free pool on the validation units, and the two winners are compared once on test by realized decision gain, with the signal pools, test statistic and decision rule specified in advance. Over the ten core cells and PDM-Closed's scalar loss (the other nuPlan cells have at most seven affected validation states), selecting by the missed-object gain lowers realized test gain by 0.074 $[0.010, 0.108]$ of the all-cheap loss at a 20% budget, and by 0.039–0.091 at the other budgets, each interval excluding zero. Under the risk-weighted gain the reduction at 20%, 0.064 $[-0.004, 0.099]$, is not resolved. Most of the effect lies in KITTI under oracle geometry, where the decision objective selects ego speed and the perception objectives select uncertainty or cheap-side criticality; in four nuScenes cell–objective pairs both select the same signal.

Training on perception gain instead. The objective can also be swapped at training time. We refit every learned allocator on a perception-gain label, with the same inputs, hyperparameters, seeds and training units, and score it against decision value; the comparison and its interpretation were specified in advance, and the same fitting procedure is used for the decision-value baselines.

The mean paired difference at a 20% budget over the twelve pooled cells is $+0.16$ $[-0.04, +0.27]$ for the ridge gate and $+0.13$ $[-0.03, +0.23]$ for the gradient-boosted gate, and every interval includes zero, for all six architectures and all three perception-gain labels. Five of the six lean towards the decision-value target and the gradient-boosted regression router leans the other way. The effect is cell-specific rather than uniform: the decision-value target wins significantly in 10 of the 72 pooled cell and architecture pairs and the perception-gain target in 8: four are the gradient-boosted routers on PDM-Closed, where the two labels can be compared on only 11 and 28 training states, which is consistent with sparse decision-value supervision, and four are on the nuScenes planner. On the ten core cells the target moves the gates by $+0.052$ and $+0.049$. We therefore do not claim that the objective rather than the architecture produces the allocation advantage: Table 2 reports what these architectures achieve when they are trained on decision value. This is the controlled counterpart of routers that predict perception quality, such as RTScale (Heo et al., 2022), and, with the risk-weighted label, of a task-aware risk target (Antonante et al., 2023): inputs, architecture and split are fixed and only the reward changes.

Published routing objectives. The perception-gain labels above are perception losses, whereas the published detector routers optimize detection quality. We therefore also refit the four detection-list router variants and the pixel router on the two published objectives, taken from the papers and the authors' released implementations: ORIC's offloading reward, the gain in contextual mAP@0.5 (101-point interpolation, 1,000 sampled context frames) from replacing the weak detector's output on a frame with the strong detector's (Qiu et al., 2024), and the objective of budget-adaptive routing, the change in dataset-wide AP@0.5 (one precision–recall curve, all-point interpolation) when one frame's weak output is swapped for the strong one while every other frame keeps its weak output (Geng et al., 2026). Each enters in its published CDF-normalized form, with the normalization fitted on the fit units only, for the regression variants and in its binary form for the classification variants. Architecture, features, hyperparameters, seeds, training units and scorer are unchanged and only the label differs, so this evaluates the published objectives on our architectures rather than the published methods; AP is computed on the cached detections, which keep confidences down to 0.10, ORIC's reward at that floor and the budget-adaptive objective at its published threshold of 0.30; and the one component of a published method that is reimplemented, the budget-adaptive arbiter of Geng et al. (2026), picks the gradient-boosted estimator at every quota and so coincides with it. The nuPlan cells are excluded, since their observations carry no scored detections from which AP can be computed.

The choice of objective matters. Across the ten core cells and four quotas, a router trained on a published objective beats random where its decision-value counterpart does not in 24 rows, and the reverse holds in 84. The direction depends on the architecture: on the published objectives the gradient-boosted regression router is significantly ahead of its decision-value version in 10 and 11 of 40 rows and the pixel router in 2 and 2, while the MLP regression router falls behind in 4 rows under ORIC's reward. Decision-value training yields more wins against random overall, 40 against 12 and 15 across the five architectures, but no objective serves every architecture best.

Target transform. The published objectives enter the regression routers in CDF-normalized form, whereas V enters raw, so a difference between them may come from the objective or from the transform. To separate the two, we also refit both regression routers on the rank of V and on a signed empirical CDF of V ($F^{+}(V)$ for $V > 0$, zero at zero, $F^{-}(V) - 1$ for $V < 0$, fitted on the fit units), with everything else unchanged (Figure 7). This comparison was decided after the published-objective results were seen, and its intervals are not corrected for multiplicity.

For the MLP router, the deficit under ORIC's reward is consistent with a target-transform effect. Over the ten core cells at a 20% budget, the reward trails raw V by 0.139 $[0.021, 0.244]$ in nDG and rank(V) trails it by 0.170 $[0.048, 0.244]$, while the reward and rank(V) differ by only $+0.032$ $[-0.058, 0.095]$. For the gradient-boosted router, no pooled comparison separates the AP-swap objective from a transform of V: it leads rank(V) by 0.120 $[-0.005, 0.219]$ and the signed CDF of V by 0.053 $[-0.058, 0.128]$, its own lead over raw V, 0.106 $[-0.012, 0.214]$, is not resolved, and the signed CDF recovers about half of that lead, as well as part of the published objectives' gain on the planner cells.

	KITTI mono q_{brake}	KITTI mono q_{traj}	KITTI oracle q_{brake}	KITTI oracle q_{traj}	nuSc. mono q_{brake}	nuSc. mono ADE	nuSc. mono FDE	nuSc. oracle q_{brake}	nuSc. oracle ADE	nuSc. oracle FDE
MLP, rank(V)	−0.03	**−0.75***	−0.01	−0.12	**−0.39***	−0.17	−0.13	−0.21	−0.05	+0.16
MLP, signed CDF(V)	0.00	−0.17	−0.09	−0.16	−0.08	0.00	−0.02	−0.06	−0.04	−0.02
MLP, ORIC reward	−0.13	−0.26	−0.14	−0.10	−0.32	**−0.17***	**−0.30***	−0.19	+0.14	+0.09
MLP, AP swap	−0.07	+0.11	−0.13	−0.09	−0.13	−0.05	−0.16	−0.07	+0.12	+0.16
GBM, rank(V)	+0.01	−0.68	−0.02	**−0.08***	−0.12	+0.06	**+0.17***	+0.13	+0.17	+0.22
GBM, signed CDF(V)	−0.01	−0.08	−0.01	−0.03	+0.04	**+0.12***	**+0.22***	−0.02	+0.09	+0.20
GBM, ORIC reward	−0.17	0.00	**−0.22***	−0.26	−0.27	+0.28	**+0.25***	−0.11	**+0.16***	+0.26
GBM, AP swap	−0.06	+0.13	−0.16	−0.16	−0.12	**+0.29***	**+0.19***	+0.25	**+0.28***	**+0.42***

Figure 7: **Target transform against training objective.** Difference in nDG from the same regression router trained on raw decision value, on the ten core cells at a 20% budget; bold with an asterisk: paired 95% interval excludes zero. Rank and signed CDF keep the decision-value objective and change only its transform; ORIC's reward and the AP-swap objective change the objective in its published CDF form. Post hoc, uncorrected for multiplicity.

Training seeds. The bootstrap holds each fitted allocator fixed, so it contains no training-seed variation. Refitting with five further seeds leaves the gradient-boosted gate and routers bit-identical, since their fits use no randomness at these sample sizes, so each of their wins holds for every seed. The MLP routers vary: over six seeds their nDG at a 20% budget spans up to 0.22 within a cell for the regression router, 0.15–0.37 on nuScenes braking under monocular geometry, and up to 0.28 for the classification router. Of the five MLP wins on the core track in Table 2, counted by the filtered nDG interval for each seed, the classification router's on KITTI's trajectory controller under oracle geometry holds for all six seeds and each of the others for two or three.

Transfer between consumers. Decision value differs between downstream systems, but an allocator fitted for one of them need not. Scoring each cached allocator against another consumer's decision value on that consumer's test split, the median transfer regret, the nDG of the transferred allocator minus that of the allocator fitted for the evaluated consumer, is -0.009 over the 528 off-diagonal entries, and with every draw kept 14 of 78 diagonal entries beat random at a 20% budget against 9 of 132 off-diagonal ones. These are descriptive counts over dependent entries, not a paired test of fitting against transferring. Cached router scores exist only for the frames of the consumer they were fitted for, so 64 planner-to-braking entries cover 78.7% of the braking frames; on the 464 off-diagonal entries with full coverage the median regret is -0.004 and 9 of 116 beat random at 20%, so the counts do not come from missing scores.

On nuPlan the direction of transfer does: an allocator fitted for IDM realizes none of PDM-Closed's value, while the gradient-boosted gate fitted for PDM-Closed's safety loss reaches 0.94 on IDM, where the same gate fitted for IDM reaches 0.00. Consumer dependence is a property of the decision values, and the allocators recover part of it.

Unnormalized gain. nDG divides by the oracle prize, which is smallest in exactly the cells whose value sits in one unit. The benchmark therefore judges significance on realized decision gain with every bootstrap draw kept. Of the 18 deployable rows other than ego speed whose filtered nDG interval lies above random at a 20% budget, 14 also beat it on realized gain: the KITTI and nuScenes rows, and PDM-Closed's scalar loss under the ridge gate, whose advantage over random is 30.5% of the all-cheap loss. The four that do not are the two PDM-Closed safety gates, the scalar-loss gradient-boosted gate, and IDM's detection-list router; for the three gates the lower bound is exactly zero, because draws without the dominant log carry no difference to random. The prize filter is one-sided in this sense: across the four budgets it adds 4 to 11 wins and removes none. Removing that log leaves the gates' ratio at 0.99 and their realized gain at 12.0 of 162.9.

Uniform full fidelity. A cascade pays C_{c} on every input, so a per-input budget also admits running the full mode alone, without the cheap pass, once $b \geq C_{\mathsf{f}}$, that is, once the escalation allowance $b - C_{\mathsf{c}}$ reaches $C_{\mathsf{f}} - C_{\mathsf{c}}$: from $\alpha = 28.6\%$ on KITTI, 34.3% on nuScenes and 39.0% on nuPlan under the measured latencies. At the 10 and 20% budgets of Figure 6 it is infeasible. Where it is feasible, what it realizes is the gross benefit less the harm it absorbs: on KITTI's trajectory controller under oracle geometry, where harm is rare, it reaches $\mathrm{nDG} = 0.998$ against 0.601 and 0.836 for the detection-list router at the 30 and 50% budgets.

Charging each allocator its measured overhead, enforcing the budget causally and comparing loss reductions at the 50% latency budget, uniform full fidelity is ahead of the best feasible allocator in eight of the ten core cells; the two exceptions, the learned planner under oracle geometry and KITTI's trajectory controller under monocular geometry, are the cells in which uniform escalation raises the loss on the test split. It is therefore the baseline to beat once it fits; when every beneficial input fits within the budget, the harm ratio is what an allocator can still recover from it.

Quota and capacity. A ranking submission escalates exactly its $\mathrm{round}(\alpha N)$ highest-scoring inputs, while the oracle of Eq. (2) may leave capacity unused. On every test cell at the reported budgets at least that many inputs have non-negative value, so the two oracles coincide there; the ranking track has no abstention, and allowing it would define a separate track.

Bootstrap draws. A draw is dropped when its oracle prize falls below a quarter of the full-sample prize, since nDG is not comparable across draws with near-zero denominators. At a 20% budget this removes at most 8 of 1,000 draws on nine of the ten core cells and 89 on KITTI's trajectory

controller under oracle geometry, but 330 and 354 on the two PDM-Closed cells, whose prize rests on few states; their intervals are conditional on that filter. The bootstrap resamples test units with each fitted allocator held fixed, so it does not include the variation from refitting with other training seeds. Counts of cells in which a signal beats random are descriptive: the cells share frames, fidelities and allocators and are not independent replications.

Measured budgets. Figure 6 gives nDG for three core cells at every budget level. The allocator costs in Table 11 come from one profiling run of every allocator on the device; energy is charged under one module convention for detectors and allocators alike, the GPU, CPU and SoC rails over idle, and every energy claim in this paragraph also holds when only the GPU rail or all rails are charged. On nuPlan, with its own detector costs, the batched gradient-boosted gate also beats random at the 20% latency budget on PDM-Closed's scalar loss, escalating 5% of states, and the classification-target detection-list router beats random on IDM's scalar loss at every latency and energy budget.

On the KITTI fidelity ladder at a 20% budget, the best 640-only allocation reaches 0.73 of the multi-fidelity optimum on braking and 0.57 on the trajectory controller, so choosing among fidelities is worth about a third more than choosing whether to escalate on braking and three quarters more on the trajectory controller. Latency and energy disagree about the intermediate modes — 384 px costs 0.78 of 640 px in milliseconds but 0.57 in millijoules — so at a 10% budget the energy-optimal braking plan sends more inputs to 384 px than the latency-optimal one and exceeds the latency budget by 21%. The deployable multi-fidelity gate pays three model evaluations per input, cannot run within budgets below 30%, and reaches $\mathrm{nDG} = 0.068$ on braking and 0.172 on the trajectory controller at 50%.

Benefit captured and harm incurred. The gap between an allocator and the oracle at the same budget splits exactly into the positive value the allocator misses and the harm it selects, less the positive value the budget forces even the oracle to leave out. At a 20% budget the last term is negligible on the core cells, so the gap is mostly missed benefit. Averaged over the ten core cells, as shares of the all-cheap loss, the oracle realizes 0.189; ego speed captures a benefit of 0.092 at a harm of 0.012, the gates and detection-list routers 0.060–0.075 at 0.007–0.012, the pixel router 0.039 at 0.015, and random 0.038 at 0.013. Per cell, the ten non-random deployable signals incur significantly less harm than random in 11 of 100 signal–cell pairs and more in 12, and capture more benefit in 11 and less in 9.

Overhead a ranking can afford. Scoring each cached ranking at every capacity with the scorer's own budget rule gives the overheads at which it still realizes at least random's gain. At a 20% latency budget on the core cells, the median largest such overhead is 1.6–3.0 ms for the learned allocators. The MLP routers' measured 0.65 ms lies inside that set in eight and seven of the ten cells, the batched gradient-boosted gate's 3.5 ms in three, and the gradient-boosted routers' 15.9 ms, the single-call gate's 21.4 ms and the ridge gate's 3.9 ms in none. Uncertainty and cheap-side criticality cost little but beat random in few cells even at no cost. Because V is signed, the set of affordable overheads need not be an interval; at the measured overheads every result equals the budget tables.

Where the gap to the oracle goes. For the learned allocators, with train-only fits and validation-calibrated thresholds, we split the gap between the zero-overhead oracle and a causal allocator at the same latency budget into three terms: the oracle value lost to the allocator's overhead, the gap between that oracle and top-k by the allocator's scores at the remaining capacity, and the difference between top-k and the causal cap policy above. With one inference call per input, the gradient-boosted models cannot run at either the 20 or the 50% budget. For the MLP routers at 20%, over the ten core and two PDM-Closed cells, the gap averages 0.18–0.20 of the all-cheap loss, of which 0.175–0.177 is ranking, 0.007–0.021 the causal cap, and 0.0001 on average (at most 0.0012 in any cell) overhead; at 50% the ranking term is 0.094–0.125 of a 0.138–0.156 gap across the MLP routers and the ridge gate. The causal term is negative in 16 of the 62 feasible rows, where the cap skips harmful inputs that top-k keeps, so the terms are an accounting of this protocol rather than losses or separate causes.

Table 11: Share of inputs each deployable allocator can escalate under a measured per-input latency budget after paying for itself, on nuScenes / KITTI (median over cells). inf.: the allocator's own overhead exceeds the budget, $C_S > b - C_c$, so it cannot run at all. Diagnostic signals require the full output or the reference state and are not deployable under measured budgets.

Allocator	Overhead per input	Escalated at 20% budget	at 50% budget
Random	0 ms	20.0 / 20.0%	50.0 / 50.0%
Cheap-detection uncertainty	0.21–0.29 ms	18.5 / 18.8%	48.5 / 48.8%
Cheap-side criticality	1.30–1.47 ms	12.4 / 12.9%	42.4 / 42.9%
Detection-list router, MLP	0.65 ms	16.6 / 16.5%	46.6 / 46.5%
Gate, ridge	3.93 ms	inf. / inf.	29.7 / 28.7%
Gate, GBM, batched inference	3.54 ms	1.7 / 0.8%	31.7 / 30.8%
Gate, GBM, one call per input	21.4 ms	inf. / inf.	inf. / inf.
Pixel router	9.8 / 4.8 ms	inf. / inf.	inf. / 23.9%

F External Planner Track

Setup. PDM-Closed and IDM were checked against their released configurations field by field, and an unmodified run of the nuPlan closed-loop simulator succeeded for both on the same scenarios; this run is a configuration check, and the evaluation below is statewise and open-loop. For each of 1,440 logged states (60 scenarios $\times$ 24 states), a fresh planner instance is built for each of three branches — reference tracks, tracks the cheap detector would report, and tracks the full detector would report — since PDM-Closed carries internal state. Each proposed trajectory is scored against the logged tracks over 0.5–4.0 s by five losses: collision, clearance shortfall c below 1 m (clipped at 2 m), mean deviation from the logged ego trajectory, the safety loss $10 \cdot \mathbf{1}[\text{collision}] + 1.5\, c^2$, and the scalar loss, which adds 0.1 times the mean deviation in meters to the safety loss.

Real detector outcomes. We fetch the 12,921 front-camera images inside the benchmark's scenario windows and run the YOLOv8s engines at 320 and 640 px; the image nearest each state is at most 41 ms from its lidar sweep. Logged tracks are projected into the image with the camera calibration and matched to each mode's detections at threshold 0.25 by Hungarian assignment over class-compatible pairs with IoU of at least 0.3. A vehicle, pedestrian or bicycle track inside the image is kept only if that mode detects it; tracks outside the image and static classes pass through unchanged; and unmatched detections enter as zero-velocity false positives, lifted onto the road plane with a class-median size. Per state, about 18.5 tracks fall inside the image; the cheap mode keeps 5.4 and adds 0.83 false positives, and the full mode keeps 7.7 and adds 1.68. Dropping the unmatched detections, so that each mode only removes logged tracks, leaves PDM-Closed's safety loss changing on 6 states instead of 45. Recall is 0.29 at 320 px and 0.42 at 640 px, and 0.65 at both within 10 m. A branch in which every in-image track is kept and no false positive is added reproduces the reference observation on every state, and the reference branch reproduces the stored reference values exactly.

Sensitivities. For PDM-Closed under the safety loss, removing false positives leaves 6 affected states (harm 33%); a full-mode threshold count-matched on the training logs (0.48) gives 35 affected states (harm 26%, $\rho_q = 0.25$); and IoU 0.5 matching gives 51 (harm 47%, $\rho_q = 0.37$). A synthetic alternative to this construction, a detection profile fitted on 46,469 KITTI object outcomes, predicts a recall gap of 0.34 between the two modes against the 0.12 measured here, and correspondingly denser decision value for both planners (Table 12).

The cells are not independent: the safety aggregate contains collision and clearance, and the scalar aggregate contains safety. We read them as two planners under a family of related losses rather than ten replications. Under the detection profile, where both planners respond on enough states, their rankings on commonly affected states have Spearman correlation 0.62–0.81 and Goodman–Kruskal γ of $+0.49$ to $+0.69$; under real detector outcomes IDM responds on too few states for this comparison.

G Tie-Breaking and Lift Conventions

Ties in top-k selection. Selecting the top k of a discrete-valued signal leaves large tie groups, and how they are broken can dominate the result. At a 20% budget over 3,376 nuScenes frames,

Table 12: All five losses for both published planners under real detector outcomes, over all 1,440 states, and the two aggregate losses under the KITTI detection profile. The losses are geometric and are not nuPlan's metric suite. The IDM collision row is a single affected state.

Planner	Loss	Affected	Harmed	ρ_q	All-full / Oracle@20
Real detector outcomes					
PDM-Closed	collision	24	25.0%	0.33	11.54 / 17.31%
PDM-Closed	clearance shortfall	45	44.4%	0.34	5.22 / 7.89%
PDM-Closed	logged-trajectory deviation	104	37.5%	0.55	1.90 / 4.25%
PDM-Closed	safety aggregate	45	44.4%	0.33	9.81 / 14.74%
PDM-Closed	scalar aggregate	104	37.5%	0.35	8.26 / 12.68%
IDM	collision	1	0.0%	0.00	1.61 / 1.61%
IDM	clearance shortfall	5	40.0%	0.31	0.79 / 1.13%
IDM	logged-trajectory deviation	30	36.7%	0.76	0.14 / 0.56%
IDM	safety aggregate	5	40.0%	0.06	1.42 / 1.50%
IDM	scalar aggregate	30	33.3%	0.14	1.09 / 1.26%
KITTI detection profile					
PDM-Closed	safety aggregate	70	37.1%	0.30	14.71 / 21.12%
PDM-Closed	scalar aggregate	175	30.3%	0.30	12.58 / 17.90%
IDM	safety aggregate	47	27.7%	0.22	16.82 / 21.58%
IDM	scalar aggregate	105	33.3%	0.23	14.00 / 18.08%

only 326 frames lie strictly above the cut value of the false-negative gain while 366 share it, so 52% of the selected set is decided by the tie-break rather than by the signal. The effect is larger in the score-free comparison, where the braking controller has zero decision value on most frames: under monocular geometry on the 2,655 frames with a logged future, 74% of its top-20% set and 83% of its top-30% set are tied at zero. Breaking ties by row order makes two independent systems select the same early frames and manufactures agreement between them. The cross-target analyses of Appendix B therefore break ties with a random key, using a different seed per system, averaged over eight draws, and the benchmark evaluates each selection in exact expectation over random tie-breaks, which agrees with 20,000 sampled orders to within Monte Carlo error. Continuous signals — the risk-weighted gain, uncertainty, PKL and TIP — have tie groups of size one and are unaffected. Agreement between independent implementations does not test a tie-breaking convention they share: two implementations of the metric agree to three decimal places whenever both use a stable sort, so the convention is specified here rather than inferred from that agreement.

A tie-invariant version of the transfer test. Random tie-breaking removes the row-order artifact but does not show that *no* optimal allocation for one target serves the other. Let $\Omega_{q_1}(k)$ be the set of all allocations of size at most k that attain the optimum of Eq. (2) for q_1, and define the most favorable member for q_2,

$$\mathcal{A}^*_{q_1 \to q_2}(k) = \arg \max_{\mathcal{A} \in \Omega_{q_1}(k)} \sum_{i \in \mathcal{A}} V_i^{q_2}, \tag{9}$$

which gives the transfer every advantage. The two directions behave differently. The planner's optimum is nearly determined, and even its most favorable completion reaches $\mathrm{nDG}_{q_{\mathrm{brake}}} = 0.067$ at a 20% budget under oracle geometry and 0.227 under monocular. The braking controller's optimum is heavily under-determined — 111 responsive frames among the 2,655 with a logged future against a 531-frame budget under oracle geometry — and its most favorable completion reaches $\mathrm{nDG}_{q_{\mathrm{plan}}} = 0.930$ and 0.896. We report both, and rest the cross-system comparison of Section 5.1 on per-frame sign disagreement, on the pairwise statistic below and on the binding direction, rather than on a symmetric statement.

Robustness of the pairwise result. The share of strictly ranked frame pairs that the two systems order differently is stable across geometries: 0.510, with Goodman–Kruskal $\gamma = -0.021$, under oracle geometry and 0.502, with $\gamma = -0.003$, under monocular geometry.

Box centers in the monocular lift. The lift reports the distance to the box center, as the nuScenes submission format and the published planning-aware metrics expect. The convention matters for

those metrics, which consume the 3D submission: scoring instead from the near face of each box would place every object about half an object length closer, 2.3 m for a car and 5.6 m for a bus.

H HARM MECHANISM

Table 13 gives the per-frame detection statistics behind the mechanism of Section 5.1. Escalation recovers missed objects and adds false positives at once. Under q_{brake}, harmed frames gain $+1.980$ false positives against $+1.176$ on helped frames while recovering only 0.600 misses against 1.396; on 56.0% of harmed frames escalation recovers no miss at all, introducing only spurious detections for a collision-averse controller to react to. The same ordering holds for the learned planner. Detection-level summaries such as mAP aggregate these effects over the dataset, and per-input planning-aware scores such as PKL and TIP value them through their own planner objectives; neither measures whether the exchange improves the target downstream decision on that frame.

Table 13: Mean change in detection errors from cheap to full mode (Δ = full − cheap), by the sign of decision value. Harmed frames add more false positives and recover fewer misses.

	nuScenes, q_{brake}			nuScenes, $q_{\text{plan}}^{\text{self}}$		
	helped	harmed	unaffected	helped	harmed	unaffected
Δ false positives	$+1.176$	$+1.980$	$+0.849$	$+1.093$	$+1.251$	$+0.671$
Δ false negatives	-1.396	-0.600	-0.790	-1.321	-1.014	-0.522
Δ detections	$+2.505$	$+2.480$	$+1.581$	$+2.315$	$+2.190$	$+1.154$
frames with an added FP	61.5%	78.0%	—	60.0%	64.2%	—
frames recovering no miss	25.3%	56.0%	—	31.0%	44.6%	—

Figure 8 shows where these changes fall relative to the controller's corridor, and Figures 10 and 11 show examples of both mechanisms in camera images.

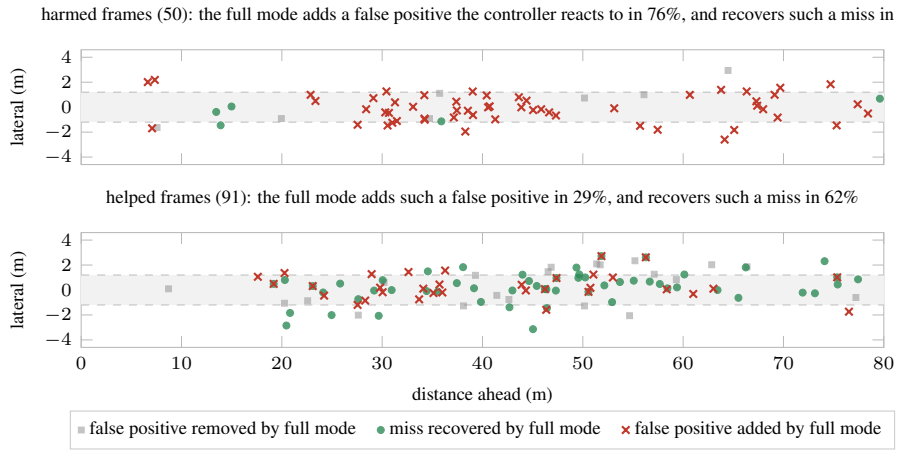


Figure 8: **Where escalation changes what the braking controller sees.** Bird's-eye view of every object the controller considers — inside its corridor (shaded) and within 80 m — that differs between the cheap and full modes, on the nuScenes frames with non-zero decision value under oracle geometry whose corridor objects differ between the modes. Harmed frames are dominated by false positives that the full mode adds; helped frames by misses that it recovers.

Table 14 tests directly whether a perception-level reward, of the kind used by selective offloading, determines the sign of decision value. It does not. Spearman correlations between perception gain and decision value on affected frames are at most 0.11, among affected frames with positive perception gain, 16–24% are harmed under braking and about half under the planner, and for the planner the sign disagreement is close to 50% for all four perception gains. On the 79 frames with non-zero decision value under both systems, 40 have opposite signs, so the same perception transition helps

one downstream system and harms the other. Figure 9 shows the per-frame joint distribution for one perception gain.

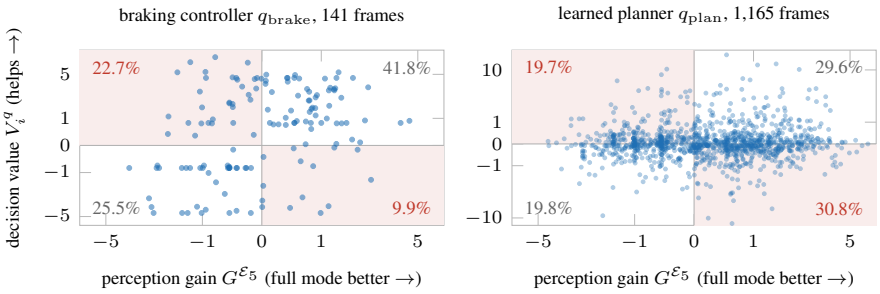


Figure 9: **Perception gain does not determine decision value.** Each point is a nuScenes frame with non-zero perception gain and decision value (YOLOv8s 320→640, oracle geometry), on asinh axes. Shaded quadrants mark sign disagreement: 32.6% of frames for the braking controller and 50.6% for the learned planner.

Table 14: Perception-level gains do not fix the sign of decision value. nuScenes, YOLOv8s 320→640, oracle geometry, shared threshold, frames with non-zero decision value. Corr.: Spearman correlation between gain and decision value. $P(\text{harm}|G{>}0)$: share of frames with positive perception gain that escalation harms. Disagree: share of frames with non-zero gain and non-zero decision value whose signs differ. Per-mode thresholds are in Appendix D.

	q_{brake} (141 frames)			q_{plan} (1,179 frames)		
Perception gain	Corr.	$P(\text{harm}\|G{>}0)$	Disagree	Corr.	$P(\text{harm}\|G{>}0)$	Disagree
Exact (false-negative count)	+0.08	24.4%	25.0%	+0.02	50.6%	50.4%
False negatives + false positives	+0.07	15.7%	31.9%	+0.02	50.3%	49.8%
Combined error ($\mathcal{E}_5$)	+0.08	19.2%	32.6%	+0.01	51.0%	50.6%
Risk-weighted ($\mathcal{E}_{\text{risk}}$)	+0.11	23.8%	26.8%	+0.01	50.5%	51.0%

I Robustness Across Perception and Downstream Settings

Ego speed, a causal ego-motion signal with no perception content, separates from random on no held-out nuScenes cell at a 20% budget (Table 8), so ego speed alone does not explain the perception-level results on nuScenes. The signal is causal on every track: a backward difference over the previous 0.5 s on nuScenes, where a scene's first frame has no predecessor and takes zero, the OXTS forward velocity of the current frame on KITTI, and the current ego state's velocity on nuPlan. Ego speed is far stronger in a KITTI longitudinal setting, where braking demand scales as $v^2/(2g)$, so both the loss and the room to reduce it grow with speed; trivial baselines must therefore be reported per setting rather than dismissed. Across the 16 configurations below, false-negative gain never exceeds $\text{nDG} = 0.337$ and is negative in six.

J Downstream Systems and Hardware Details

Braking controller. q_{brake} computes the deceleration a^{req} needed to stop 2 m behind the most constraining obstacle that overlaps a corridor of half-width 1.2 m within 80 m, allowing 0.4 s of reaction time and taking the larger of the requirements at ego speed and at closing speed, capped at 9 m/s^2. It keeps speed for $a^{\text{req}} < 1.0$, decelerates at 2.5 m/s^2 for $1.0 \le a^{\text{req}} < 3.5$ and brakes at 6.0 m/s^2 above. With commanded deceleration a, previous command a_{prev} and the requirement $a^\star$ computed from the reference scene, the loss is

$$\mathcal{L}_{\text{brake}} = (a^\star - a)_+^2 + 0.12\,(a - a^\star)_+^2 + 0.02\,|a - a_{\text{prev}}| + 6 \cdot \mathbf{1}\big[(a^\star - a)_+ \ge 8\big], \qquad (10)$$

whose terms are the safety shortfall, over-braking, a change of command, and a shortfall large enough to count as a collision.

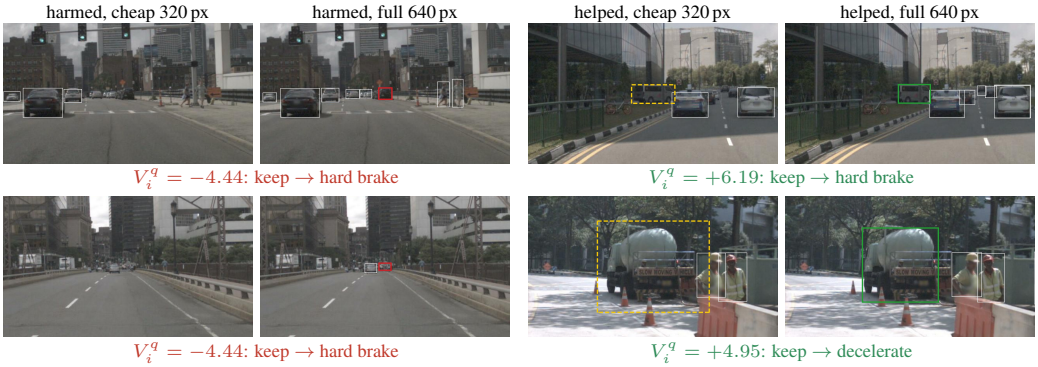


Figure 10: **Escalation adds detections the controller reacts to, and recovers the objects it needs.** Braking controller on nuScenes, monocular geometry; each pair is cropped around the object that decides the braking action. Left pairs: harmed frames in which the full mode adds a detection with no reference object (red) that triggers hard braking. Right pairs: helped frames in which the full mode detects an object (green) that the cheap mode misses (dashed yellow). Thin white boxes are the other detections. More examples and the selection rule are in Figure 11.

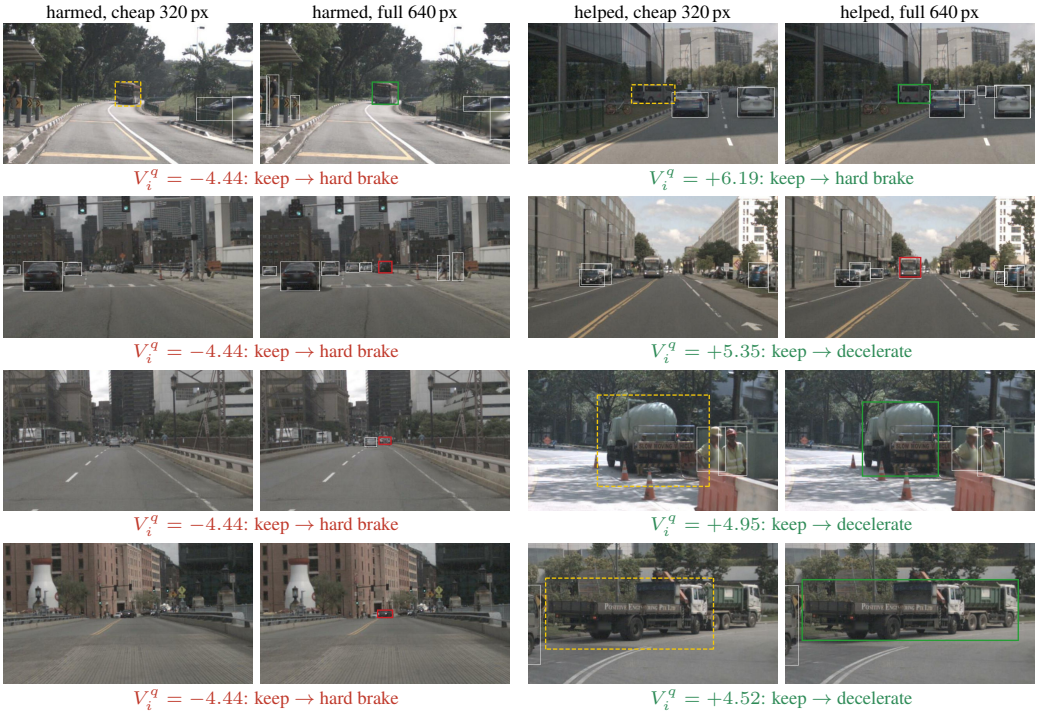


Figure 11: **Qualitative examples under the braking controller** (nuScenes, monocular geometry), extending Figure 10; colors as there, each pair cropped around the object that decides the braking action. The examples are the frames among the six most harmed and six most helped, at most one per scene, whose decisive change is an added detection or a recovered object; in the other four, escalation changes the estimated range of an object that both modes detect. The first harmed frame is harmed by a recovered object: the full mode detects a vehicle that the cheap mode misses, and the monocular lift places it inside the controller's corridor although its reference box lies outside it. In the second helped frame, the full mode's box around the bus overlaps the reference box by less than the matching threshold and is therefore drawn as an added detection.

Receding-horizon controller. q_{traj} evaluates seven longitudinal accelerations a from 0 to $-8\,\mathrm{m/s}^2$ crossed with five lateral offsets d from -3 to 3 m over a horizon $T = 3$ s with a 0.25 s rollout step, and selects the candidate of least cost under perceived geometry, with obstacles treated as stationary.

Table 15: nDG of the false-negative gain (the reduction in missed objects) and of a multi-metric diagnostic at a 20% budget, across 16 configurations. The diagnostic is an out-of-fold gradient-boosted model over perception-error features computed from the reference state, so it is not an upper bound over perception signals: in four configurations it falls below the false-negative gain beside it. It exceeds 0.5 only in the two aggressive-gap KITTI longitudinal cells and in nuScenes longitudinal under oracle geometry.

Configuration	Task	nDG of $G^{\mathcal{E}_{\mathrm{FN}}}$	Multi-metric diagnostic
KITTI/Y8 320→640 mono	longitudinal	0.162	0.672
KITTI/Y8 320→640 mono	lateral	−0.230	−0.118
KITTI/Y8 320→640 oracle	longitudinal	0.200	0.810
KITTI/Y8 320→640 oracle	lateral	−1.041	−0.231
KITTI/Y8 384→640 mono	longitudinal	0.152	0.388
KITTI/Y8 384→640 mono	lateral	−0.297	−0.273
KITTI/Y8 512→640 mono	longitudinal	0.171	0.213
KITTI/Y8 512→640 mono	lateral	−0.283	−0.764
KITTI/RT 320→640 mono	longitudinal	0.074	0.153
KITTI/RT 320→640 mono	lateral	−0.448	−0.206
KITTI/RT 480→640 mono	longitudinal	0.104	−0.005
KITTI/RT 480→640 mono	lateral	−0.091	0.138
nuScenes/Y8 320→640 mono	longitudinal	0.053	0.350
nuScenes/Y8 320→640 mono	lateral	0.337	0.245
nuScenes/Y8 320→640 oracle	longitudinal	0.171	0.556
nuScenes/Y8 320→640 oracle	lateral	0.266	−0.261

The selected candidate is re-simulated against the reference obstacles, and its loss is

$$\mathcal{L}_{\mathrm{traj}} = 10{\cdot}\mathbf{1}[\mathrm{collision}]{+}1.5\,c^2{+}\Delta_s^2{+}0.01\,a^2T{+}0.5\,d^2{+}0.05\big((a{-}a_{\mathrm{prev}})^2/4{+}(d{-}d_{\mathrm{prev}})^2\big), \quad (11)$$

where c is the shortfall of the worst lateral clearance below 1 m, clipped at 2 m, and Δ_s the progress shortfall relative to traveling at $\max(v, 2\,\mathrm{m/s})$ over the horizon, as a fraction of that distance.

Previous action. Both controllers depend on the previous action through a_{prev} and d_{prev}, which the receding-horizon controller also uses to plan. For decision values, both branches of input i take the action of all-cheap operation at the previous frame of the sequence, and none at its first frame, so V_i^q is the value of escalating input i alone and does not depend on which other inputs are escalated. Under this convention the change terms carry at most 2.9% of the total harm and 0.8% of the total benefit in every braking and trajectory setting; harm to the braking controller is 73–96% over-braking, and harm to the receding-horizon controller is almost entirely collision and clearance.

Loss and corridor sensitivity. Table 16 varies one parameter at a time: the braking controller's over-braking weight (0.06, 0.12, 0.24), collision penalty (3, 6, 12) and change-of-command weight (0, 0.02, 0.04), whose losses recombine exactly because its actions do not depend on them; its corridor half-width (1.0, 1.2, 1.5 m), which changes its actions; and the receding-horizon controller's collision (5, 10, 20), clearance (0.75, 1.5, 3) and progress (0.5, 1, 2) weights, changed in planning and loss together. The variations were chosen after the main results. The harm rate stays at 33–55% in every monocular-geometry setting, while ρ_q moves more and rises with the over-braking weight, for instance from 0.28 to 0.62 on nuScenes braking under oracle geometry.

Learned planner. We use the released planner on which PKL is built, at its published weights. It emits one BEV occupancy heatmap per future timestep over 16 steps, and we read its decision as the argmax path. q_{plan} is the mean Euclidean displacement, in meters, between that path and the logged future ego trajectory from the dataset's ego poses. We do not claim the logged trajectory is uniquely optimal. Frames whose four-second horizon extends past the end of a scene have no logged future and are excluded, leaving 2,655 of 3,376. $q_{\mathrm{plan}}^{\mathrm{self}}$ replaces the logged trajectory with the planner's own reference-box-conditioned path. The learned planner trails a constant-velocity predictor on q_{plan} and is included for its link to PKL; the other downstream systems clear that viability check.

Pre-escalation gates. Both gates regress V_i^q on 65 cheap-side features in four groups: confidence statistics, uncertainty (entropy, margin and suppression ambiguity), scene complexity (counts, areas, overlaps, image and motion statistics), and criticality from the monocular geometry of the cheap detections; on nuPlan they use 18 analogous features of the cheap tracks and the ego state. The ridge gate standardizes its inputs and selects its penalty from 13 values between 10^{-2} and 10^4 by

Table 16: Harm rate (%) and harm ratio ρ_q across loss and corridor variants, as the range over variants that include the default, over all units. Braking weights: over-braking, collision and change-of-command weights; corridor: half-width 1.0–1.5 m; trajectory weights: collision, clearance and progress weights in planning and loss.

	Braking weights		Braking corridor		Trajectory weights	
Setting	Harmed	ρ_q	Harmed	ρ_q	Harmed	ρ_q
nuScenes, oracle	35.5–36.2	0.28–0.62	35.5–36.5	0.33–0.46	—	
nuScenes, mono	38.1–42.5	0.28–0.55	39.2–41.6	0.37–0.41	—	
KITTI Y8 320→640, mono	32.6–35.3	0.11–0.28	32.7–34.6	0.15–0.22	46.0–48.8	0.27–0.32
KITTI Y8 384→640, mono	38.8–40.1	0.35–0.65	38.9–39.8	0.41–0.51	47.3–51.0	0.45–0.51
KITTI Y8 512→640, mono	42.9–43.4	0.51–0.72	42.6–42.9	0.59–0.71	51.6–54.5	1.10–1.15
KITTI RT 320→640, mono	43.4–44.3	0.61–0.79	44.0–44.8	0.67–0.81	47.1–49.7	0.74–0.79
KITTI RT 480→640, mono	40.7–43.2	0.68–0.79	42.2–43.1	0.72–0.86	42.9–45.2	0.57–0.60
KITTI Y8 320→640, oracle	23.2–25.0	0.04–0.15	22.2–26.3	0.07–0.10	2.7–3.7	0.00

Table 17: Jetson AGX Xavier FP16 TensorRT profiling of the KITTI detector modes on an idle board, one image per call. End-to-end time includes image decoding and preprocessing. Energy is per frame under the module convention, the GPU, CPU and SoC rails over idle.

Mode	GPU ms	End-to-end ms	Energy mJ
YOLOv8s 320	5.09	13.18	41.1
YOLOv8s 384	6.09	14.38	47.8
YOLOv8s 512	7.89	16.72	62.0
YOLOv8s 640	9.28	18.47	84.4
RT-DETR-l 320	16.82	25.31	397
RT-DETR-l 480	21.35	31.21	535
RT-DETR-l 640	30.97	43.43	816

leave-one-out cross-validation; the gradient-boosted gate uses histogram gradient boosting with depth 3, 200 iterations, learning rate 0.08 and at least 40 samples per leaf. Both are fit on the training and validation units of each track and scored once on its test units; the streaming thresholds of Appendix E are refit on the training units and calibrated on the validation units.

Monocular geometry. The deployed lift estimates each detection's range by fusing a ground-plane estimate from the bottom of its box with a class height prior, and takes lateral extent from the box edges; the exact formulas and constants are given in the released code. Geometry is reported in the ego frame. The camera-to-ego rotation enters the ground cue, whose horizon it corrects in both the cue and its fusion weight, and the translation moves the fused point and the bottom box corners to the ego origin; the calibration places the camera 1.08–1.14 m ahead of the ego origin and 0.31–0.33 m to its right on KITTI, and 1.70–1.72 m ahead on nuScenes. Time to collision, which is read from the change in box height, is unaffected. Applying the full rigid transform to the fused point instead gives a larger median range error in every range band and fidelity checked. The lift overestimates range by 1.2–1.3 m at 0–15 m and by 1.8–2.4 m at 15–30 m, because the bottom edge of a loose box lands on the wheel contact rather than on the near face. The two fidelities share this bias, differing by at most 0.12 m in every band up to 30 m and by 1.0–2.6 m beyond. The lift is not recalibrated against this bias, since that would tune it on the reference it is scored against. Oracle geometry gives each detection the reference box of highest image IoU above 0.5, not a one-to-one match, and inherits its range, lateral extent and time to collision; unmatched detections keep their monocular geometry.

Measured costs. End to end on the board, the cheap and full pipelines cost 12.71 and 19.35 ms on nuScenes and 13.18 and 18.47 ms on KITTI; the gate's feature vector costs 3.52 ms and ridge inference 0.41 ms, the MLP detection-list routers 0.65 ms in total, and the pixel router 1.55 ms of TensorRT inference after 3.3–8.2 ms of resizing and upload. On nuPlan front-camera images, the 320 and 640 px modes cost 14.4 and 23.6 ms, timed over preprocessing, inference and postprocessing without image decoding, and 69.5 and 143.8 mJ per frame under the module convention; the nuScenes modes cost 54.6 and 132.7 mJ. The gate features and models run on the CPU of the board; nuPlan has no separately timed feature pass, so its allocators are charged the nuScenes feature time.