PERSLIS RESEARCH

A Witness, Not a Detector

Checking what a game engine claims against the pixels,
and what the check is worth to the decisions that read it

An eye that answers one question per engine claim, *seen*, *not seen* or *cannot judge*, measured in Wolfenstein 3D and BZFlag, including a null result where it matters most

Perslis Research | September 2026

Research prototype · simulation and games · not a certified safety system

Abstract. A game engine knows where every object is; it cannot tell an agent whether the camera shows it. We study an eye built to answer only that: for each claim the engine makes (a kind, a distance, a bearing) it says *seen*, *not seen* or *cannot judge*. In a rule pilot for Wolfenstein 3D, the existing eye proved to be wired to nothing that decided about enemies: "in view" came from the tile map's centre-to-centre ray, and the eye's teacher taught only enemies the map already saw, so the eye had no enemy samples. A confirm-only witness step, a geometric teacher and an on/off switch repaired the wiring; over three 120-second blocks per arm the eye added about two sightings a minute and left deaths, kills and times hurt at parity. In BZFlag, free-search detection of tanks reached 3.5% precision. Judging the engine's claims instead, with predicted boxes, nearer-first occlusion, masked glows and boosted depth-2 trees that read as rules, passed a fixed gate under blocked 4-fold cross-validation on 7,863 frames of one match: recall 0.984 and 6 false confirms in 1,258 hidden claims (0.48%; 95% interval 0.18–1.04%), five of the six in one fold. Across 28 later matches on new random maps, live recall was 0.982. But the rule pilot's own line-of-sight test already avoided blind shots, so the check prevented none we could measure, and the fire it held cost shots at visible targets. Checking a claim proved easier to make reliable than searching the frame; what the check is worth is set by the decisions that read it.

1 Introduction

A game engine keeps an exact list of every object in the world: where each tank or guard stands, how far away, at what bearing. That list is not what the camera shows. A tank 200 m away may stand behind a building; a guard half out of a doorway may be in full view while a crude sight test calls it hidden. A pilot that fires on the engine's number alone can fire blind, and a pilot that trusts a crude sight test can ignore the guard that is shooting it.

This paper is about an eye built for exactly that gap. It does not search the frame for objects. It takes each *claim* the engine makes (a kind, a distance, a bearing) and answers one question about it: does this frame show it? The answer is one of three verdicts: *seen*, *not seen*, or *cannot judge*, with the reason. We call such an eye a **witness**: it testifies about claims someone else made, and it may decline to testify. Figure 1 contrasts it with a detector.

We report two linked results, the second motivated by the first.

A. An eye wired to nothing that decides (§3). Our commanded game runtime VDSG [2] already had an eye, taught by the engine while it plays, that labels doors and walls on the frame. In its Wolfenstein 3D lane we found that nothing that decides about enemies read it. The pilot's "in view" came from the tile map's centre-to-centre ray, and the eye's teacher taught an enemy only when that ray already called it in view, so the eye had no enemy samples at all and could never be a second witness for one. We fixed

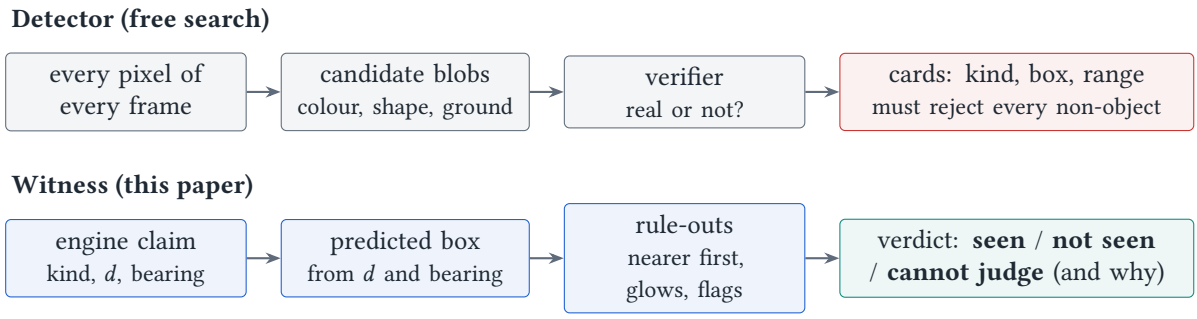

Figure 1. A detector must find objects anywhere in the frame and reject everything else; its errors are counted per frame. A witness is handed the engine's claims and judges each one in the one place it must be; its errors are counted per claim, and it may abstain. In BZFlag the free-search detector made 15.8 false alarms per frame (§4.2); the witness passed a gate of at most 0.5% false confirms per hidden claim (§4.4).

the wiring with a witness step that may only *confirm* a radar-listed enemy or pickup, a teacher driven by screen geometry, and an eye on/off switch with a per-arm ledger. Measured on one floor, the eye contributed about two sightings a minute and changed nothing we could measure: parity.

B. A witness for engine claims (§4–§6). For BZFlag, a 3D tank game in which our rule pilots, three language-model drivers and the game's own AI share one recorded arena [4], we first built a detector. Free search reached 3.5% precision: the dark stone walls looked like tanks. We replaced it with a game-agnostic witness, `vcard-eye`, that judges each engine claim: it predicts where the claimed tank must be on screen, rules out what cannot be judged (a nearer object, a glow over the box, the engine's own flags, point blank), and decides with boosted depth-2 trees whose paths read as IF…THEN rules. It passed a gate fixed in advance under blocked cross-validation, with a thin margin, and its live recall over 28 later matches stayed close to the cross-validated one. Then we measured what it is worth to the pilot, and found that in this arena the pilot's own line-of-sight test had already avoided the blind shots the eye was built to prevent.

The lesson we draw is stated only as far as these two cases support it:

> **A witness that checks a claim was easier to make reliable than a detector that searches. What the check is worth is set by the decisions that read it, and has to be measured there.**

1.1 Contributions

C1. A diagnosis, a repair and a null result (§3). We show how an eye taught by the engine could learn nothing about the one object class that mattered, repair it with a confirm-only witness rule and a geometric teacher, and measure eye-on against eye-off in alternating blocks on one engine: parity, reported with its confounds.

C2. A game-agnostic witness eye (§4). Box prediction from distance and bearing with curves in $1/d$, nearer-first occlusion, glow masking at a reach measured from the pixels, engine flags and three abstention kinds, a boosted-tree verifier that reads as rules, aim read from the pixels, a standard teacher-sample format, an onboarding loop and a Model Context Protocol server that onboards a new game the same way.

C3. An evaluation protocol and its results (§4.4–§5). Four-fold cross-validation over contiguous blocks, a decision threshold set on out-of-fold scores at half the gate's false-confirm limit, a gate that an eye must pass before any pilot may act on it, and an evolution curve that re-grades every stage of the eye on the same frames and folds. The curve is not monotone, and we report why.

C4. A measurement at the decision (§6). Live recall over 28 matches on new maps, blind shots with the check on and off, and every held shot graded against the engine's depth buffer. In this arena the check prevented no measured blind shot, and its holds cost shots.

1.2 What we claim, and what we do not

We claim the measurements below at the sample sizes stated beside each. We do not claim that the witness eye is accurate on any game but the two it was measured on (BZFlag, and a synthetic game in its tests), that it generalises across maps beyond the 28 live matches reported, or that it makes any pilot play better: in both lanes it did not. The comparison with free search is not a controlled one: the detector was abandoned after two configurations, and we report it as the reason for the switch, not as a general result about detectors. Ideas are credited where they come from (§9); the verification-first framing is our own earlier work [1, 2]. Every number is ours and none has been replicated by a third party. Where a number survives only as recorded tool output in a session log, rather than as a committed evidence file, we say so (§8).

2 Terms, and the work this builds on

2.1 Terms

Definition 1 (Claim, teacher, label)**.** A *claim* is what the engine states about one object at the instant a frame is drawn: its kind, identity, distance d and bearing b from the camera. The *teacher* is the engine's own answer about that frame: in BZFlag, for each object, the screen box it covers and the share $v \in [0, 1]$ of that box actually in view, read from the depth buffer; in Wolfenstein, the class each screen column shows, read from rays over the tile map. The teacher is used to learn and to grade. It is never given to the eye at play time.

Definition 2 (Verdict, false confirm, blind shot, hold)**.** A witness returns one *verdict* per claim: *seen*, *not seen*, or *cannot judge*, each with a reason. In BZFlag a claim is labelled *seen* when its effective visible share is at least 0.5, *hidden* when it is at most 0.05, and *partial* otherwise; partial claims are neither taught nor counted in recall or false confirms. A *false confirm* is a hidden claim judged seen. A *blind shot* is a shot whose target the teacher showed at most 5% visible at the moment of firing. A *hold* is a moment when the pilot was on target and ready to fire but did not, because the eye had not confirmed the target.

A false confirm is the witness's version of a blind shot, and the gate in §4.4 is written in those terms: a pilot that fires only on confirmed claims can fire blind only when the eye falsely confirms, or when the target leaves view in the moment between the verdict and the shot.

2.2 Our earlier work

Four earlier reports set the frame, and we cite rather than repeat them. *TinkyVision* [1] described a vision layer as "perception and witness, not the control authority": it lets a model see and records what it saw, but does not decide what the model may do. We keep the word and narrow it: here a witness testifies about claims someone else made. *VDSG* [2] is a commanded runtime that decides what a game pilot may do from rules over facts, in which orders can only narrow the admissible set. It introduced an eye taught by the engine while it plays, a per-column colour labeller that works as a *second witness* to the engine's first, "whose disagreement is a number the operator can read"; it measured the eye's agreement with its teacher at 78–92% on DOOM and 79% on Wolfenstein 3D with four classes voting. *Rules at the Wheel* [4] describes the BZFlag arena used here: every tank its own game client, the same engine state for every driver, a new random world each match, our rule pilot, BZFlag's own AI, and a gate that holds fire unless the eye confirms the target. That paper measures the gate's effect on firing; this one describes the eye behind it and extends that measurement (§6). The fail-first paper [3] reported a DOOM memory at parity

(paired $t = 0.78$) as parity; we keep that practice.

3 A: the eye that decided nothing (VDSG, Wolfenstein 3D)

3.1 How the pilot saw enemies

VDSG's Wolfenstein lane runs the 1992 WL6 data under a tapped ECWolf engine [2]. Its situation report lists every enemy and pickup from the engine's object list (the "radar"), each with a distance, a bearing and a flag `in_view`. That flag was the tile map's own sight test: the thing must lie within the 90° field of view, and a segment sampled from the pilot's position to the thing's position must cross no blocking tile.[1] The rule policy engages only enemies with `in_view` set (`rules.visible`), and the goal ATTACK is admissible only when such an enemy exists [2]. A centre-to-centre segment is a crude test: a guard half out from behind a door frame, or standing in an open doorway the segment clips at the jamb, is reported "(unseen)" while it shoots.

The eye was supposed to be a second witness for exactly this. It was not. Its outputs reached the pilot at three places: a second witness that the tile ahead is a door (a reason to press USE), the range profile that sizes an escape when the pilot is stuck, and the cards drawn on the console page. None of them concerned enemies. Worse, the eye's teacher, which labels each screen column from the map, taught an enemy's columns only when the map already called that enemy in view. The eye could therefore learn only what the map already saw, and never become a witness for what it did not. A console run before the repair shows it plainly: the eye's colour samples stood at ceiling 4,686, floor 3,141, wall 27,068, door 7,757, **enemy 0** and pickup 0.

3.2 The repair

Commit `71b94d1` made four changes, each pinned by tests at every site where it applies (186 tests in the DOOM lane and 12 in the Wolfenstein lane at that commit, both loops, both consoles and both report texts).

A teacher driven by geometry. A sprite is now taught on every screen column its body covers, wherever it stands nearer than the wall that column's ray meets, whether or not the map's sight test passes. Within 38 seconds of a console restart the eye held 1,043 enemy colour samples; after 128 seconds, 8,912.

A confirm-only witness step. Each decision, every radar-listed enemy or pickup that the map calls unseen is checked against the eye's current cards (`doom_floor/witness.py`). With b_e, d_e the thing's radar bearing and distance and $\phi = 90°$ the field of view, the thing is marked seen by the eye when

$$|b_e| \le \frac{\phi}{2} \;\wedge\; \exists k \;:\; \mathrm{kind}(k) = \mathrm{kind}(e),\;\; \mathrm{conf}(k) \ge 0.5,\;\; |b_k - b_e| \le 6°,\;\; |d_k - d_e| \le 0.5 \max(d_e, 1), \tag{1}$$

taking the card k nearest in bearing. The step can only turn "unseen" into "seen" for something the radar already lists and the screen could show. It cannot invent a thing, remove one, or act off screen, so a false positive can at worst make the pilot aim early at a listed enemy, never at an empty wall. It does widen what the pilot may engage: a flipped enemy makes ATTACK admissible. That is a change in the facts the admissible set is computed from, not an order; VDSG's orders still only narrow.

A switch and a ledger. With the eye off, the pilot does not look and does not learn, and the cards, the door witness and the range profile are blank, so the navigator and the stuck rule fall back to the map. A per-arm ledger credits time to the arm that was on and counts per-decision changes (deaths, kills, times hurt, stuck and at-a-door decisions, map coverage, eye sightings) rather than reading level counters, because

[1] `wolf_floor/state.py`, lines 81–90 and 102, at commit `71b94d1`.

Table 1. Eye on against eye off, Wolfenstein 3D MAP01, skill 3, three 120-second blocks per arm, alternating on one engine. Counts marked ≈ are per-minute rates times minutes played. Coverage is the highest share of the map walked so far in one shared session and cannot separate the arms; the Wolfenstein tap reports no damage dealt.

	eye on	eye off	
time played (s)	359.9	360.4	
decisions (per second)	2,780 (7.72)	2,913 (8.08)	eye arm decides 4.5% less often
deaths	7 (1.17/min)	6 (1.00/min)	per block: 2, 3, 2 against 3, 1, 2
kills per minute	2.5 (≈15)	2.0 (≈12)	
times hurt per minute	6.17 (≈37)	4.99 (≈30)	decisions on which health fell
stuck share of decisions	5%	8%	
at-a-door share of decisions	36%	35%	
coverage (not comparable)	30%	33%	the last block was eye-off
eye sightings per minute	2.17 (≈13)	0	unseen → seen, Eq. (1)

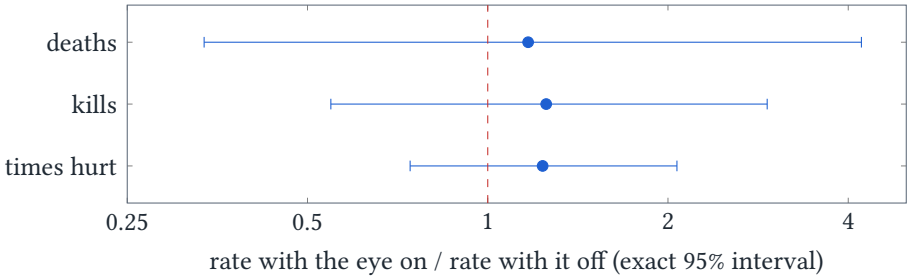

Figure 2. Every interval spans 1: parity within noise. Deaths 1.17 (0.34–4.21), kills 1.25 (0.55–2.93), times hurt 1.24 (0.74–2.07); exact conditional tests on the event counts give $p = 0.79$, 0.57 and 0.40. The tests treat events as independent, which in one game they are not, so they are a check on "no difference", not a finding.

the map's kill counter survives a respawn on one engine and resets on the other [2]. A driver alternates eye and blind blocks on one engine, so both arms play the same floor with the same respawns.

First observations. Over 75 polls of the running console after the repair, the map called an enemy in view in 20, the eye had an enemy card in 44, and the witness step changed a verdict in 1. When the two agreed they agreed closely: an eye card at −23.2° and 3.2 m (confidence 0.6) beside the map's own sighting of the same guard at −23.5° and 3.23 m.

3.3 Eye on against eye off

The measurement ran `python -m wolf_floor.ablate --level MAP01 --skill 3 --block 120 --blocks 6`: six alternating 120-second blocks on one engine, eye first, three per arm. Table 1 is the ledger it printed; Figure 2 shows the three event rates as eye-to-blind ratios.

3.4 What the numbers say

Parity. The eye now contributes to decisions: about two sightings a minute that make ATTACK admissible where the map's segment said unseen. It does not measurably change the outcome. With three blocks per arm on one floor, the deaths per block overlap completely, and every rate ratio's interval spans 1. We report this as parity, not as a small win for either arm.

Confounds. Four orphaned ECWolf engines, left behind by earlier console restarts, were running at full tick rate through the whole measurement, as was the live console's own engine: six engines in all on the machine, counting the run's own. They were found and killed when the run finished, and the consoles have since been changed to close their engine when they are stopped (commit `7693b88`). Both arms shared

the contention because the blocks alternated, but the engine runs in real time and the eye's look costs time, which may be why the eye arm decided 4.5% less often. The column agreement between eye and teacher, a running share since the session started, fell from 0.854 (38 s, 1,043 enemy samples) to 0.77 (128 s, 8,912) and ended the measurement at 0.577 with 23,833 enemy samples. VDSG reported 79% with four classes voting and no enemy class [2]; enemies are a harder class, and the drop is what we would expect, but we did not isolate it. Likewise, the eye's own range error over this run was 26% (range constant 38.3 from 600 samples), against the 7% (constant 33) in the VDSG report; we did not isolate the cause.

The repair was necessary: an eye with no enemy samples cannot testify about enemies. It was not sufficient to change what the pilot achieves on this floor. Section 7 returns to why.

4 B: a witness for engine claims (vcard-eye)

4.1 The teacher

BZFlag is an open-source 3D tank game; in our arena every tank is its own game client with its own driver [4]. A patch to the client, FloorEye, makes the engine a teacher. At up to ten instants a second, inside the renderer and just before the buffer swap, it reads the tank's own frame and depth buffer and writes a *pair*: the frame, and the truth about it. For every other tank it projects the tank's oriented box (eight corners) through the same view and projection matrices the scene was drawn with, and reports the screen box, the distance, the bearing, and the visible share: the on-screen share of the box times the fraction of a 5×5 grid of depth samples in the box that are not occluded by anything nearer than the tank's own depth span; it also writes the tank's silhouette from the same depth test. Shots and explosions are reported too, as the billboards they are drawn as. They are blended and write no depth, so the teacher cannot measure what they cover; a shot's billboard radius is 2.5 m. Later teacher versions added what the eye needed as it was found (§5.3):

teacher	adds
1	tanks only
2	every shot in flight, ours marked
3	explosions; per tank, the engine's flags "inside a wall" and "seen through a teleporter field"
4	our own view blocked (our tank pressed into a wall); the clean HUD also hides the shot-reload bars
5	no lock-on or waypoint markers; metres of open ground straight ahead (built, not yet collected)

The field of view is read from the projection matrix itself; an early version took it from an engine call that returned the vertical angle in radians, which skewed every bearing. A collector turns pairs into standard samples (a PNG frame plus a JSON file of objects), keeping frames with a hidden tank claim at up to four a second and others at up to one. The eye reads nothing game-specific from a sample, which is what lets the same eye be onboarded on another game.

4.2 Why not a detector

The first eye searched. It turned each frame into a map of per-pixel colour log-likelihood ratios (tank against background, from 512 colour bins, the same quantiser as the VDSG eye [2]), kept connected blobs above a threshold that stood where a tank on the ground could stand, and read range from blob height. Graded on 1,000 held-out frames from 4,000 samples, it found 89.7% of visible tanks at **3.5% precision**: 15.8 false alarms per frame, with bearing error 1.37° at the 95th percentile. On inspection of six held-out frames, the false alarms were almost all segments of dark stone walls, near and along the horizon (Figure 3). A learned verifier over the candidates, fitted on 2,661 real and 38,726 false ones and cut for 99% precision, pushed its threshold to 1.0 and recall to 1.8%. Both grades used single held-out frames, which later proved to flatter an eye (§4.4); the true numbers were, if anything, worse. We stopped there, after two configurations. The free-search comparison is not a controlled study; it is why we changed the question.

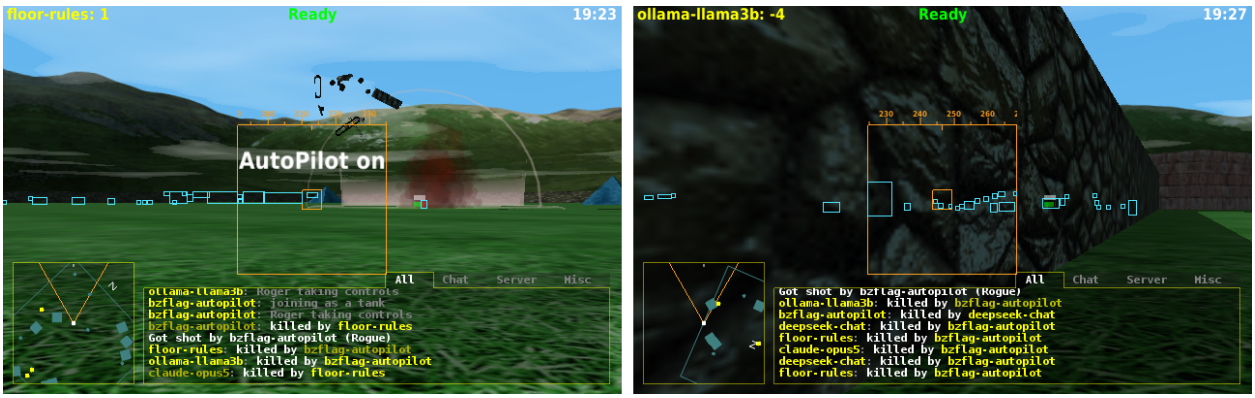

Figure 3. The first eye, free search, on two held-out teacher frames from an early match. Cyan: the detector's cards. Green and red: tanks the teacher marks visible and hidden. Orange: BZFlag's own targeting box. Most cards sit on the horizon band (left) and on a dark stone wall (right). These frames predate the clean HUD: the two grey bars right of centre are the shot-reload indicator (§5.3).

4.3 The witness, step by step

The witness (`vcard_eye/witness.py`) takes the frame and the engine's claims at the frame's instant: kind, identity, distance and bearing only, never the teacher's box or visible share. Claims of the witnessed kind (tanks) are judged; other kinds (shots, explosions) are not witnessed but are placed on screen as occluders.

1. Where to look. With image width w and horizontal field of view ϕ, the claim's column follows from its bearing, and the top, bottom and width of its box are curves in $1/d$ fitted by least squares on the teacher's exact boxes:

$$x(b) = \frac{w}{2}\left(1 - \frac{\tan b}{\tan(\phi/2)}\right) - \frac{1}{2}, \quad y_{\text{top}}(d) = a_0 + \frac{a_1}{d} + \frac{a_2}{d^2}, \quad y_{\text{bot}}(d) = c_0 + \frac{c_1}{d} + \frac{c_2}{d^2}, \quad \text{width}(d) = \frac{e_1}{d} + \frac{e_2}{d^2} + e_0. \tag{2}$$

Here e_0 is the 90th percentile of the width residuals. A single rule, height $= K/d$, was tried first; dominated by the many distant tanks, it drew near ones 20–25% too tall and up to 15 px too low, because a near tank's closest corner, not its centre, sets its bottom edge. On the 5,957 teacher boxes wholly inside the frame, the curves' median error is 0.35 px at the top and 0.29 px at the bottom. With $\phi = 60^\circ$ at 640×400, a tank at 50 m is predicted 62×26 px, at 150 m 26×9 px and at 300 m 17×5 px.

2. Nearer first. Claims are judged in order of distance. A claim whose predicted box is at least half covered by a nearer tank the eye has already confirmed is *not seen*: those pixels belong to the nearer tank.

3. Occluders. Every shot and explosion the engine reports and that is nearer than the claim is masked out of the claim's window, like the static HUD, over the area its glow really covers: an ellipse of s times the billboard's half-size plus 1.5 px, with s the 90th percentile of the glow's measured reach (§5.3). The shipped eye masks shots at $s = 0.72$ (from 2,373 clear glows) and explosions at $s = 0.87$ (2,412). A shot's screen position follows its distance (curve error 0.31 px); an explosion's does not (median error 10.8 px), so explosions are placed by the box the engine projects. If the glows cover 70% or more of the claim's box, the verdict is *cannot judge*. Because glows write no depth, the teacher's visible share overstates what the camera shows behind one, so every label uses the effective share $v_{\text{eff}} = v\,(1 - g)$, where g is the share of the box the masked glows cover.

4. What the eye does not judge. A claim the engine itself flags (inside a wall, behind a teleporter field, or our own view blocked) is *cannot judge*, as is any claim nearer than 12 m, where the box overflows the view and the depth teacher is unreliable, and any window at least half washed out by glare (luminance ≥ 235).

5. Look. In the window (the predicted box plus a margin of 0.4 box heights and 2 px) the eye computes 14 readable features: how much of the box is the kind's colour and how much more than the ring around it, whether that colour runs on sideways like a wall band, texture, contrast, a bright highlight inside the box and whether it is brighter than anything around it, glare, the densest patch of the kind's colour, the halo around the brightest spot, and the share of bright pixels. A 3×6 grid over the box adds each cell's brightness relative to the ring and its share of the kind's colour: 36 more, 50 in all. The colour model is a 512-bin log-likelihood ratio from the pixels the teacher's silhouettes mark as tank against background pixels; a pixel counts as tank colour above a threshold chosen by cross-validation from $\{0.5, 1.0, 1.5\}$ (1.5 was chosen).

6. Decide. A verifier gives the probability that the claim is seen. It is a gradient-boosted ensemble [20, 21] of 250 depth-2 trees (logistic loss, class-balanced, shrinkage 0.1, 32 quantile bins per feature, 80% row subsampling, fixed seed), written in plain NumPy and stored as JSON. A depth-2 tree reads as a rule. The strongest in the shipped eye is:

> IF a bright highlight inside the box > 0.522 AND the tank colour in row 1, column 3 of the box > 0 THEN $+1.12$ toward *seen*.

A logistic score was used first. Its evidence can only add, and the eye's hardest confusion is an interaction: a saturated highlight is a tank's best evidence when a dark body surrounds it, and a shot's glow when a bright halo does. A depth-2 tree expresses exactly that AND; the package's unit test requires the trees to learn a two-feature AND to over 97% accuracy. The *cut* is set on out-of-fold scores (each scored by a verifier that never saw its block) as the lowest score that confirms at most 0.25% of hidden claims, half the gate's limit, and never below 0.5. A cut chosen on in-sample scores had let 1.06% through against a 0.5% target. The shipped cut is 0.947. In the shipped eye, 63% of the trees' split gain is the in-box highlight, 13% its excess over the ring, 9% texture, and 7% the 36 grid cells together.

7. Aim. A confirmed claim becomes a card whose bearing is read from the pixels: the centroid of the kind's colour in the window.

4.4 Grading, and the gate

Neighbouring frames of one tank's recording are seconds apart and nearly alike. Grading on single held-out frames therefore grades an eye on frames it has effectively seen, and the early eyes' grades were flattered by it. The grade that counts uses **4-fold cross-validation over blocks of 60 consecutive frames**: samples are sorted by tank and time, cut into blocks, and the blocks rotate through the folds, so every claim is judged exactly once by an eye that never learned from its block, and every part of the match appears in every fold. The code comments call a block "about 20 seconds". In the data graded here a block spans a median 69.7 s of play (10th–90th percentile 54–80 s), because the collector saved about 0.85 frames a second per tank; the held-out runs are longer than intended, which makes the grade stricter rather than looser. Grading applies the play-time decision exactly (nearer first, masks, abstentions, the cut) on stored crops that are verified by a unit test to give the same features as whole frames.

Table 2. The gate run: teacher 4, 7,863 frames of one match, 4-fold cross-validation over blocks of 60 frames. The gate passed. Intervals: Wilson for recall, Clopper–Pearson for false confirms.

check	result	gate
recall, seen claims ≥4 px	**0.984** (3,896 of 3,959; 95%: 0.980–0.988)	≥ 0.98
false confirms, hidden claims	**0.48%** (6 of 1,258; 95%: 0.18–1.04%; one-sided 0.94%)	≤ 0.5%
hidden claims tested	1,258	≥ 400
the eye's own abstentions (washed out)	0 of 3,959	≤ 2%
cannot judge: a nearer glow covers it	602	not a miss
cannot judge: point blank (< 12 m)	65	not a miss
partly visible, not graded	322 (118 of them judged seen)	
aim error, 95th percentile	0.55° from the pixels; the engine's bearing 0.13°	

The gate (fixed before the gate run)

An eye is *armed* only if, pooled over the folds: **recall** on seen claims predicted at least 4 px tall ≥ 0.98; **false confirms** ≤ 0.5% of hidden claims; at least **400 hidden claims** tested; and the eye's own abstentions (washed out) ≤ 2% of seen claims. Claims the camera cannot show for a reason the engine knows (a nearer glow, point blank, an engine flag) are counted apart, not as misses. The store refuses to load an unpassed eye for a pilot, and the live eye never arms witness firing without a passing report. The thresholds were committed (`30401f2`) before the gate run.

4.5 Onboarding a game

Any lane that writes standard samples can be onboarded without per-game code. An onboarder checks what the samples hold (seen and hidden claims, occluder kinds, teacher versions, one resolution), learns a round in a separate low-priority process whenever 500 new samples have arrived, and stops when the gate passes. A Model Context Protocol server exposes the loop to any agent session through eight tools: `eye_check_samples`, `eye_onboard`, `eye_job`, `eye_jobs`, `eye_stop_job`, `eye_status` (the grade, the gate and every rule the eye applies, in words), `eye_games`, and `eye_judge`, which refuses an eye that has not passed its gate unless explicitly overridden. So far one real game (BZFlag) and one synthetic test game have been onboarded; a second real game has not.

5 B: results

5.1 The gate run

The gate run learned from every teacher-4 sample on disk: **7,863 frames, all from one 25-minute match**, seen through the cameras of six tanks, three driven by our rule pilot and three by BZFlag's AI, at 1,185–1,402 frames per tank (0.79–0.93 per second). The match's frames hold 6,206 tank claims within 350 m (the teacher labels 4,782 seen, 1,323 hidden and 101 partial), 7,468 shot claims (3,483 of them our own) and 5,777 explosion claims. No tank claim carried an engine flag and no frame had our view blocked, so those two rule-outs were never exercised by this data. The folds used 132 blocks, five of which straddle two tanks' recordings. After the eye's own rule-outs, 3,959 seen and 1,258 hidden claims were graded. Table 2 gives the pooled grade, Table 3 the four folds.

The margin is thin. The pooled grade passes. It would not pass if one fold stood alone: fold 1 falsely confirmed 5 of 387 hidden claims (1.29%) and fold 4 found 0.970 of seen ones (Figure 4). The six false confirms are not six independent events: two fall in one frame. The 95% interval on the false-confirm rate reaches 1.04%, twice the limit, and the interval on recall reaches 0.980, the limit itself. The folds come from

Table 3. Per fold and per size. Neither the recall nor the false-confirm limit holds in every fold on its own: fold 4 misses the recall line and fold 1 the false-confirm line. No graded claim was predicted under 4 px tall.

fold	frames	recall	missed / seen	false confirms / hidden	rate
1	1,980	0.987	12 / 952	5 / 387	1.29%
2	1,980	0.997	3 / 1,075	1 / 214	0.47%
3	1,980	0.981	18 / 948	0 / 351	0
4	1,923	0.970	30 / 984	0 / 306	0
size		**recall**	**missed / seen**	**false confirms / hidden**	**rate**
small, 4–7 px		0.980	58 / 2,948	5 / 1,000	0.50%
medium, 8–19 px		0.995	4 / 775	1 / 173	0.58%
large, 20 px and over		0.996	1 / 236	0 / 85	0

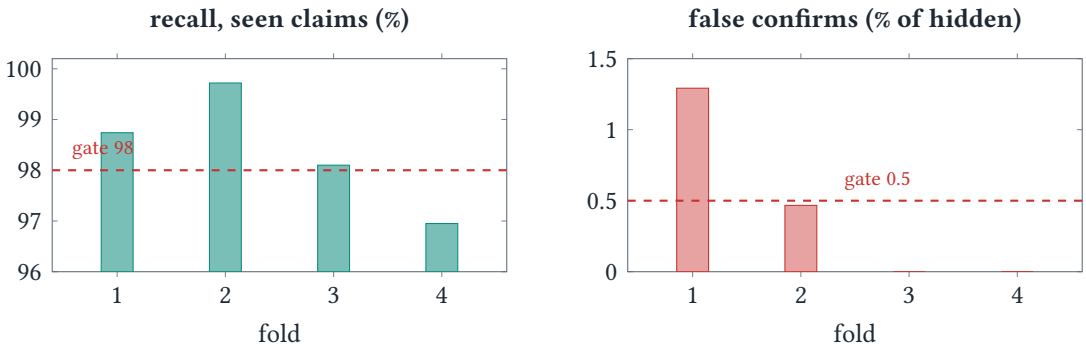

Figure 4. The margin is thin. Pooled, the eye passes; fold by fold, recall ranges 0.970–0.997 and false confirms 0–1.29%. Five of the six false confirms fall in fold 1, two of them in a single frame. The Wilson interval on pooled recall reaches below the gate line (0.980).

one match, so they share a map, a lighting and six tanks' habits; blocks keep neighbouring frames apart, but not those things. And the last design choices (the layout grid, and measuring glow reach only on glows at least 6 px in radius) were made by cross-validated results on these same frames and folds, so the gate run is not a fully independent test of the final design. The live matches on new maps (§5.4) are the check against that.

What the six false confirms were. Figure 5 shows each at the moment of the verdict. Two, in the same frame, are from a camera pressed flat against a wall the engine did not report as blocking our view, so the whole window is wall texture. One is the red lock-on bracket BZFlag draws around its AI's target even when a wall hides it; teacher 5 hides the markers, but no teacher-5 samples have been collected. One is a frame streaked white. Two are predicted boxes on distant dark wall, the free-search confuser, now rare but not gone. These readings are by inspection, not measured causes.

Aim. The bearing read from the pixels is less accurate than the engine's own (0.55° against 0.13° at the 95th percentile). The witness exists to say *whether* to believe the engine's claim, not to replace its geometry.

5.2 How the eye got there: the evolution curve

The numbers measured on the day each idea arrived came from different teachers and from single-frame grading, and stitched together they would draw a curve that is partly the data changing. So every stage of the eye was re-learned and re-graded on the same 7,863 frames, with the same folds and with labels

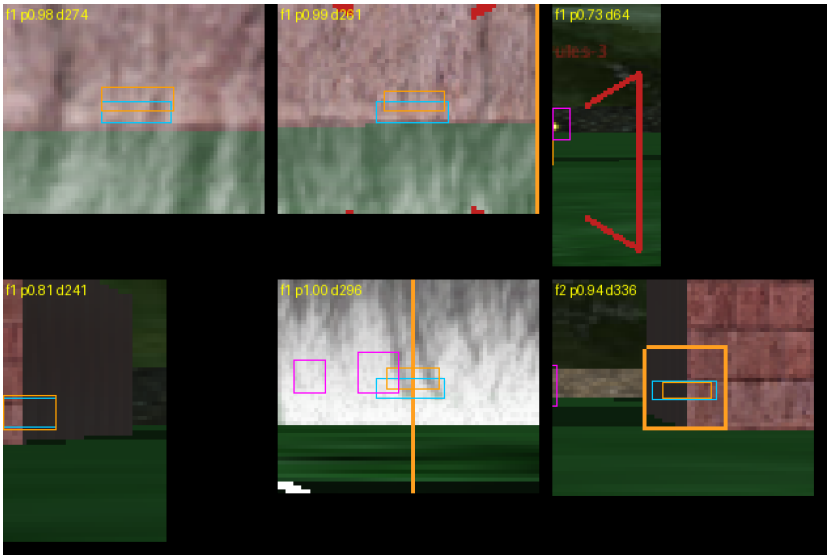

Figure 5. The six false confirms of the gate run, each cropped around the eye's predicted box (cyan). Teacher boxes: orange for a hidden tank, green for a visible one, magenta for shots and explosions. Labels give fold, the verifier's probability p and the claim's distance in metres. Top: the camera pressed into a wall (two claims, one frame), and a lock-on bracket. Bottom: a dark wall, a frame streaked white, a dark wall.

Table 4. Every stage re-learned and re-graded on the same frames and folds (3,958 seen claims ≥4 px, 1,315 hidden). Only the last stage passes.

stage	what it adds	features	recall	false confirms	
first witness	colour and a highlight, one logistic score	10	96.1%	15 (1.14%)	fails
+ glow clues	a glow's halo; a highlight's dark body	14	79.5%	7 (0.53%)	fails
+ shots masked	glow pixels masked out	14	78.3%	2 (0.15%)	fails
+ engine flags	flagged claims not judged	14	78.3%	2 (0.15%)	fails
+ decision trees	AND rules instead of a sum	14	92.7%	3 (0.23%)	fails
+ turret layout	3 × 6 grid over the box	50	98.4%	6 (0.46%)	passes

fixed from the engine's geometry (Table 4, Figure 6). Each stage is today's code with the later additions switched off, not a checkout of the code of that day. This grading is stricter than the gate's in two ways. Every hidden tank counts, even behind a glow or with our view blocked, since confirming one is a blind shot whatever is in front of it. An abstention on a seen claim counts as a miss.

Glow clues hurt, then paid. The glow clues replace one feature (the brightest pixel in the window) with five: a highlight inside the box, its excess over the ring, glare, the halo around the brightest spot, and the share of bright pixels. Added to a logistic score they halved false confirms and cost 16.5 points of recall, because a sum of evidence cannot say "bright spot *and* dark body" as opposed to "bright spot *and* halo". The same fourteen features in depth-2 trees recovered recall to 92.7%. The 3 × 6 layout grid brought it to 98.4%: a far tank is a few pixels of dark body with a bright top, and only the layout says so. Decided on the same table and folds (with the glow reach then in use), the grid took recall from 0.952 to 0.989, small-tank misses from 144 to 37, and false confirms from 5 to 4 of 1,252. Masking shot glows cut false confirms from 7 to 2 at a cost of 1.3 points of recall. The engine flags changed nothing here, because no claim in these frames was flagged.

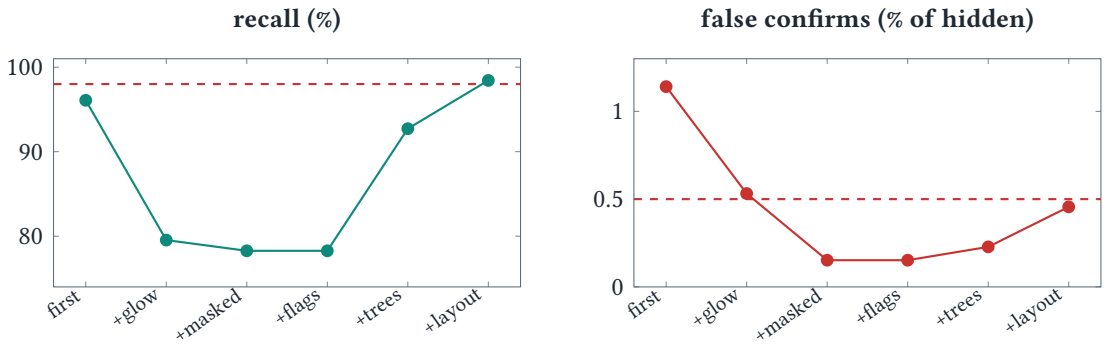

Figure 6. The curve is not monotone. Glow clues cut false confirms, but a logistic score cannot use them without losing 16.5 points of recall; decision trees recover most of it; the box layout recovers the rest. Dashed: the gate.

Table 5. Real glow reach against the teacher's billboard: 551 clear shot glows in frames of one earlier match. Edge: where brightness falls to 15% of peak above background.

distance	glows	half-max radius	edge radius	billboard radius	edge / billboard
30–60 m	18	8.0 px	10.5 px (0.86 m)	30.5 px	0.34
60–120 m	61	4.0 px	5.0 px (0.84 m)	16.0 px	0.31
120–200 m	139	2.0 px	3.0 px (1.00 m)	9.0 px	0.33
200–350 m	330	1.0 px	2.0 px (1.13 m)	6.0 px	0.33

The grading moves the curve. An earlier version of this grading left hidden tanks behind a glow ungraded; under it the first witness falsely confirmed 5 of 1,255 (0.40%), not 15 of 1,315 (1.14%). The eyes being graded were the same, so the ten extra false confirms are all hidden tanks behind a glow: the confuser the next stages were built to remove. The day-by-day grades are not comparable either: the first witness, graded on single held-out frames of older data, scored recall 0.916 and 2 false confirms in 188, and the first eye with $1/d$ curves 0.963 and 2 in 188; the first block-graded eye on teacher-2 data, recall 0.962 and 0.20% false confirms, failed the gate.

5.3 The findings that cost hours

Our own shot sat on the target. A glow flying straight away from the camera barely moves on screen, so our own shot sits on the very tank it was fired at for as long as it flies, including a tank that is hidden behind a wall. Before the teacher reported shots, the eye "confirmed" hidden tanks from our own shot's glow. Teacher 2 reports every shot, ours marked, and the witness masks them (step 3).

The glow is smaller than its billboard. The teacher reports a shot as the 2.5 m billboard it is drawn as. Masking that whole box over-masked: on the older teacher-2 and teacher-3 data the eye could not judge 444 of 2,691 seen tanks (16.5%). Measured on 551 clear glows in the pixels, the radius at which a glow's brightness falls to 15% of its peak above background was 0.31–0.34 of the billboard's radius in every distance band from 30 to 350 m (Table 5). Masking at the measured reach (its 90th percentile, 0.57 on that data) cut the unjudged seen tanks to 227 of 2,691 (8.4%). The learner now measures the reach per occluder kind on its own data: 0.72 for shots and 0.87 for explosions in the gate run.

"=" on the walls was the HUD. Two short grey bars that sat on distant walls, and on far tanks' boxes, were first read as the "interdimensional lights" BZFlag draws where a tank is partly inside a wall. They were the shot-reload indicator, drawn at a fixed spot right of the screen centre (`HUDRenderer.cxx`, lines

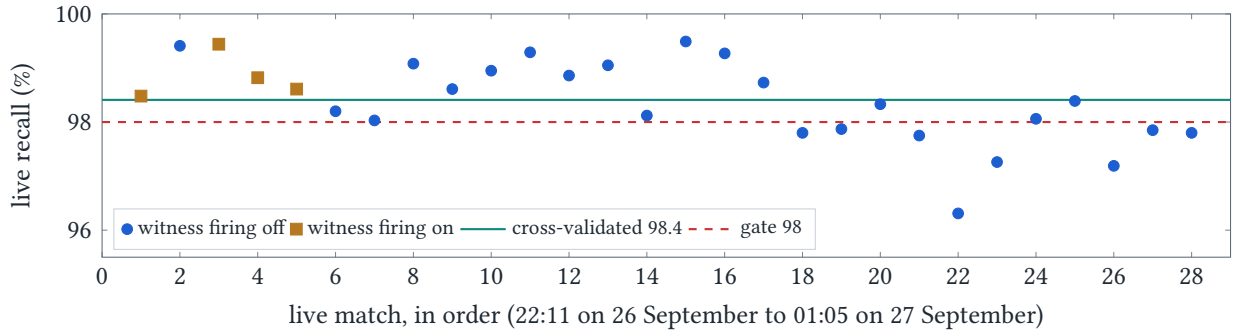

Figure 7. Live recall per match. Pooled 0.982 (124,852 of 127,086 seen verdicts), median 0.984, range 0.963–0.995. Eight matches fall below the gate line, all among the learning loop's A/B matches from 00:19 on 27 September; its eight A/B matches before that stayed above it. The eye itself did not change.

1750–1752). A static-pixel HUD mask misses it because it changes length as the gun reloads. Teacher 4's clean HUD hides it.

The depth teacher cannot see glows. Shots and explosions are blended billboards that write no depth, so a tank half behind a glow reads fully visible to the depth test while the camera shows half of it. Every label therefore uses the effective visible share $v(1-g)$ (step 3). Two smaller engine facts had to be measured rather than assumed: an explosion's screen size is not a function of its distance (the $1/d$ law missed by a median 10.8 px), and the engine call first used for the field of view returned the vertical angle in radians.

5.4 Live, over 28 matches on new maps

The gate grades frames of one match. The live eye judged every frame each tank's client wrote, about seven a second per tank, and graded itself against the teacher in the same pair by the gate's own definitions. From the first match after the live grade was aligned with the gate's (22:11 on 26 September) to the cut-off of the arena paper (the match that ended 01:05:50 on 27 September), **28 matches** ran with the live eye: 98.5 minutes of play and 250,966 judged frames, each match on a new random map [4]. Ten matches set three of our rule tanks against three BZFlag AI tanks; eighteen were the pilot-learning loop's A/B matches, two tanks with the current pilot and two with a candidate against two BZFlag AI tanks. The eye judged every tank's view, the AI tanks' included. The eye was not re-learned in that period: its saved profile is byte-identical to the gate run's, and its learning history holds that one round.

Live recall pooled to **0.982**, against the cross-validated 0.984 (Figure 7); per match it ranged from 0.963 to 0.995, with eight of 28 matches below 0.98. The eight are all A/B matches of the pilot-learning loop from 00:19 on, while the loop was changing its pilots; the loop's eight earlier A/B matches, with the same lineup, all stayed above 0.98. The eye did not change. We did not investigate whether the pilots' new ways of engaging explain the drop. The live eye falsely confirmed 37 of 162,185 hidden verdicts (0.023%) and could not judge 20,288 claims. The two false-confirm rates are not comparable. Live verdicts on consecutive frames repeat one situation, so the pooled counts are far from independent trials; and the gate's samples were deliberately enriched with hidden claims, while live hidden claims were not selected for difficulty. The honest reading is narrower: across 28 maps the eye had never seen, its recall stayed near the cross-validated value, and it did not fall apart.

Excluded. Two earlier live matches are left out, with reasons recorded in the session that ran them. In the first (22:02, recall 0.815, 51 false confirms), the eye still judged while its own tank was dead and judged claims beyond its 350 m training range; shot counters did not exist yet. In the second (22:07, recall 0.936),

Table 6. The 26 holds of match 20260926-225220 (witness firing on, 3 minutes), by what the teacher showed of the target at the moment of the hold. None was on a hidden target. For comparison, in the next match with a black box (witness firing off, 61 shots), every shot's target was visible, and the eye had not confirmed 6 of them.

target at the moment of the hold	holds	the eye's verdict
point blank, 0.2–1.3 m (no teacher reading)	12	cannot judge (all one encounter: one tank, one enemy, 1.75 s)
at least half visible	12	not seen 6, cannot judge 3, no fresh verdict 2, seen 1
partly visible (20%)	2	not seen 2
hidden	**0**	

the live grade still used looser visibility rules than the gate, and shots were graded against whichever enemy was nearest the gun line rather than the pilot's lead target. Both were fixed before the 22:11 match.

6 B: what the check is worth at the trigger

6.1 How the verdicts reach a decision

Each engine state a tank receives is annotated with the eye's verdict for every enemy it lists, if that verdict is at most 0.35 s old; otherwise the enemy is marked *no fresh look*. With *witness firing* on, our rule pilot fires only at an enemy the eye currently confirms, and otherwise holds ("on the sight: holding fire, the eye does not see it"); a language-model tank's shot is held unless the eye confirms the enemy nearest its gun line; BZFlag's own AI never consults the eye [4]. Every shot our tanks fire is graded afterwards against the teacher in the newest pair, on the pilot's own lead target. The detail that decides this section is upstream of the eye: the rule pilot engages only an enemy to which the engine reports a clear line of sight, reaching to within a metre of the enemy, inside shot range.[2] The witness sits on top of an engine test that already asks nearly the same question.

6.2 Shots, with the check on and off

The controlled pair. Two consecutive 3-minute matches with the same code, three rule tanks against three BZFlag AI tanks, differed in witness firing (and, like every match, in the random map). With it on, our tanks fired **48 shots, 0 blind**, and held fire 4 times; with it off, **58 shots, 0 blind**. The pilot's line-of-sight test had already avoided every blind shot.

Across the arena. Over every counted arena match to the tank-arena paper's cut-off whose results record graded shots, blind shots were rare either way: 1 in 197 with witness firing on (three matches) and 10 in 3,012 with it off (24 matches) [4]. The one blind shot with the gate on fell in the 22:07 match, graded under the earlier gun-line target attribution (§5.4). No match ran a language-model tank with the gate on.

6.3 Every hold, graded

From the 22:52 match on, a black box records every shot and hold with the eye's verdict and the teacher's visible share of the target at that moment. The one 3-minute match with witness firing on and a black box has 75 shots, every one at a target the eye had confirmed and the teacher showed visible, and 26 holds (Table 6).

Twelve of the 26 holds were one close encounter in which the eye, by design, would not judge a tank closer than 12 m, and the pilot held again and again at a tank it could hardly have missed. Twelve more were at targets the teacher showed at least half visible: the eye's misses, abstentions and stale verdicts. One hold

[2] `bzflag_floor/pilot.py` at commit `9ecdb4a`: an enemy is engageable when its line-of-sight distance is at least its distance minus 1 m.

came while the eye's latest verdict was *seen*, a mismatch we did not investigate. None was at a hidden target. In the witness-off match that followed, the eye was still judging: of 61 shots, all at visible targets, it had confirmed 55, called 5 not seen and had no verdict for 1. With the gate on, those 6 shots, 10% of the pilot's fire, would have been held although none was blind.

6.4 What this says

In this arena, in these matches, the check did not prevent a blind shot that we could measure, and it cost shots: at point blank, and wherever its recall fell short. That is not a defect of the eye's grade, which held live. It is the value of a second witness where the first is already right. The rule pilot's own line-of-sight test answers nearly the same question from the engine's geometry, and answers it well. What the check is for is a driver whose aim we do not write: a language-model tank that fires at a bearing it chose, or a platform where no engine test exists. Gating a model's trigger is built and pinned by a unit test, and it has not been measured in a match. Trusting the engine's line of sight inside 12 m, where the eye cannot frame a target, is the obvious repair for the point-blank cost; it has not been built.

7 Discussion

7.1 Why checking a claim was easier than searching

We did not run a controlled comparison, so this is an account of why the switch worked here, not a theorem. Three things changed when the eye stopped searching.

The question got smaller. The engine supplies the kind, distance and bearing, so the eye knows where the tank must be and how large it must look (Eq. 2): at 300 m, 17 × 5 pixels. That is too small to find by colour among dark wall texture, which is what the free-search eye tried to do, but enough to check in the one place the engine names.

The errors are counted per claim, and claims are few. The gate run's frames held 0.79 tank claims each (6,206 in 7,863 frames). A detector has to reject every wall segment in every frame; its own gate asked for at most one false alarm per hundred frames, and it made 15.8 per frame. A witness answers for the handful of places the engine names, and a false confirm is counted against hidden claims, the case that matters for a shot.

It may decline, for reasons that can be checked. Point blank, a glow over the box, an engine flag, glare: each abstention has a stated cause, and the gate forbids the eye from buying recall with its own abstentions.

None of this is new as an idea. Hypothesise-and-verify recognition checks a pose hypothesis against the image rather than searching for the object anew [8, 9, 10], and analysis by synthesis casts perception as testing hypotheses against the image [11]. What is particular here is where the hypotheses come from: the engine the pilot already trusts. The same fact bounds the witness. It cannot see anything the engine does not claim. In BZFlag the engine claims every tank; on a physical platform, where claims would come from other sensors or a map, a witness's recall could be no better than theirs.

7.2 An eye is worth what the decisions that read it are worth

Both lanes produced an eye that works by its own measure, and no measurable gain at the decision. In Wolfenstein, the eye went from zero enemy samples to about two sightings a minute that made ATTACK admissible, and deaths, kills and times hurt stayed at parity. In BZFlag, the eye passed a strict gate and held its recall live across 28 new maps, and the gate it arms prevented no blind shot we could measure, because the pilot's own line-of-sight test had already prevented them.

Neither result would have been visible from the eye's grade alone. A second witness can only pay where the first witness is absent or wrong, and how often that is, is a property of the pilot and the engine, not of the eye. Wolfenstein's centre-to-centre segment is wrong for a half-hidden guard, so there the witness found work, about two sightings a minute, but not enough, over three blocks per arm on one floor,

to move an outcome. BZFlag's line-of-sight test is right almost always (10 blind shots in 3,012 without the gate), so there the witness mostly found cost. The lesson we take is procedural: grade the eye, and then measure at the decision it feeds, with the eye switched on and off.

7.3 How much authority a witness should have

The two lanes give their witnesses different authority. In BZFlag the gate can only remove shots; like a shield on actions [19], it cannot make the pilot do anything the pilot would not otherwise do, and its cost is the fire it holds. In Wolfenstein the witness can make an enemy engageable that the map called unseen, so it widens what the pilot may attack. That authority is bounded by construction (only things the radar lists, only on screen, only with a matching card) and it changes a fact from which the admissible set is computed, never an order. Which design is right depends on which error is worse for the platform: a missed shot or a blind one. The two lanes also differ in how the eye earns its authority. The BZFlag eye cannot arm the gate until it has passed a fixed, cross-validated grade, and every rule it applies can be printed in words; the Wolfenstein eye learns online and has no such gate, only the bounds of the confirm rule. That is a gap in the Wolfenstein lane, not a design choice we would defend.

8 Limitations and threats to validity

- **One match behind the gate.** All 7,863 graded frames come from one 25-minute match, on one random map, through six tanks' cameras. The folds separate contiguous blocks, not maps or matches, so the cross-validated grade measures generalisation within that match. The 28 live matches (new maps, the same eye) are the only evidence beyond it, and they were graded by the live tally, which counts correlated frames.
- **Design chosen on the graded data.** The layout grid and the rule for measuring glow reach were chosen by cross-validated results on the same 7,863 frames and folds the gate run then graded. The gate's thresholds were committed before the run (`30401f2`), but the design was not frozen before the data, so the cross-validated grade is optimistic by an unknown amount.
- **A thin margin.** The pooled grade passes; the 95% intervals reach 1.04% false confirms and 0.980 recall; neither limit holds in every fold; two of the six false confirms share a frame. A second gate run on another match could fail.
- **Untested rule-outs.** No tank claim in the gate data carried an engine flag and no frame had our view blocked, so those abstentions were never exercised by the grade. Two of the six false confirms came from a camera against a wall that the engine did not flag. Teacher 5 (no lock-on markers, open ground ahead recorded) was built and never collected.
- **One real game each.** The witness eye has been onboarded on BZFlag and on a synthetic test game, not on a second real game; "game-agnostic" describes the code and the sample format, not a measured result.
- **The decision measurements are small.** The controlled BZFlag pair is two 3-minute matches; the graded holds come from one 3-minute match; no match ran a language-model tank with the gate on. The Wolfenstein comparison is three 120-second blocks per arm on one floor, with four orphaned engines and the live console's engine competing for the processor, a coverage metric that cannot separate the arms, and no damage-dealt count.
- **Evidence held in session logs.** Most numbers come from committed evidence files (the gate report, the saved eye, the evolution curve) or from recorded match files. Some survive only as recorded tool output in the two sessions' logs: the free-search results, the early witness grades, the glow-reach diagnostic and the layout-grid comparison, the list of the six false confirms, the Wolfenstein enemy-sample

counts and polls, and the Wolfenstein ledger, whose own JSON file was written to a session scratch directory that no longer exists. The paper's claims file lists each with its log line, and our extraction script fails if a pattern does not match, but these numbers cannot be re-derived from a committed file.

- **A coarse teacher.** The teacher's visible share is a 5×5 sample of the box against a depth tolerance of 0.75 tank lengths, so labels near the 0.5 and 0.05 thresholds inherit its coarseness, and both the gate and the live tally grade against it.
- **Interpretation by inspection.** The causes given for the six false confirms and for the free-search false alarms are readings of images, not measurements.
- **Tests not re-run.** The package's 18 unit tests, which onboard the synthetic game end to end, were not re-run for this paper: the test machine's system volume was full while we prepared it, and we did not want to compete with experiments running there. The lane's session log records all 18 passing shortly before the package's last commit (`74bd4ed`).
- **No third-party replication**, and the authors of the lanes and of this paper are one group.

9 Related work

Detection. Object detectors search: sliding-window cascades of boosted features [5], region proposals scored by a network [6], and single-pass grid predictors [7]. Viola and Jones built their cascade around the fact that almost every window is a negative, so a detector's false-positive burden scales with the windows it must reject. Our free-search eye is a small colour-statistics member of this family, and its 15.8 false alarms per frame are that burden. We make no claim about learned detectors, which we did not try.

Hypothesise and verify. Model-based recognition verified pose hypotheses against the image instead of searching for the object again [8, 9]; RANSAC turned the same loop into a general method for robust fitting [10]; analysis by synthesis frames perception as testing hypotheses against what is seen [11]. The witness is a verify-only stage whose hypotheses come from the engine.

Engines as teachers. Game engines have long supplied exact labels for learned perception: pixel-level ground truth from a commercial game [12], label and depth buffers in ViZDoom [13], and the engine-taught eye of VDSG [2]. Using information at training time that is withheld at play time is learning with privileged information [14], and a privileged agent has taught a sensorimotor one in driving simulation [15]. Our teacher is privileged in the same sense; what is added is a grade under which a learned eye must earn its authority before a pilot may act on it.

Monitoring perception and runtime assurance. Fault detection with analytical redundancy checks a sensor against a model of what it should read [16]; here the engine is the model and the camera the sensor. Recent work monitors perception systems through consistency between modules [17]. Runtime assurance places a trusted component beside or beneath an untrusted one, as in the Simplex architecture [18] and shielding for learned controllers [19]; the BZFlag gate is a shield on the trigger whose trigger condition is learned and graded.

Readable boosted trees. The verifier is gradient boosting [20] with split candidates restricted to feature quantiles, as in XGBoost's approximate algorithm [21], and with trees limited to depth 2 so that each reads as a two-condition rule, in the spirit of rule ensembles [22].

Evaluation. Cross-validation over blocks of dependent data follows the practice recommended for temporally, spatially or hierarchically structured data [23]. Intervals are Clopper–Pearson [24] and Wilson [25]; their behaviour for small counts is reviewed in [26].

Our own earlier work. TinkyVision [1] (a vision layer as witness, not control authority), VDSG [2] (the eye taught by the engine as a second witness; the admissible set), Fail-First Models [3] (parity reported as parity), and Rules at the Wheel [4] (the BZFlag arena and the firing gate).

10 Conclusion

We set out to give game pilots an eye they could trust, and measured two things: how good the eye is, and what it changes. On the first, the answer is encouraging within its limits. Turning the question from "find the tanks" into "is this claimed tank shown?" took the same colour statistics from a detector at 3.5% precision to a witness with a cross-validated recall of 0.984 and 0.48% false confirms, a grade that held at 0.982 across 28 new maps; the margin is thin and the gate data is one match. On the second, the answer is plain. In Wolfenstein 3D, repairing an eye that fed nothing that decides left the pilot's outcomes at parity. In BZFlag, a gate that holds fire until the eye confirms prevented no blind shot we could measure, because the pilot's own line-of-sight test already prevented them; judged on one match, it would have held about a tenth of the pilot's shots at targets that were in plain view, and it held fire twelve times through one point-blank encounter.

Both results point the same way. An eye's grade says what it can be trusted with; the decision it feeds says whether trusting it is worth anything. The next measurements are therefore at decisions where the first witness is weak: a language-model driver whose aim nobody wrote, the engine's line of sight trusted inside 12 m where the eye cannot frame a target, a second gate run on a different match and map, and a second real game onboarded through the same tool.

References

[1] Perslis Research. *TinkyVision: Let There Be Sight — continuous, low-latency vision for any model.* Technical report, 2026. https://research.perslis.com/tinky-vision

[2] Perslis Research. *VDSG: A Commanded Admission-Control Runtime for Autonomous Agents.* Systems paper, 2026. https://research.perslis.com/vdsg

[3] Perslis Research. *Fail-First Models: Failure Becomes Structure.* 2026. https://research.perslis.com/fail-first

[4] Perslis Research. *Rules at the Wheel: Tanks, Language Models and an Honest Loss.* Empirical study, 2026. https://research.perslis.com/tank-arena

[5] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. IEEE CVPR*, vol. 1, pp. I-511–I-518, 2001. doi:10.1109/CVPR.2001.990517

[6] R. Girshick, J. Donahue, T. Darrell and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. IEEE CVPR*, pp. 580–587, 2014. doi:10.1109/CVPR.2014.81

[7] J. Redmon, S. Divvala, R. Girshick and A. Farhadi. You Only Look Once: unified, real-time object detection. In *Proc. IEEE CVPR*, pp. 779–788, 2016. doi:10.1109/CVPR.2016.91

[8] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395, 1987. doi:10.1016/0004-3702(87)90070-1

[9] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212, 1990. doi:10.1007/BF00054921

[10] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. doi:10.1145/358669.358692

[11] A. Yuille and D. Kersten. Vision as Bayesian inference: analysis by synthesis? *Trends in Cognitive Sciences*, 10(7):301–308, 2006. doi:10.1016/j.tics.2006.05.002

[12] S. R. Richter, V. Vineet, S. Roth and V. Koltun. Playing for data: ground truth from computer games. In *Computer Vision – ECCV 2016*, LNCS, pp. 102–118, 2016. doi:10.1007/978-3-319-46475-6_7

[13] M. Kempka, M. Wydmuch, G. Runc, J. Toczek and W. Jaśkowski. ViZDoom: a Doom-based AI research platform for visual reinforcement learning. In *Proc. IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8, 2016. doi:10.1109/CIG.2016.7860433

[14] V. Vapnik and A. Vashist. A new learning paradigm: learning using privileged information. *Neural Networks*, 22(5–6):544–557, 2009. doi:10.1016/j.neunet.2009.06.042

[15] D. Chen, B. Zhou, V. Koltun and P. Krähenbühl. Learning by cheating. In *Conference on Robot Learning (CoRL)*, 2019. arXiv:1912.12294

[16] P. M. Frank. Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: a survey and some new results. *Automatica*, 26(3):459–474, 1990. doi:10.1016/0005-1098(90)90018-D

[17] P. Antonante, D. I. Spivak and L. Carlone. Monitoring and diagnosability of perception systems. arXiv:2011.07010, 2020.

[18] L. Sha. Using simplicity to control complexity. *IEEE Software*, 18(4):20–28, 2001. doi:10.1109/MS.2001.936213

[19] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum and U. Topcu. Safe reinforcement learning via shielding. In *Proc. AAAI Conference on Artificial Intelligence*, 32(1), 2018. doi:10.1609/aaai.v32i1.11797

[20] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, 29(5), 2001. doi:10.1214/aos/1013203451

[21] T. Chen and C. Guestrin. XGBoost: a scalable tree boosting system. In *Proc. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016. doi:10.1145/2939672.2939785

[22] J. H. Friedman and B. E. Popescu. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3), 2008. doi:10.1214/07-AOAS148

[23] D. R. Roberts, V. Bahn, S. Ciuti, M. S. Boyce, J. Elith, G. Guillera-Arroita, S. Hauenstein, J. J. Lahoz-Monfort, B. Schröder, W. Thuiller, D. I. Warton, B. A. Wintle, F. Hartig and C. F. Dormann. Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40(8):913–929, 2017. doi:10.1111/ecog.02881

[24] C. J. Clopper and E. S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934. doi:10.1093/biomet/26.4.404

[25] E. B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi:10.1080/01621459.1927.10502953

[26] L. D. Brown, T. T. Cai and A. DasGupta. Interval estimation for a binomial proportion. *Statistical Science*, 16(2), 2001. doi:10.1214/ss/1009213286

A Where the numbers come from

Every number in this paper is listed with its source in the paper's claims file (`CLAIMS.md`, shipped with the paper's sources). The sources are of four kinds.

1. **Committed evidence of the BZFlag lane:** the gate report, the saved eye and the evolution curve (`bzflag-floor/evidence/eye-gate/*-teacher4.json`; byte-identical to the copies in the eye store), the lane's code at the commits named in the text (`30401f2–74bd4ed` for the eye; `9ecdb4a` for the live wiring), and the teacher patches (`bzflag-src`, branch `floorlink`, `4eb684a–c7aada9`).
2. **Recorded matches:** each match's `meta.json`, `results.json`, `scoreboard.jsonl` and, from 22:52 on 26 September, per-shot black boxes; and the teacher samples' JSON metadata (7,863 teacher-4 files).
3. **The VDSG lane's code** at commits `71b94d1` and `7693b88`.

4. **Recorded tool output in the two sessions' logs**, for the numbers listed in §8, extracted by line number.

Four scripts shipped with the paper recompute everything from those sources without writing to them:

- `analysis/transcript_extract.py`: the session-log extracts;
- `analysis/eye_stats.py`: the gate, the saved eye, the evolution curve, the teacher data, the live matches, and the graded shots and holds;
- `analysis/wolf_stats.py`: the Wolfenstein ledger's derived counts and tests;
- `analysis/make_figdata.py`: the chart data, from the outputs of the other three.

No game, console, arena, learning round or evolution loop was run to write this paper.

B The shipped eye in words

The eye's own explanation, printed by its `eye_status` tool from the saved profile, lists every rule it applies. The five strongest tree paths, with the identical conditions of boosting's near-copy trees merged and their votes added, are:

1. IF a bright highlight inside the box > 0.522 AND the tank colour in row 1, column 3 of the box > 0 THEN $+1.12$ toward seen;
2. IF brighter inside the box than anywhere around it > 0.075 AND the tank colour in row 2, column $3 > 0$ THEN $+0.45$;
3. IF the tank colour in row 1, column $4 > 0.111$ AND mottled texture > 0.046 THEN $+0.43$;
4. IF much more of the tank colour inside than around > 0.082 AND mottled texture > 0.036 THEN $+0.33$;
5. IF brighter inside the box than anywhere around it > 0.075 AND strongly the tank colour > -1.74 THEN $+0.24$.

Row and column count the 3×6 grid over the predicted box from the top left. The first rule is an AND of two clues, which a logistic score cannot state: a highlight inside the box together with tank colour in the upper middle of the box, where we read a distant tank's turret.