OPENCV AI COMPETITION 2026 · POWERED BY AWS

FlyBot on AWS

Trap cameras that look before they count: OpenCV 5 and a guarded agent on AWS for fruit-fly monitoring.

Team: AGRHUB, makers of FlyBot · **Agronomy advisor:** MSc Trương Quốc Ánh (IAS, VAAS)

Live app: flybot.agrhub.com (press Guest access) · **Code:** github.com/agrhub/flybot-aws

State of the system: 27 October 2026

1 · PROBLEM

A fly count alone has three blind spots

- **Every photo is counted as it comes.** Glare, blur, a dark or infrared frame, a hand in front of the trap or a camera that no longer sees it all produce a number. One camera in our data filmed a building: every "fly" it reported was false.
- **A count is not a catch.** Flies stay in the trap; the signal is what changed since the previous photo.
- **The camera never reacts.** Nothing asks for a better photo, changes the exposure or the schedule, or shows a doubtful photo to a person.

Who it is for

Fruit growers and farm managers in Vietnam who watch the oriental fruit fly (*Bactrocera* spp.) with lure traps and yellow sticky cards, and the agronomists who advise them.

They use FlyBot's web app in Vietnamese or English, its chatbot and Telegram, with EZVIZ cloud cameras they already own or our open ESP32-S3 trap camera.

2 · SOLUTION

Perceive, decide, let a person confirm, act, look again

Perceive
OpenCV 5 on each photo and the previous one: count, quality, trap, change, occlusion, faces

→

Decide
an agent under guardrails: a new photo, other image settings, a review, an alert, or nothing

→

Confirm
a confirm card in FlyBot and on Telegram; nothing reaches a camera without a person

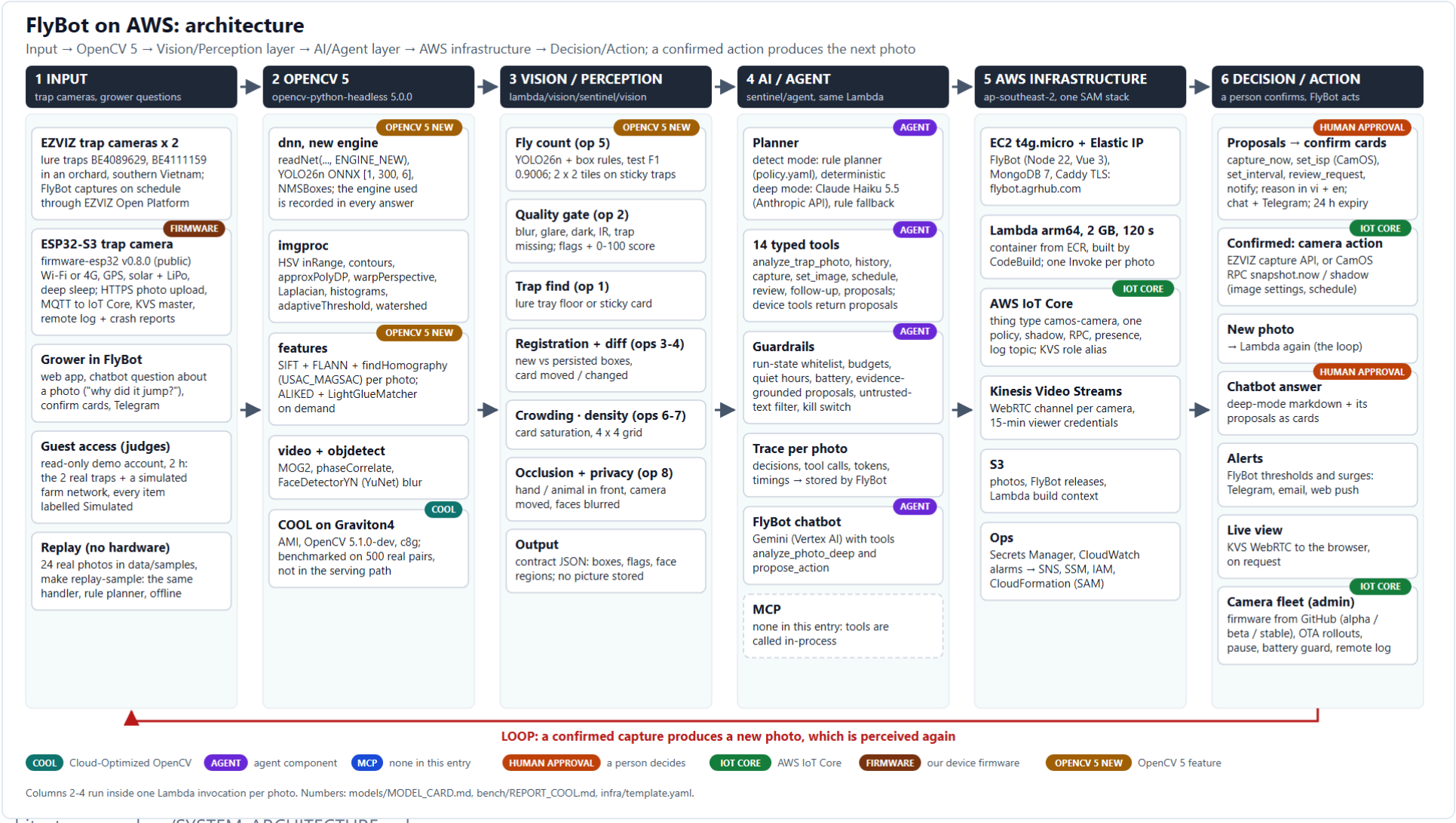
→

Act
FlyBot captures or changes the camera; the new photo starts the loop again

- Bad photos are **not counted**, and the reason is shown.
- Every decision leaves a **trace**: what OpenCV measured, what the agent decided, which tools it called.
- Growers can ask the chatbot "why did the count jump?": the answer comes from OpenCV's numbers, never from the pixels.

3 · ARCHITECTURE

Input → OpenCV 5 → Perception → Agent → AWS → Decision



docs/diagrams/architecture.svg; docs/SYSTEM_ARCHITECTURE.md

4 · OPENCV 5

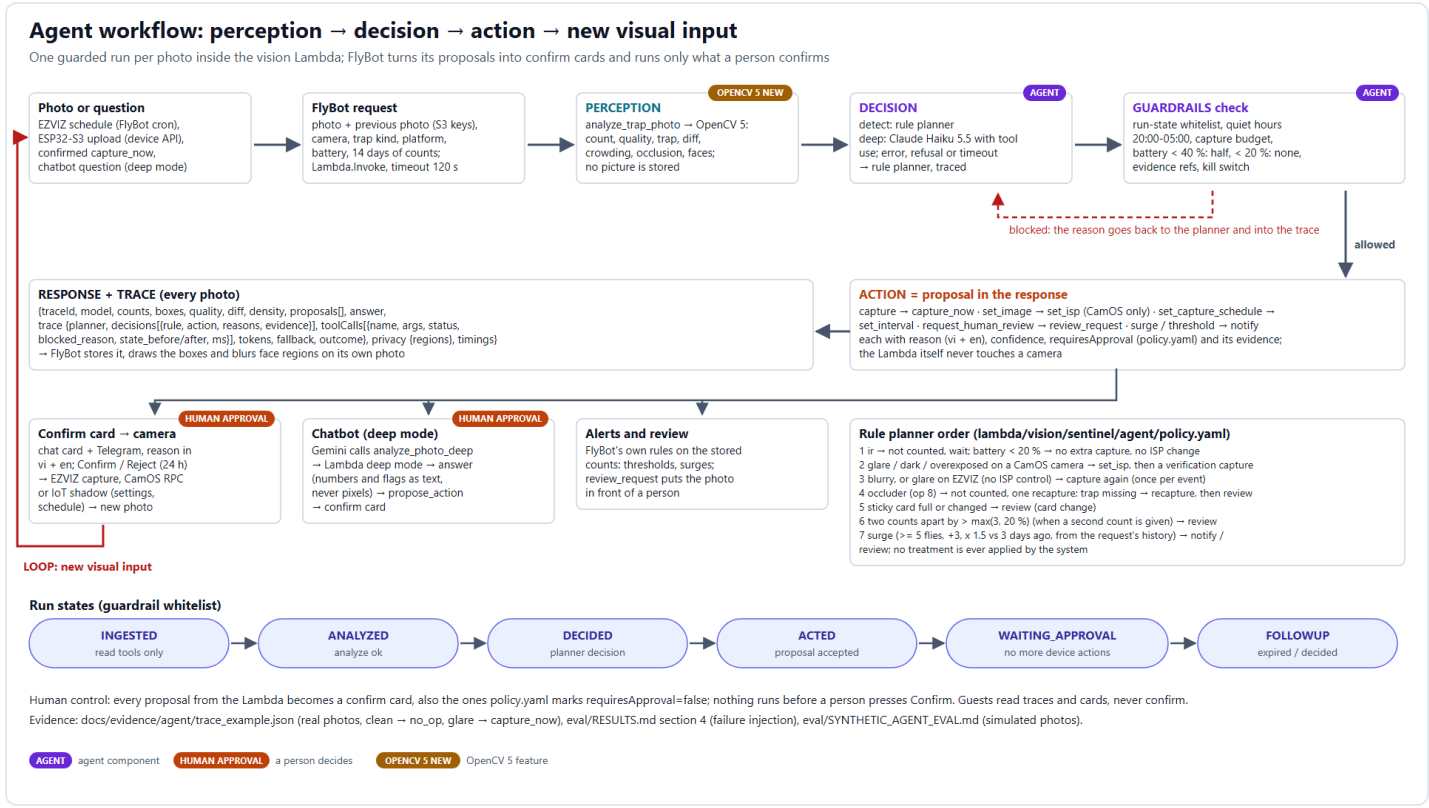
Eight vision ops in one Lambda call per photo

Op	OpenCV 5	Measured
Fly count	**dnn, new engine**: YOLO26n ONNX, `NMSBoxes`, box rules; 2 x 2 tiles on sticky cards	test mAP@50 0.953, F1 0.9006
Quality gate	Laplacian variance, histogram clipping, specular mask, IR detection	0 of 42 clean photos flagged
Trap find	HSV `inRange`, contours, `approxPolyDP`, `warpPerspective`	lure floor found 50/50
Registration	**features**: SIFT + FLANN + `findHomography(USAC_MAGSAC)`; ALIKED + LightGlue on demand	SIFT 69.2 ms on Graviton4
Change, crowding, density	`absdiff`, `adaptiveThreshold`, `watershed`, grid counts	crowding r 0.931 vs labels
Occlusion, privacy	MOG2, `phaseCorrelate`, `FaceDetectorYN` (YuNet)	occluder recall 0.94 (synthetic)

Learned: `ENGINE_AUTO` can fall back silently, so the engine used is recorded. Letterboxing to 640 shrinks sticky-card flies below 20 px; tiling the frame took recall from 0.07 to 0.82 on synthetic cards, without retraining.

5 · AGENT AND HUMAN APPROVAL

A guarded agent that only proposes



- Rule planner on every photo; Claude Haiku 5.5 for grower questions, with a rule fallback.
- 14 typed tools; guardrails: run states, budgets, quiet hours, battery, grounded proposals, kill switch.
- Every action is a proposal; only a person's Confirm runs it; cards expire after 24 h.
- No photo ever goes to an LLM: it gets OpenCV's numbers as text.

docs/REPORT.md section 8; lambda/vision/sentinel/agent/

6 · AGENTIC VISION EVIDENCE

OpenCV 5 output changes the next action

Trace: OpenCV 5 output changes the next action (real photos, camera BE4111159, 7 Oct 2026)

The vision Lambda's handler on 12 consecutive photos of one lure trap (make replay-sample, YOLO26n, rule planner); source docs/evidence/agent/trace_example.json

photo (UTC)	OpenCV count	FlyBot count	quality score	glare share	flags	decision (rule)	proposal	registration
04:30:06	0	0	99.86	0.0000	none	no_op	-	-
05:30:05	0	0	99.90	0.0008	none	no_op	-	failed
05:30:06	0	0	99.90	0.0008	none	no_op	-	ok, 72 inliers
06:00:06	0	0	71.28	0.1076	glare	quality.recapture + quality.awaiting_photo	capture_now	ok, 28 inliers
06:00:07	0	0	74.72	0.0672	glare	quality.recapture + quality.awaiting_photo	capture_now	ok, 255 inliers
06:30:05	0	0	100.00	0.0000	none	no_op	-	failed
06:30:07	0	0	100.00	0.0000	none	no_op	-	ok, 24 inliers
07:00:11	0	0	49.57	0.3761	glare	quality.recapture + quality.awaiting_photo	capture_now	failed
07:00:11	0	0	64.33	0.2022	glare	quality.recapture + quality.awaiting_photo	capture_now	ok, 258 inliers
07:30:05	0	0	100.00	0.0000	none	no_op	-	failed
07:30:10	0	0	97.59	0.0478	none	no_op	-	ok, 45 inliers
08:00:07	0	0	100.00	0.0000	none	no_op	-	failed

Orange rows: op 2 sees glare, so the planner asks for a new photo; the same camera minutes earlier and later gets no_op. FlyBot's count is its production model's, not a human count.

OPENCV 5 NEW

PERCEPTION (OpenCV 5)

photo 6ac5dfe69d7333beb125be13.jpg, 768 x 432
op 5 YOLO26n: 0 flies, DNN 99.3 ms (laptop)
op 2: issues ['glare'] → flag ['overexposed'], score 71.28
glare share 0.1076, clipped highs 0.0669, sharpness 679.84
op 1: lure floor found, covers 0.5684 of the frame
ops 3-4: registered (28 SIFT inliers), new 0, gone 0
op 8: person suspected False

AGENT

DECISION (agent)

planner: rule, fallback false, LLM calls 0
1 quality.recapture → capture
"['glare']: capture again"
2 quality.awaiting_photo → wait
the new photo arrives as a new event
event: photo_not_counted, why ['glare']
outcome: quality.recapture, quality.awaiting_photo

HUMAN APPROVAL

ACTION (guarded tool calls)

tool 0 analyze_trap_photo: ok, INGESTED → ANALYZED, 775.31 ms
tool 1 capture(recapture, main): ok,
DECIDED → ACTED, transport flybot_proposal
proposal capture_now: confidence 0.9, requiresApproval false
FlyBot: a confirm card + Telegram; Confirm runs the
EZVIZ capture → a new photo → this loop again
total 825.9 ms on the laptop (Lambda warm p50 980 ms)

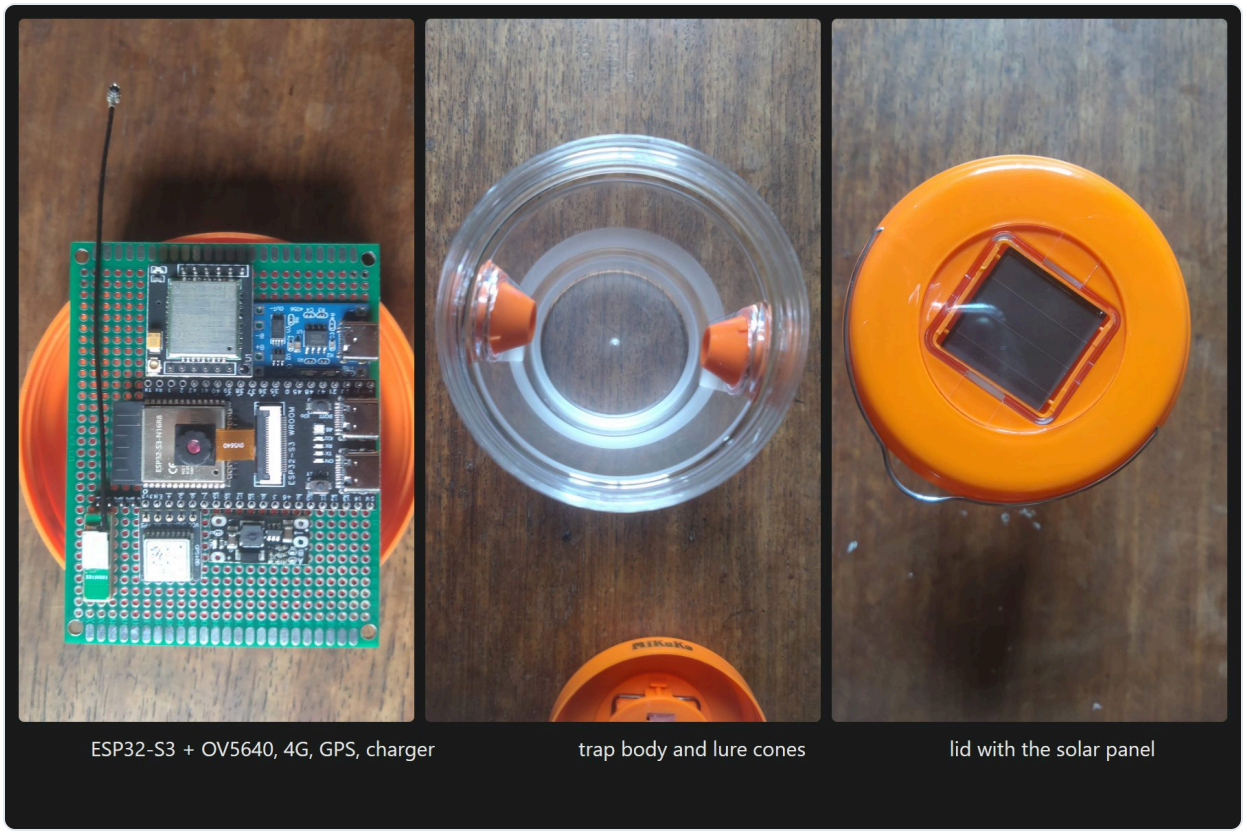
Not measured: whether these glare flags are right (no human look) and the confirm-to-new-photo time on AWS. The replay cannot take photos: the photos after a proposal are the camera's own.

Lambda warm p50: bench/results/lambda_latency.json. Small specks on the lid are missed by YOLO26n (models/MODEL_CARD.md), so FlyBot's production count is higher here.

12 real photos of one trap; docs/evidence/agent/trace_example.json; docs/diagrams/trace_example.svg

7 · OUR OPEN TRAP CAMERA

ESP32-S3 trap camera, managed from FlyBot



ESP32-S3 + OV5640, 4G, GPS, charger | trap body and lure cones | lid with the solar panel

- ESP32-S3 + OV5640 wide-angle, Wi-Fi or 4G (A7680C), GPS, LiPo cell and solar; open firmware v0.9.2.
- As built in a hanging solar trap with 4G, GPS and a flat LiPo cell: about US$39 per camera, housing included.
- Zero-touch registration through AWS IoT Core, QR claim, shadow and RPC control.
- Live view over Kinesis Video Streams WebRTC (JPEG 640x480).
- Fleet: OTA on alpha, beta and stable channels with a battery guard, remote log, crash reports.

docs/REPORT.md section 3.1; camos-esp32-trap README and docs/BOM.md (prices read on 17, 24 and 25 Oct 2026)

8 · AWS

One SAM stack, one command

Lambda arm64 container	OpenCV 5 pipeline + agent, one call per photo
ECR + CodeBuild	the Lambda image, built on ARM
EC2 t4g.micro	FlyBot (Node 22, Vue 3), Caddy TLS; MongoDB 7 or an external one
IoT Core	camera things, shadow, RPC, presence, log topic
Kinesis Video Streams	WebRTC live view, 15-minute viewer credentials
S3, Secrets Manager	photos (or an S3-compatible store), release pointer; every secret
CloudWatch + SNS	logs and alarms by email

./deploy.sh

creates or updates the whole system, ends with a health check and a latency measurement

980 ms

Lambda warm p50 per photo (cold 3,633 ms)

≈ 17 USD

a month for the demo; ≈ 0.046 USD per 1,000 photos

docs/REPORT.md section 5; infra/template.yaml; bench/results/lambda_latency.json; infra/COST.md

9 · RESULTS

What we measured

0.953
YOLO26n test mAP@50 on the OpenCV 5 DNN engine; same detections as ONNX Runtime

0.9006
F1 through the Lambda's OpenCV path (box rules raised it from 0.8788)

38 / 38
failure-injection runs, 0 false actuations (earlier build, fake devices)

0.07 → 0.82
sticky-card recall with 2 x 2 tiles (synthetic photos, simulated)

−4.6 %
COOL on Graviton4 vs the pip wheel per photo, 500 real pairs: we serve with pip

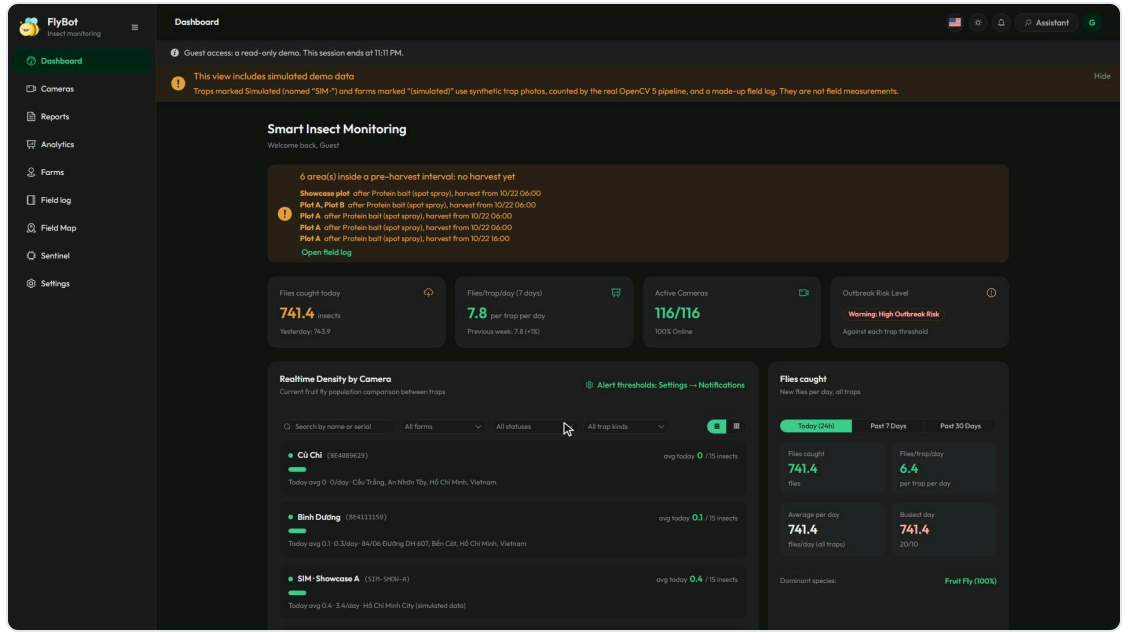
43 / 49
agent decisions judged correct by our agronomy advisor: 87.8 %, 95 % interval 75.8-94.3 %, 1 unsure; the 6 misses were needless recaptures, and after a glare limit for lure traps 1 of those 7 remains on the same photos

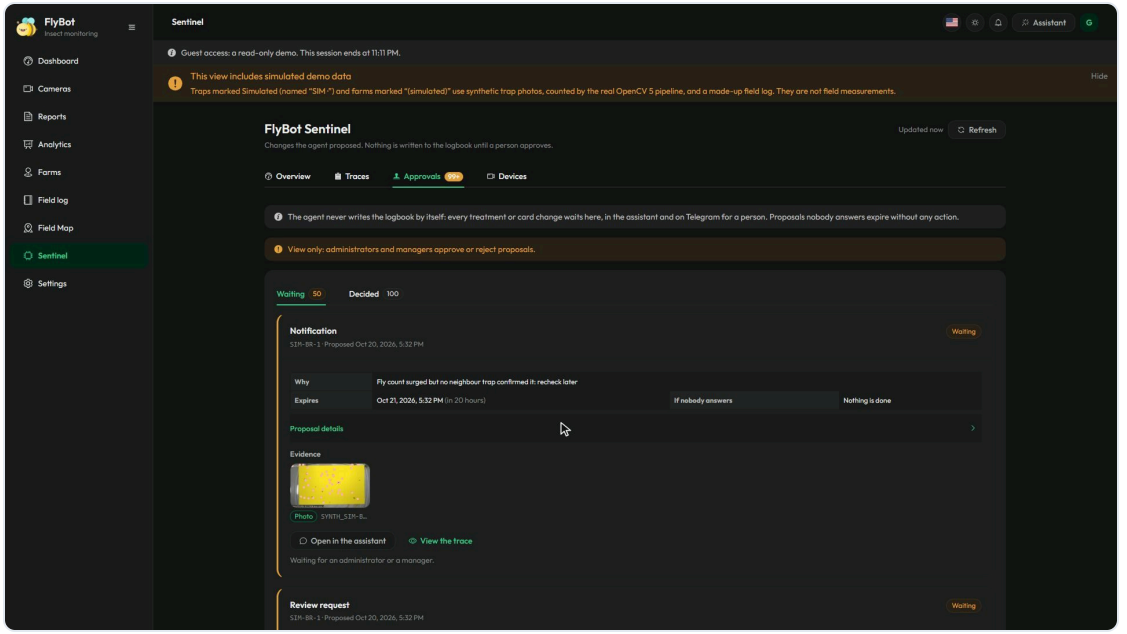
Not measured yet: counts against human labels (0 labels), a second reviewer, approval times on the live site, and the ESP32-S3 camera's battery and 4G in the field. Every number names its source file in the report; synthetic results are kept apart from real photos.

models/MODEL_CARD.md; eval/RESULTS.md; eval/SYNTHETIC_RESULTS.md; bench/REPORT_COOL.md; eval/results/expert_review.json; eval/results/glare_rule_ab.json

10 · TRY IT

Guest access: no account, two hours, read-only





flybot.agrhub.com/auth/login → **Guest access**: the two real traps, a labelled simulated farm network (114 traps on 47 farms, counted by the real Lambda), the Sentinel console with traces and proposals, and the chatbot. Offline: `make replay-sample` runs 24 real orchard photos through the Lambda's handler.

docs/REPORT.md section 3.2; README "Try it"

11 · LIMITS AND NEXT STEPS

What is not done yet

Limitations

- Little human ground truth: one reviewer's 50 verdicts (6 needless recaptures, 1 unsure; fix re-scored, not re-reviewed), 0 count labels.
- YOLO26n misses small flies in full lure-trap views; no real sticky-card photo yet.
- Agent evidence mostly from the earlier build and fake devices.
- ESP32-S3 field operation (4G, battery, solar) not measured; one region, one server.

Next

- The advisor reviews the new lure-trap glare limit on fresh photos; label 150 trap photos.
- Better recall on small flies in lure-trap views.
- The ESP32-S3 camera in the orchard on 4G and solar, with its battery measured.
- Yellow sticky cards; OpenCV 5's in-library VLM as a second opinion on doubtful photos.

docs/REPORT.md sections 9-10; docs/RESPONSIBLE_USE.md