

개요

fineAi RoboVision은 로봇 제어와 2D/3D 비전을 한 화면에서 다루는 독립 실행형 클라이언트입니다. 로봇 연결·티칭, 카메라 라이브/촬영, 궤적을 따라가며 촬영(Capture-along-trajectory), 품질 검사부터 **레시피 기반 운영, 공장(PLC) 연동, 비전 가이드 로봇(픽 보정·빈피킹·서보잉)**까지 지원합니다.

v1.2 좌측 메뉴는 "무엇을 하러 왔는가" 기준으로 **RUN / SETUP / INSPECT / GUIDE / SYSTEM** 5개 그룹으로 재편되었습니다. **Operator** 모드에서는 RUN 그룹만 노출됩니다(\$Operator · Engineer 모드).

그룹	화면	용도
RUN	Job Runner · Shiitake Harvest · Quality Dashboard · Fleet Dashboard · Robot Monitoring(Live · 3D View · Joint Status · Activity Stats) · Alarms & Events (v2.1)	운영·모니터링
SETUP	Robot Control · Teaching · Gripper (DG-3F-M) · Tool / Vision · Camera · Trajectory Capture	장비 셋업
PICK & PLACE	Object Model (Training) · Template Modeling (optional) · 3D Recognition · Comparison · 3D Pick&Place · 2D Pick Correction · Bin Picking (Legacy) · PA Engine Jobs (Legacy)	일반 3D 픽애플레이스 — 학습 → 인식 → 티칭·검증·실행 (2026-09-24 재구성)
SHIITAKE HARVEST	Harvest Lab · Harvest Teaching (운영은 RUN ▶ Shiitake Harvest)	표고 수확 = P&P 확장
INSPECT	PA Engine · Object Detection · Anomaly · Hole · Capture Quality · Model History	검사·AI
SYSTEM	Factory I/O (PLC/DB) · Reports/SPC · Digital Twin	연동·분석

NEW v2.3.0 Build261015 — 배지 3개 분리 · 빈손 판정 · J6 감김 방지 · 재탐지 단축 (2026-10-14~15 현장)

현장 아홉째·열째 날 작업입니다. 상세는 [docs/42](#) (릴리즈 노트)와 [docs/23](#) 10-14·10-15 항목에 있습니다. 아래 항목은 오프라인 재현과 단위 시험으로 확인했고, **실기 확인은 아직입니다**(맨 아래 확인 목록).

배지 인식 — 배지를 하나씩 분리합니다

- 배지를 **지지면에 누운 통나무**(규격 350 × Ø100)로 보고 맞춥니다. 표고가 뾰족하게 붙은 배지도, 방향이 다른 배지(세로·가로)도 각각 찾습니다. 로그에 **통나무 n개 (수평 · 지지면 투표)**가 찍힙니다. 그러면 `block_cyl_horizontal: false`.
- **배지 번호** = 로봇에서 가까운 줄 먼저 왼쪽 → 오른쪽, 그다음 먼 줄. 좌·우 2개 표준 배치는 번호가 예전과 같습니다. 배지가 여러 줄이면 예전에 저장한 작업 계획의 번호 뜻이 바뀌니 다시 확인하세요.
- **가로로 놓인 배지**의 윗면 표고는 배지 축에 직각인 앞/뒤에서 들어갑니다(이웃이 없으면 로봇 쪽 앞면). 11자도 그만큼 돌아갑니다.
- 같은 표고가 크기가 다른 상자로 두 번 잡히던 중복을 없앴습니다. 배지 검출 시간 3.3 → 1.2 초.
- 대 길이·갓 두께를 3D로 직접 잽니다(갓 돔 + 배지 원통 + 교점). 신뢰도가 낮으면 예전 규칙을 씁니다.

파지 판정 — 빈손은 성공이 아닙니다

- 빈손으로 닫았을 때 패드끼리 닿은 힘을 접촉으로 보던 문제를 고쳤습니다. 빈 닫힘 실측 기준이 있으면 틈이 기준 + 물체 인정 틈보다 작을 때 **힘과 무관하게 실패**입니다(단계 로그 "빈 닫힘 — ... 패드끼리 닿은 힘").
- 굵은 대(예: 24.8 mm)가 닫힐 거리가 모자라 빈손으로 판정되던 문제 — 11자 준비 벌림을 **그 표고의 3D 실측 대 굵기에 맞춰 넓힙니다**(설정값보다 작아지지 않음, 상한 40 mm). 로그 "패드 틈 n mm(대 굵기 3D 실측 x 기준)".
- 예압 상한 50°·기본 20°, 눌림 한도 3 mm(무른 대 보호), 최소 닫힘 5 mm, 유지·인출 확인은 두 증거(절대 틈 · 파지 시 틈 대비 감소) 중 하나.

⑤ 비틀어 따기 — 기준은 파지 전 11자 자세

수확은 ⑤ 프리셋을 그대로 실행합니다. 좌/우 기준선과 회전 중심을 이제 **파지 전 11자 준비 자세**로 정합니다. 대를 짹 물면 손가락이 서로를 지나쳐 좌/우가 뒤집히던 문제가 없어졌습니다.

J6(그리퍼 방향) — 감김과 360° 헛회전을 막습니다

- J6 명령은 항상 **현재 실측 J6에서 가장 적게 도는 쪽**, $\pm 180^\circ$ 를 넘어 감기는 쪽은 피합니다. $\pm 355^\circ$ 를 넘거나 한 번에 270° 를 넘는 명령은 보내지 않습니다.
- 되감기는 |J6| 300° 를 넘을 때만 합니다. 방향 전환이 뒤따르면 따로 되감지 않습니다(예: 되감기 360° + 전환 172° → 한 번 188°).
-  **작업 방향 자세로 이동**은 관절 이동 한 번으로 바뀌었습니다. 예전에는 직선 이동이라 15초 걸리고 J6를 한쪽으로 감아(+344°), 작업을 중단했다 다시 시작할 때 감김이 생기던 원인이었습니다.
- 배지 위 높이로 가는 첫 이동은 **위치 이동과 J6 회전을 동시에** 합니다. 표고 바로 위로 내려가는 이동은 높은 곳에서 J6를 먼저 돌립니다.

사이클 타임 · 동작 순서

- **로봇 먼저 출발**, 그리퍼 준비는 이동과 동시에 합니다. 손가락이 표고 근처로 가기 전에는 준비가 끝나 있습니다.
- **재탐지**: 수확 뒤 안전 높이에 도착하면 바로 다음 촬영을 시작하고, 촬영 직후 검출까지 미리 끝냅니다. 관찰 자세에서는 그 결과를 바로 씁니다 (예전에는 관찰 자세에서 처리하느라 약 7초 섰습니다). 직전에 있던 표고가 빠져 있으면 로봇이 가린 것으로 보고 관찰 자세에서 다시 찍습니다.
- 🎓 **촬영+검출**은 결과를 먼저 보여 주고, 도달(✓/X) 열린 '...' 로 두었다가 뒤에서 채웁니다.

🎓 Harvest Teaching 화면

- 수동 촬영+검출은 항상 새로 찍습니다(이전 장면이 보이던 문제).
- 크기 보정 기준을 추가해도 배지 정보가 유지됩니다(그 뒤 검증이 배지 왼쪽으로 들어가던 문제).
- 📁 **촬영 저장 체크** — 촬영할 때마다 원본 해상도 PNG 를 `data/harvest_captures/날짜/` 에 남깁니다(기본 꺼짐). 앱 로그에는 촬영 시작·완료 시각과 소요 시간이 항상 찍힙니다.
- 그리퍼 동작 세부설정의 로봇 측 수치 항목명이 한 줄로 보입니다.

로봇 설정 (RB10-1300 등)

- 도달 한계 입력 상한이 로봇 스펙 리치를 따릅니다(RB5-850 850 · RB10-1300 1300). 권장은 스펙의 약 85%.
- Robot Control 등록 패널 — 저장·삭제 뒤 닫히고 × Cancel 이 생겼습니다. 편집 저장 때 설치 기준 ·TCP·페이로드·충돌·속도 바가 지워지던 문제를 고쳤습니다.

현장에서 확인해 주실 점: ① 배지 3개·번호(좌1 우2 위3) ② 가로 배지 앞면 진입 간섭 ③ 빈손 11자 파지 → "빈 닫힘" 실패 ④ 대를 짝 문 상태의 ⑤ 비틀기 좌/우 ⑤ 굵은 대 준비 벌림 간섭 ⑥ J6 로그가 $\pm 180^\circ$ 근처 인지, 작업 방향 이동 속도 ⑦ 재탐지에서 관찰 자세 도착 직후 다음 후보 시작 ⑧ 로봇 먼저 출발 시 손가락 간섭. 처음에는 낮은 속도로 확인하세요.

NEW v2.2.0 Build261013 — 비틀기 회전 중심 · 높이 기준면 · 갓 두께 (2026-10-13 현장)

현장 여덟째 날 작업입니다. 수확 정확도에 직접 영향을 주는 네 가지를 바로잡았습니다. 상세는 [docs/23](#) 10-13 항목과 [docs/41](#) (릴리즈 노트), 사이클 타임은 [docs/40](#) 에 있습니다.

비틀어 따기 회전 중심 — ⑤ 메뉴와 같아졌습니다

"⑤ 그리퍼 메뉴에서는 밀동이 제자리인데 수확에서는 심하게 움직인다"는 보고의 원인입니다. 수확 경로가 회전 중심을 **파지점이 아니라 TCP 목표**에서 재고 있었습니다. 둘은 파지중심 보상만큼 떨어져 있어, 현장 기준 약 170 mm 어긋난 허공을 중심으로 돌았습니다. 회전 중심 모드가 **대 밀동 단일 중심**일 때는 좌우 모든 구간이 그 점을 돌기 때문에 밀동이 통째로 휘돌립니다.

- 중심을 손끝 패드 중점에서 재도록 고쳤습니다. ⑤ 메뉴와 완전히 같은 식입니다.
- 단계 로그에 "**비틀기 회전 중심 ... TCP 에서 N mm**" 가 찍힙니다. 140 mm 안팎이면 정상이고, 35 mm 처럼 작으면 보상이 빠진 것입니다.
- 회전 반경이 맞아지면서 비틀기 지령 속도도 함께 바로잡혔습니다.

높이 기준면 — 둥근 배지에 맞게

배지는 지름 100 mm 통나무인데 기준면을 **갓 중심 반경 80 mm 원의 중앙값**으로 잡고 있었습니다. 반경 80 은 배지 폭의 절반을 넘어 양 옆 비탈까지 섞이고, 중앙값이라 비탈 쪽으로 끌려 내려갑니다. 검출 88 mm 대 실제 100 mm, 자 실측 부착점 95 mm 와 맞는 설명입니다.

- 표본 창을 **배지 길이 방향으로 길고 폭 방향으로 좁게** 바꾸고 상위 백분위를 씁니다. 길이 350 mm 를 따라 높이가 일정하지 않으므로 그 표고 자리의 마루 높이를 잡습니다. 표고마다 그 자리의 실측 값을 쓰며, 배지 하나에 한 숫자를 쓰지 않습니다.
- 점이 모자라면 창을 두 번 넓히고, 그래도 없으면 예전 방식으로 물러납니다.

`height_block_local_round` 로 종전 복귀.

갓 두께와 파지점

- **갓 두께 기본값 30 mm**(수확기 표고). 종전 20 은 어린 갓 기준이라 파지점이 갓 밑면에 붙었습니다.
- 윗면 표고의 **파지점을 배지 면 기준**으로 잡습니다(배지 면 + 20 mm, 갓 밑면 5 mm·배지 면 8 mm 여유로 제한). 종전에는 갓 상면에서 고정 거리만큼 내려가서, 표고 키가 다르면 배지와 여유가 그대로 달라졌습니다. 🍌 로봇 측에서 끄면 예전 방식으로 돌아갑니다.
- **대 길이 3D 재측정** 선택 항목(기본 꺼짐). 검출 단계 프로파일이 3D 고리 높이를 기준으로 삼아 후보마다 다른 자로 재던 문제를 배지 면 기준 거리로 다시 잡니다. 표시 값과 대 짧음 게이트에만 반영되고 파지점은 바뀌지 않습니다.

크기 보정과 대 짧음 게이트

- 기준 표고가 **서로 모순**이면(같은 측정값에 다른 실제값) 높이 보정을 만들지 않습니다. 저장된 값도 다시 검사해 적용하지 않습니다. 지름 보정은 그대로입니다. 🍌 라벨에 "높이 보정 적용 안 함"과 이 유가 표시됩니다. 기준은 **서로 크기가 다른 표고를 하나씩** 넣으세요.
- 🎓 에서 **대 짧음은 거부**가 아니라 **확인**입니다. 경고 창에서 계속을 고르면 검증이 진행되고 기록에 "게이트 무시(운영자 확인)" 가 남습니다. 포인팅과 드라이런은 종전대로 묻지 않고 실행하며, 🍌 러너는 자동 제외를 유지합니다.

확인해 주실 점: 기준면이 올라가면 표고 높이와 대 길이가 그만큼 줄어듭니다. 갓 두께 30 과 겹치면 대 짧음 게이트에 걸리는 윗면 후보가 늘어납니다. 손가락이 갓 아래로 들어갈 자리가 없다는 뜻이므로 정상 동작이지만, 첫 검출에서 표의 높이·대 길이를 자 실측과 한 번 대조하시고 필요하면 높이 보정과 게이트 값을 조정하세요.

v2.1 (Build261008 이후, 2026-10-12 현장) — 수확 표고 사진·데이터 · 표고 높이 정확도 · fineAi 프리셋

현장 일곱째 날(10-12) 작업입니다. 설치본 빌드 전 변경이며 상세는 [docs/23](#) 10-12 항목과 [docs/39](#) (인식 정밀화 로드맵)에 있습니다.

-  **모니터링 대시보드(:8770/ledger)** — 수확된 표고의 ID·사진·데이터: 요약·수확 이력 표에 ID(대상/개체/track/후보 번호), 사진(갓 주변 크롭, 클릭하면 원본), 높이(출처), 배지·면, 위치(베이스 좌표, 갓 상면), 자세(기울기·축) 열이 추가됐습니다. 행을 선택하면 상세에 사진 + 항목표(갓 지름·높이·대길이·갓 두께 실측·갓 상면/파지점/축(베이스)·파지 오프셋·도달·결과·힘·등급) + 원본 JSON 이 보입니다. 사진은 [data/harvest_images/YYYYMM/](#) 에 저장되고  러너(독자/관제)· 검증 모두 기록합니다.
- **표고 높이(배지면→갓 상면) 정확도:** 높이는 배정 배지의 윗면/옆면 기준이 기본이고(자 실측으로 확인), 3D 고리 값은 배지 면 기준과 ± 15 mm 안일 때만 씁니다. 배지 윗면 위 표고의 갓 축이 수평에 가깝게 잘못 잡히면(옆면·대가 섞인 PCA) 배지 법선(중력) 축으로 갓 기하를 다시 계산합니다. 같은 표고 중복 검출 제거, 탁자 높이 물체 제외.  후보 표의 높이 셀 툴팁에 출처·메모(배지 윗면 기준값·고리값·보이는 대 끝·재계산 사유)가 나옵니다.
-  **파지 티칭** 그룹 이름이 "fineAi 프리셋 → 대상 구간 규칙" 으로 바뀌었습니다. 후보 번호 라벨이 이미지 위/아래 가장자리에서 잘리지 않습니다.
-  **Hand-Eye:** 골프공(흰 구) 방식 권장(RMS 0.4 mm), 자동 다자세 캡처 빠른 모드·속도, 계산·저장 완료 표시, 저장 샘플 재검증 때 폴더 방식 자동 전환.

v2.1 Build261008 — 작업 순서 계획 · 수확 모션 보정 · Hand-Eye 검출 수정 · 지지면 (2026-10-08)

현장 넷째~여섯째 날(10-06 ~ 10-08) 작업입니다. 릴리스 노트

[docs/38_release_notes_v2.1_build261008.md](#) .  작업 순서 계획과  보정 이력은 아래 두 절에 따로 있습니다.

수확 모션 — 현장 보정

- **역순 후퇴:** 수확 뒤 바구니로 갈 때 들어온 길을 그대로 되돌아 나옵니다(옆 → 대기점 상공 → 진입 시작점). 다른 배지에 걸리던 문제를 고쳤습니다.
- **손목(J6) 유지 · 이동 중 회전:** 바구니·관찰 자세로 갈 때 11자 방향을 되돌리지 않고 손목을 그대로 두며(빠름), 관찰 → 배지 이동에서는 작업 자세 변경을 이동과 동시에 합니다.   그리퍼 동작 설정 ▶ 로봇 측 ▶ 손목 J6 유지 / 이동 중 회전 체크(기본 켜).
- **면별 작업 방향 자동 회전:** 후보가 배지의 좌/우/앞/뒤 어느 면에 붙었는지에 따라 11자 방위를 자동으로 돌려 그 면으로 들어갑니다( 계획이 없어도, 러너도 같음). 두 배지 사이에서 1번은 왼쪽, 2번은 오른쪽으로 자동.

- **도달 한계** 700 → 750 mm(패드 거리 기준, 🤖 로봇 측 수치 **reach_max_mm**). 후보 표 '도달' 셀에 불가 사유(한계 / IK-TCP / 급변)가 붙고, 11자 방향 변경이 J6 급변으로 거부되지 않습니다.
- **옆 배지에 걸릴 때**(배지 간격을 좁힐 때): 옆 대기 거리 = max(진입 높이 approach_offset_mm, 갓 ø/2 + 측면 진입 옆 여유 entry_side_clear_mm). 먼저 **측면 진입 옆 여유**를 줄이고(60 → 30), 부족하면 **진입 높이**까지(60 → 45). 단계 로그 "표고 옆 n mm 대기점"으로 확인합니다.

🔍 Hand-Eye 지선단(손가락 위글) 보정 — 카메라를 옮긴 뒤

- 카메라를 옮기면 기존 보정 예측이 크게 벗어나도 새 검출을 버리지 않고 경고만 붙입니다("기존 보정 예측과 n mm 차이 — 카메라를 옮겼다면 정상").
- 위글 뒤 그리퍼 관절을 실측해 손가락이 실제로 움직였는지 확인합니다. 안 움직였으면 "손가락 위글이 실행되지 않음 — 관절 변화 n°" 로 알려 주니 🤖 그리퍼 화면에서 제어 모드·토크를 확인하세요.
- 손가락 세 끝점 고르기를 위글 변위 기준으로 바꿔 RMS 164 mm(끝점을 잘못 고름) → 7 mm. 등록 메시지에 매칭 비용·변위·후보 수가 붙습니다. 자동 다자세 캡처에 경과 시간이 표시됩니다.
- RMS 가 나쁘면 재캡처 전에 **저장 샘플 재검증**으로 같은 샘플을 새 알고리즘으로 다시 계산해 보세요.
- **골프공(파지 구) 방식이 가장 정밀합니다**(현장 0.4~0.5 mm, 지선단 3점은 5~7 mm). 손가락 각도가 비대칭이어도 공만 밀리지 않으면 됩니다. 파라미터 **자동 캡처 이동 속도(%)**(기본 40)와 **자동 캡처 빠른 모드**(1 = 12자세, 약 1분)로 캡처 시간을 줄일 수 있고, Σ 계산은 수 초입니다.

🎓 지지면 평면 — 바닥 오인

- 카메라 시야에 바닥이 많이 들어오면 지지면이 바닥으로 잡혀 배지가 검출되지 않던 문제: 인식 ROI 안의 점만으로 평면을 잡고, 갓이 평면보다 400 mm 이상 위면 바닥으로 보고 다음 평면을 찾습니다. 단계 로그 "지지면: z≈... · 인식 ROI 안 · 제외 평면 n개".
- 지지면·배지 검출은 점군 기하만 쓰므로 탁자(바탕색) 교체에는 영향이 없습니다. 탁자 높이가 바뀌면 인식 ROI 상자가 새 선반 면을 포함하는지 확인하세요.

🕒 수확 사이클 타임(CT) 줄이기 — 어디를 조절하나 (2026-10-13 실측)

관찰 자세에서 표고를 수확해 바구니에 담고 돌아오기까지 걸리는 시간은 **로봇 이동**이 절반을 넘고, 그 다음이 **비틀어 따기와 손가락 달힘의 접촉** 판정입니다. 아래는 2026-10-13 13:37:07~13:38:23 현장 세션(전체 수확, 이동 속도 40 %)의 단계 로그를 그대로 분해한 값입니다. 비틀어 따기만 현재 설정(2회)으로 환산했습니다(측정 당시 1회). 단계 로그는 1 초 해상도여서 각 구간에 ±1 초가 있습니다. 더 자세한 분해·재측정 절차는 [docs/40_harvest_cycle_time_guide.md](docs/40_harvest_cycle_time_guide.md) 와 결과보고 **부록 A** 에 있습니다.

관찰 자세 → 표고 2개 수확 = 약 85 초 (표고당 42.5 초)

구간	시간	비고
초기 3D 촬영 + 인식	4.7 s	첫 스캔은 선촬영이 없습니다
작업 계획 정렬 · 로봇 재생 준비	6.0 s	실행당 1회(재생 준비 3.5 s 포함)
표고① 수확 (배지1 · 우측면)	34.6 s	측정 32 s + 비틀기 2회 환산 2.6 s
재탐지 (관찰 복귀 + 인식)	6.0 s	촬영은 바구니 도착 직후 미리 찍어 복귀와 겹칩니다
표고② 수확 (배지2 · 윗면)	34.0 s	측정 32 s + 비틀기 2회 환산 2.0 s

표고 N개를 이어서 수확하면 앞의 두 구간은 실행당 1회이므로 $10.7 + 34.3 \times N + 6 \times (N-1)$ 초가 됩니다(10개 = 약 407초, 표고당 41초).

표고 1개 단계별 (초, 비틀기 2회 기준)

단계	배지1 · 우	배지2 · 위	무엇이 시간을 쓰나	조절 파라미터 · 위치
후보 계산 · 파지 중심 보상	1	1	계산만	—
접근 이동(상승 · 작업 방향 전환 · 수평)	8	4	이동 속도, 손목 방향 전 환 $\pm 120^\circ$, 이동 거리	이동 속도 · 측면 진입 상공 높이 — 🎓 👉 로봇 측
측면 대기점 하강	2	2	이동 속도, 상공 높이	측면 진입 상공 높이(120 → 90 mm)
수평 진입(스텝마 다 정지 · F/T 확 인)	5	5	진입 거리를 10 mm 씩 나뉘 스텝마다 정지	진입 스텝 10 → 20 mm(설정 파일 harvest.step_mm) · 진입 높이 60 → 45 mm
파지(손가락 닫 힘)	3	6	각도 이동이 아니라 접 촉 판정 — 아래 설명	11자 파지 감속 속도 2 → 4 %/s · 감속 구 간 폭 25 → 20 mm — 🎓 👉 로봇 측
파지 유지 확인 + 예압 재적용	1	1	유지 0.5 s + 예압 정착 0.3 s	예압 재적용 정착 시간 0.3 → 0.1 s. 유지 시간은 0.5가 하한
비틀어 따기 2회	5.6	5.0	좌 → 우 → 좌 → 우 → 중립 5점, 50%/s	비틀기 각속도 50 → 70 %/s · 반복 2회 — ⑤ 그립퍼 파지 프리셋
인출 + 예압 + 유 지 확인	1	1	30 mm 당김	인출 거리 30 → 20 mm
후퇴 + 진입 경로 역순 복귀	6	5	이동 속도, 상공 높이	이동 속도 · 측면 진입 상공 높이
바구니 이동 · 완 전 개방 · 낙하 확 인	3	4	이동 + 낙하 확인 0.8 s	이동 속도
합계	34.6	34.0	이동 19 s(55 %) · 비틀 기 5.3 s · 파지 4.5 s · 대 기 2 s	

파지 닫힘이 3~6 초인 이유

손가락은 목표각을 한 번 받고 움직이며, 앵은 실측 각도와 힘만 지켜봅니다. 시간을 쓰는 것은 각도 이동이 아니라 "닿았다"를 확인하는 과정입니다.

- 명령을 보낸 뒤 장비가 실제로 출발할 때까지 약 1.5 초가 걸립니다. 감속 구간에 들어갈 때 속도를 바꿔 다시 명령하므로 이 지연이 한 번 더 생깁니다.
- 대에 닿아 손가락이 멈춘 뒤, **막힘 판정 창**(그 속도로 2° 움직일 시간 = $2 \div \text{감속 속도} = 2^\circ/\text{s}$ 에서 1.0 초) 동안 움직이지 않으면 접촉으로 확정합니다.
- 부드러운 대에서는 F/T 가 0.0~0.1 N 밖에 안 읽히므로 힘으로는 멈추지 않고 **항상 막힘 경로**를 탑니다. 그래서 감속 속도는 달는 속도이면서 동시에 판정 창의 분모입니다. 2 → 4 °/s 로 올리면 창이 1.0 → 0.5 초가 됩니다.

순서대로 적용하기 — 누적 약 -12 초 / 표고

1. **이동 속도 40 → 55 %**(-5.0 s). 🎓 🍌 ▶ 로봇 측 ▶ 이동 속도. 이동이 CT의 55 % 입니다. 충돌 감지·힘초과 한계는 그대로 두세요.
2. **11자 파지 감속 속도 4 °/s, 감속 구간 폭 20 mm**(-1.8 s). 단계 로그 "달힘 6°/s → 틈 20 mm 안 4°/s" 로 적용값을 확인합니다. 부서짐은 '부서짐 한계'(max_n)가 그대로 보호합니다.
3. **진입 스텝 10 → 20 mm**(-1.5 s). 화면에 없으므로 `config/smartfarm.yaml` 의 `harvest.step_mm` 을 고칩니다. 갓 접촉 감지 해상도는 거칠어집니다.
4. **비틀기 각속도 50 → 70 °/s**(-1.5 s). ⑤ 그리퍼 파지 프리셋 ▶ 비틀어 따기. 대가 찢어지면 되돌립니다. **반복 2회는 그대로 두세요** — 1회로 줄이면 2.3 초를 벌지만 안 떨어질 위험이 커지고, 재시도 1회는 34 초입니다.
5. **예압 정착 0.1 s · 진입 높이 45 mm · 상공 높이 90 mm · 인출 거리 20 mm**(-1.9 s).
6. 🍌 **스캔당 연속 1 → 2**(표고 2개에서 -6 s). 한 스캔의 후보를 재스캔 없이 이어서 수확합니다. 옆 표고가 흔들려 좌표가 어긋날 수 있으니 서로 30 mm 이상 떨어진 후보가 많을 때 올리세요.

전부 적용하면 표고 1개가 34.3 → 약 22 초, 관찰 자세에서 표고 2개가 85 → 약 55 초가 됩니다. 한 번에 하나씩 바꾸고 🎓 실수확 검증으로 확인하세요.

줄이면 안 되는 것

- **예압 재적용** — 끄면 1.2 초를 벌지만 비틀기 중에 파지력이 빠집니다(2026-10-01 "잡았을 때 힘이 빠진다"의 원인이었습니다).
- **손목(J6) 사전 정렬** — 0 으로 끄면 배지1↔배지2 전환에서 손목이 ±360°를 넘어 이동이 거부되고 그 뒤 명령까지 막힙니다.
- **힘초과 한계 · 부서짐 한계 · 충돌 감지** — 표고 손상과 장비 보호용이며 CT 와 무관합니다.
- **파지 유지 확인 단계** — 0.5 초를 아끼는 대신 빈 파지를 성공으로 기록합니다.

🍌 화면의 🏠 **관찰 자세** / 🛒 **바구니 자세** 버튼은 🎓 에 저장한 자세로 로봇을 옮깁니다(러너 정지 상태에서만). 수확 전 시야 확인, 바구니 교체 때 쓰세요.

NEW v2.1 — 📄 Hand-Eye 보정 이력·적용 (2026-10-08)

- 🔗 Harvest Lab ▶ Hand-Eye 그룹 ▶ 📄 보정 이력·적용: 보정을 저장할 때마다 남는 이력(일시, 종류, 로봇, 카메라, 방법, 샘플 수, RMS, 카메라 위치, 보정 시각, 메모, 샘플 폴더)을 봅니다. 현재 적용 중인 보정은 굵게 ✓, RMS 25 mm 초과는 빨강입니다.
- 이 보정 적용: 고른 이력의 변환·RMS·방법을 `config/hand_eye.yaml` 에 다시 씁니다(현재 값도 이력에 남아 되돌릴 수 있음). 샘플 폴더 재검증: 그때의 캡처로 다시 계산합니다.
- 이력 파일: `config/hand_eye_history.jsonl`. 처음 열면 기존 보정이 '기존' 항목으로 들어갑니다.

NEW v2.1 — 🕒 작업 순서 계획: 배지 × 면 (2026-10-06)

표고는 배지의 여러 면에 붙고 11자 측면 진입은 들어오는 방향이 정해져 있으므로, 붙은 면 쪽에서 들어가야 합니다. 배지가 3~5개 놓였을 때 "몇 번 배지의 어느 면"을 어떤 순서로 작업할지 정해 두면 🎓 순서대로 검증과 🚫 독자 모드 러너가 그대로 수행합니다. 상세 `docs/37_harvest_work_plan.md`.

- 배지·면 자동 분류: 촬영+검출 때 지지면 위 덩어리를 배지로 찾아(로봇이 볼 때 왼쪽부터 1번) 후보마다 붙은 면(좌/우/앞/뒤/위)을 정합니다. 대가 뺀 방향(측정 갓 축)이 우선이고, 축이 없으면 배지 발자국 기준 위치로 정합니다. 후보 표의 배지·면 열과 단계 로그에서 확인합니다. 배지 번호는 배지가 늘어선 방향을 따라 매기며(로봇에서 멀어지는 방향으로 늘어서 있으면 가까운 배지가 1번), 좌/우 면은 그 방향의 양쪽, 앞/뒤 면은 그 수직입니다. 베이스에서 1.5 m 이내(그리고 인식 ROI 안)의 덩어리만 배지로 봅니다. 좌/우가 현장과 반대면 `harvest.rack_right_flip: true`.
- 🕒 작업 순서 패널(🎓 왼쪽 아래): 계획을 이름 붙여 만들고 단위(배지 번호, 면, 관찰 자세·작업 방향·바구니 프리셋, 메모)를 순서대로 둡니다. 관찰·바구니 프리셋을 비우면 현재 활성값을 쓰고, 작업 방향을 비우면 그 면으로 들어가도록 활성 작업 방향을 자동 회전합니다(좌 면은 오른쪽으로, 우 면은 왼쪽으로 진입). 윗면 표고는 그 배지를 처음 들어가는 단위가 수확합니다. 콤보는 보기/편집용이고 ☒ 적용 버튼을 눌러야 활성이 됩니다(상태줄 ☒ 적용 중: "...", 콤보의 ✓ 표시, 🚫 해제로 끄). 💾 저장은 적용을 바꾸지 않습니다. 패널은 🎓 오른쪽 열(실수확 검증 아래)에 있습니다.
- ▶ 순서대로 검증: 단위마다 관찰 자세 이동 → 촬영·검출 → 그 배지·면 후보만 순차 실수확 검증(검증 단계·드라이런·재탐지 설정 적용).
- 🚫 러너: 독자 모드는 활성 계획의 단위 순서대로 수확하고, 남은 후보가 없으면 다음 단위로 넘어갑니다. 관제 모드는 대상의 면에 맞는 단위의 작업 방향·바구니를 씁니다. 계획이 없으면 종전과 같습니다.
- 준비 순서: ① 촬영+검출로 배지 번호·면 확인 → ② 왼쪽 작업용·오른쪽 작업용 자세 프리셋 저장 → ③ 계획 작성 → ④ 드라이런으로 순서대로 검증 → ⑤ 러너.

NEW v2.1 Build261002 — 🚫 수확 모니터링 대시보드 · 수확 장부 · 정합성 검사 · ⑤ 비틀기 현장 보정 (2026-10-02)

현장 셋째 날 작업입니다. 릴리스 노트 `docs/36_release_notes_v2.1_build261002.md`.

⑤ 비틀어 따기 — 굵김 제거와 회전 중심 보정

- 좌 ↔ 우 사이에서 멈추지 않음: 중립을 거치지 않고 좌 + θ 에서 우 - θ 로 바로 이동하고, 마지막 반 복 뒤에만 중립으로 돌아옵니다(반복당 명령 4 → 2). ⑤ 비틀어 따기 ▶ **중립 경우** 0(기본) / 1(중전 좌 → 중립 → 우 → 중립). 직선 보간이라 중간에서 회전 중심 쪽으로 약간(단계 로그에 mm) 치우칩니다.
- **회전 중심 보정(세계 기준)**: 좌/우 젓힘에서 배지 부착점이 움직이면 두 값으로 맞춥니다. **수평 보정** (젓힘 축에 수직) — 좌(+ θ)에 내려가고 우(- θ)에 올라가면 음수, 반대면 양수, 크기 $\approx$ 오르내린 양 $\div \sin \theta$ (15° 면 $\times 3.9$). **상/하 보정** (+ 위) — 양쪽 다 같은 방향으로 움직이면, 크기 $\approx \div (1 - \cos \theta)$ (15° 면 $\times 29$). 좌/우로 옆으로 밀리면 **깊이** (twist_pivot_mm = 패드 접촉점에서 배지 표면까지 거리)를 맞춥니다. 단계 로그에 보정 방향 벡터가 표시됩니다.
- **연필 검증**: 11자로 연필을 잡고 끝을 종이에 댄 채 비틀기를 실행해 끝이 한 점에 머물면 완성입니다. 옆으로 선이 그어지면 깊이, 끝이 뜨면 수평 보정을 다듬습니다.
- 🎓 실수확 검증 🍄 러너도 같은 순서·보정을 씁니다.

🔬 Lab · 안정성

- 촬영+전체검출 뒤 라이브 화면으로 바뀌며 후보 라벨이 다른 곳에 찍히던 문제를 고쳤습니다(정렬 텍스트 → 원본 2D 호모그래피 변환).
- 앱 종료 중 위치독이 닫힌 이벤트 DB 에 쓰며 나던 오류를 막았습니다.

🔧 수확 모니터링 시스템

SmartFarm 관제와 주고받은 **수확 대상 수신** → **수확 수행** → **결과 전송** 전 과정을 로봇 PC 의 수확 장부 (`data/harvest_ledger.db`)에 기록하고, 웹 대시보드에서 이력·전송·정합성·통계를 봅니다. 설계 `docs/34`, 구현·운용 `docs/35_harvest_monitoring_system.md`.

- **열기**: 🍄 Shiitake Harvest 화면 ▶ 상태 그룹 ▶ 📊 **모니터링 대시보드 열기**(브라우저 `http://<로봇 PC>:8770/ledger`). 다른 PC 에서는 토큰 (`config/smartfarm.yaml` ▶ `harvest.monitor_token`, 최초 기동 때 자동 생성)을 입력합니다. 기존 관제 임베드 페이지(/)는 그대로입니다.
- **탭**: 요약(오늘 수확·평균 사이클·전송 대기·실패/폐기·미전송·정합성 🟢 🟡 🔴 ·대상 수) · 실시간 · 대상(관제에서 받은 대상의 수신→점유→수확/반납) · 수확 이력(기간·모드·결과 필터, 행 클릭 상세, CSV) · 전송(아웃박스 상태·오류, **재전송** 버튼) · 정합성(**지금 검사**, 서버 큐 재동기화, 확인) · 통계(일 별 성공/실패, 사유·배지·품종·갯 크기·손상 등급, 사이클/파지력 분포, CSV).
- **정합성 검사**(기본 5 분마다 + 세션 종료 뒤): 미전송 초과, 서버에 없는 수확(🔴), 서버에만 있는 수확(중복 의심), 결과·파지력 불일치, 세션 수확 수 불일치, 수확 완료 대상 재등장, 리스 만료, 아웃박스 폐기(🔴). 서버 대조에는 SmartFarm **viewer** 계정이 필요합니다 — `smartfarm.yaml` 최상위에 `reconcile_user` / `reconcile_password` (또는 `reconcile_token`)을 적습니다. 없으면 로컬 규칙만 검사합니다.
- 🎓 **실수확 검증·드라이런**도 장부에 기록되어 통계에서 **모드**로 구분됩니다(관제 전송 대상은 아님). 요약의 '오늘 수확' 은 드라이런을 제외합니다.
- **설정**(`harvest:`): `reconcile_enabled` (true) · `reconcile_interval_s` (300) · `reconcile_server_limit` (200) · `reconcile_unsent_warn_s` (600) · `reconcile_match_window_s` (30) · `monitor_token`.

NEW v2.1 Build261001 — 🎓 실수확 검증에 11자 사이클 적용 · 측면 진입 · 재탐지 · 핸드아이 재보정 (2026-10-01)

현장 둘째 날 작업입니다. 🎓 Harvest Teaching 실수확 검증이 ⑤ 11자 대칭 사이클을 그대로 사용하고, 로봇이 표고 옆에 서서 수평으로 진입해 잡습니다. 상세는 [docs/33_release_notes_v2.1_build261001.md](#), 그리퍼 기능·설정 전체는 [docs/32_gripper_functions_and_settings.md](#), 시간순 기록은 [docs/23](#).

- **사이클 지정:** 🎓 🍌 그리퍼 동작 설정 ▶ 사이클 콤보에서 "11자 대칭 파지(패드 평행)" 을 고르면 준비·파지·비틀기·유지 확인·해제·완전 개방을 ⑤ 저장값대로 수행합니다(🔴 러너 공용). 파지 프리셋 콤보에는 **파지 종류 프리셋만** 보입니다 — 자세 프리셋이 규칙에 남아 있으면 무시하고 로그에 ⚠ 가 뜹니다.
- **측면 진입 경로:** 배지 상공(기본 120 mm)으로 먼저 올라가 표고 옆 대기점 위로 수평 이동 → 하강 → 손끝 전진 방향으로 수평 진입 → 파지 → 비틀기 → 인출 → 옆으로 빠져 상공. 대기점 거리 = max(진입 거리 60, 갓 $\varnothing/2$ + **측면 진입 옆 여유** 30). 갓이 하강 때 걸리면 옆 여유를 50~60 으로.
- **로봇 끝은 표고 옆 ~130 mm**에 섭니다(11자 패드 중점 보상). 들어오는 쪽은 🎓 작업 방향이 정하며, 손가락이 로봇 베이스 반대쪽을 향하게 저장해야 도달 범위가 넓어집니다. 도달 검사의 700 mm 한계는 표고 거리 기준이고 로봇 끝 목표는 IK 로 판정합니다.
- **유지·인출 확인:** 11자 예압 파지는 F/T 가 0.0~0.3 N 이라 **패드 톡**으로 판정합니다(단힘 기준 + 8 mm). **예압 재적용**(비틀기 전·인출 전, 🍌 설정 체크)이 대 눌림으로 사라진 파지력을 복구합니다. 더 세게 잡으려면 예압이 아니라 🍌 그리퍼 티칭 패널의 토크 리미를 다루어야 합니다(docs/32 §4.1).
- **바구니:** 해제 뒤 완전 개방 자세까지 벌려 손가락 위에 갓이 남지 않게 합니다.
- **수확 뒤 재탐지**(체크, 기본 커짐): 순차 검증에서 두 번째 후보부터 관찰 자세 → 재촬영·재검출 → 이전 좌표와 가장 가까운 후보로 진행. 장면이 안 바뀌었으면(드라이런·도달 불가) 생략, 못 찾아도 원래 좌표로 계속합니다.
- **티칭 자세 프리셋:** 관찰·바구니·작업 방향을 이름·설명과 함께 여러 개 저장하고 콤보로 활성을 고릅니다. 자세마다 🍌 **그리퍼 자세 프리셋**(완전 개방·이동 대기·11자 대칭 준비 등)을 지정하면 ➡ 이동 때 함께 실행됩니다. 작업 방향이 이미 3° 이내면 "회전 생략" 으로 표시됩니다.
- **후보 표:** '자세' 열(갓 축 기울기·출처 중력/PCA)과 6D 툴팁, 높이·대 셀 툴팁에 산출 출처. 검출 제외 사유가 "갓 지름 n mm 가 범위 밖 · ROI 크롭 가장자리 걸림" 처럼 구체적으로 나오고, 2D 화면에서 **ROI 사각형 안이지만 3D 상자 밖이라 잘린 영역이 자홍**으로 보입니다.
- **핸드아이·앵커:** 지선단 3점 보정의 아웃라이어 제외를 중앙값 기준으로 바꿔 배경 오검출 2건이 전체를 망치지 않게 했고(현장 캡처 rms 12.4 mm, 책상 기울기 4.2°→1.0°), 지지면 기울기가 2.5° 를 넘으면 핸드아이 회전 오차 안내가 뜹니다. **HEC-Board-35** 는 비스듬히 보이는 타원 원도 검출(수확 설정 anchor_type hec35 · 35 mm 권장), ArUco 는 다중 스케일 + 거짓 검출 차단. 핸드아이를 다시 잡으면 🌀 손끝 실측도 다시 하세요.
- **로봇 자동 재연결:** 전원 OFF→ON 뒤 COMM_LOST 가 반복되던 문제 — 10 s 뒤 30 s 간격 재연결, Fault 리셋 때 통신이 정체돼 있으면 재연결합니다.
- **주의:** 옛 기준으로 켜 파지점 위치 보정(dz -69 등)과 파지 높이 보정은 0 으로 되돌린 뒤 "📏 현재 TCP = 파지점" 으로 다시 재세요(단계 로그에 ⚠ 안내).

v2.1 Build260930 — 11자 대칭 파지 · 비틀어 따기 현장 확정 (2026-09-30)

하루 현장 작업(RB5-850 + DG-3F-M + EpicEye Laser)으로 11자 대칭 수평 파지 → 비틀어 따기 → 표고 분리까지 확인한 변경입니다. 상세는 [docs/31_release_notes_v2.1_build260930.md](docs/31_release_notes_v2.1_build260930.md), 시간순 기록은

<docs/23>.

- 👊 ⑤ 11자 대칭 수평 파지 3종(준비·파지·사이클): F1·F2 를 같은 각으로, 센서 패드가 대를 정면으로 보게 잡습니다. 준비 J2 는 실측 목표 83° + 정상상태 보정 7°(사용자 모드에서 J2 가 명령보다 6~7° 덜 가는 장비 특성), 파지는 J3 만 닫힘(J4 비움 0). 파지력은 목표 접촉력이 아니라 '밀착 예압'(권장 5~8°, 상한 12°)이 정합니다 — 패드 정면 접촉에서 F/T 는 0.1~0.2 N 만 읽힙니다. ⑤ 단일 실행은 3 s 뒤 예압을 풀므로 파지→비틀기를 따로 시험할 때는 '예압 유지' 를 0 으로.
- 비틀어 따기 재설계: 회전 중심은 좌/우 손끝 교대 지렛목(좌 +θ 는 왼쪽 손끝, 우 -θ 는 오른쪽 손끝을 중심으로) — 손끝이 버섯에서 미끄러지지 않습니다. 축은 0 수직 비틀기(표고 분리, 기본) → 2단계 1 좌/우 젖힘(갓이 왼쪽/오른쪽 손가락 쪽으로 기울)을 이어서 수행합니다. 속도는 각속도(°/s)로 정하며 각속도가 0 이 아니면 로봇 속도 % 는 쓰이지 않습니다. 방향이 비스듬하면 손끝 선 미세 보정(°)으로 맞춥니다(현장 -10°).
- 손끝 위치는 실측으로: 📷 Harvest Lab ▶ 핸드아이 그룹 🌀 손끝 실측 → 비틀기 티칭 저장 — 3D 카메라(eye-to-hand 핸드아이)로 두 손끝을 직접 측정합니다. 벌림이 다른 두 자세(11자 준비 / 표고 없이 11자 파지로 닫은 상태)에서 각 1회 실행하면 파지 자세의 손끝을 실측값으로 보간합니다. 실패해도 원본이 [data/harvest_lab/tipmeasure/](data/harvest_lab/tipmeasure/) 에 저장되고 단계별 진단 수치가 표시됩니다. 핀을 쓰는 4점 티칭도 ⑤ 에 있습니다.
- Vision/Tool: 컨트롤러 TCP 정의 추정 버튼**(관절각 FK 플랜지 vs 보고 TCP — 현장은 플랜지 + 203 mm 툴 좌표) · 그리고 **장착 오프셋(mm)** 입력 · 마운트 회전 0.1° 단위. 비틀기 손끝 계산은 플랜지 기준이라 펜던트 TCP 설정과 무관합니다.
- 그리고 **보호·연결**: 이동 중 스톱 보호(3 s), 2단 경압 완화는 전류 60 mA 이상일 때만(J2 정상상태 오차 오탐 제거), 방사 파지 실측 틈 3 mm 하한, **자동 재연결**(시작 실패·끊김 뒤 30 s 간격), Startup 설정 "그리고 연결 후 시작 자세로 자동 이동" 끄기.
- 파지 판정**: 닫힘 끝 FK 틈 기준으로 '목표 끝 정지' 오인 방지, 패드 사이 물체 틈 판정, 정지 시 패드 틈 로그(대 굽기와 비슷하면 접촉). ⑤ ▶ 는 품 값, ③ 포즈 목록 더블클릭·사이클·수확 모션은 📁 저장 값으로 실행됨을 로그에 안내.

v2.1 — 다중 로봇 HAL · 알람 · 역할 (2026-09-28~29, Build260929)

fineAi Multi-Robot Controller 기능명세(fineAi_RB+/docs/01)를 **RoboVision** 안에 Rainbow Robotics 기준으로 구현했습니다. 로봇 기종 확대(두산·HD현대·Techman)를 염두에 두고 벤더별 차이를 드라이버 계층(HAL v2)에서 흡수하는 구조이며, 기존 화면과 기능은 그대로입니다. 상세 설계·명세 대비 현황은

<docs/30>.

- **세션 상태 8단계:** 미연결 → 연결 중 → 모니터링 → 제어권 보유 → 대기·서보 ON → 실행 중 / 일시정지, 그리고 Fault(비상정지·충돌·통신 끊김·제어권 상실). Robot Control ▶ Overview 의 **Session** pill 과 단계 스테퍼(연결 ▶ 제어권 ▶ 서보 ON ▶ 실행), Fleet 타일, Robot Monitoring 배너, 상단 바 로봇 칩에 같은 상태가 보입니다. Fault 로봇이 있으면 적색 **Fault 배너**가 나타나고 로봇별 **리셋**으로 모니터링 상태로 돌아갑니다(하드웨어 E-STOP 이 아직 눌러 있으면 리셋 불가).
- **RUN ▶ Alarms & Events**(신규): 벤더 알람 코드를 공통 레벨(정보/경고/오류/치명)로 매핑해 표시(원본 코드 병기, `config/alarm_map/<vendor>.yaml`, 매핑에 없으면 오류), 로봇/레벨/미확인 필터, 선택·전체 확인(Ack), 명령 감사 로그 탭. 상단 바 🚨 **미확인 알람 배지**를 클릭하면 이 화면으로 옵니다. 기록: `data/fineai_rc.db`.
- **SYSTEM ▶ Robots & Drivers**(신규, 관리자): 등록 로봇 표(벤더·모델·IP·포트·DOF·드라이버 구현 여부·세션 상태), 로봇 추가/편집/삭제 폼 — **벤더를 고르면 모델 목록·기본 포트·DOF·관절 한계가 자동 입력**됩니다(TCP·페이로드·설치 기준·벤더 고유값 vendor_params 포함). **연결 테스트 5단계**(ping → 제어 포트 → 상태 포트 → 컨트롤러 펌웨어 버전 → 드라이버·Capability)와 **드라이버 Capability 매트릭스**(벤더 예상 ● △ × vs 선택 로봇 드라이버 실제). 현재 실장비 드라이버는 Rainbow(RB)뿐이며 두산·현대·Techman 은 등록·표시·연결 테스트까지 됩니다.
- **역할 3단계:** 👤 Operator(RUN 화면만, 속도 바 상한 50%) · ⚙️ Engineer(관리자 전용 제외 전부) · ☑ Admin(전부). 상단 바 역할 버튼 ▾ 메뉴에서 전환하며 Engineer/Admin 은 PIN(`app.yaml` `ui.engineer_pin` / `ui.admin_pin`)이 필요합니다.
- **F12 = 전 로봇 비상정지:** 어느 화면에서든 F12 또는 상단 바 ⏻ E-STOP 을 누르면 연결된 모든 로봇에 정지 명령을 보내고(제어권·명령 큐 우회) Fault 로 전이, 알람·감사 기록을 남깁니다. **앱 비상정지는 안전 등급 정지가 아니므로** 하드웨어 E-STOP·안전 PLC 가 필수이고 앱 정지는 보조 수단입니다.
- **통신 위치독:** 실장비 로봇의 상태 채널이 3 s 이상 끊기면 정지 명령 후 Fault(통신 끊김 알람). `app.yaml` `safety` 절(운영자 속도 상한·Jog 250 mm/s·이동 전 확인 300 mm/30°·위치독 3 s)로 조정.
- **화면 재설계 — 프로토타입(fineAi_RB+/docs/prototype) 레이아웃·폰트:** 로봇 관련 화면(Fleet Dashboard · Robot Monitoring · Alarms & Events · Robot Control · Robots & Drivers)을 웹 프로토타입과 같은 카드 레이아웃으로 바꿨습니다(기능 축소 없음). 사이드바는 짙은 남색(#131a25), 카드·버튼·표·탭은 공통 디자인 토큰. **Fleet:** 벤더 필터 · KPI(연결/가동 중/미확인 알람/사이클/가동률/이동 거리/저장 궤적) · 로봇 타일(벤더 배지·세션 pill·J1 추이·속도/사이클/TCP Z/온도·리셋/연결) · 최근 알람 · 벤더별 현황 · 3D · 시스템. **Monitoring:** 한 화면 스크롤(3D | 상태 카드: TCP 자세·힘·프로그램·속도·모드·상태 유지 ▶ 시작/||/재개/■/리셋 → Joint 차트 → Joint Status(한계 대비 바 80/95%) | I/O 램프 → 가동 통계 → 사양·플래그·자세 표·그리퍼). **Robot Control:** 세션 스테퍼 + 상태별 버튼 · 전원/모드/속도 + 현재 TCP · **Jog**(관절/직교 · Base/Tool/User · 연속/0.1/1/10, 누르고 있는 동안 이동) · **목표 이동** MoveJ/MoveL(300 mm/30° 초과 시 확인 대화상자·관절 한계 검사) + 저장 자세 · TCP/페이로드/충돌 적용·저장 · 프리드라이브 전제 조건 체크리스트 · **DO 출력 램프**(클릭 토글)·**컨트롤러 프로그램 실행/정지** · 정보 · 설치 기준.
- **폰트:** 웹 프로토타입과 같은 **IBM Plex Sans KR**(본문) · **Barlow Semi Condensed**(제목·숫자 헤더) · **JetBrains Mono**(수치)를 프로그램에 내장(assets/fonts, SIL OFL)해 PC 에 설치되어 있지 않아도 동일하게 표시됩니다.
- **HAL v2(개발자):** 드라이버 공통 표면 `capabilities()` · `sample()` · `alarms()` · `stop/pause/resume` · `jog_step` · `set_tcp/set_payload/set_collision` · `read_io/write_do` · `run_program`. Rainbow DO 는 `set_box_dout` / `set_tool_out`, 컨트롤러 프로그램은 `task load +`

`task play` (현장 미검증 — 펌웨어가 거부하면 `robots.yaml` `vendor_params.do_cmd` 로 교체). 벤더 중립 프로그램 형식 `.frc.yaml` (data/programs) 모델 준비.

현장 개선 — 프리드라이브 자동 차단 · 지지면 하한 · 화면 폭 (2026-09-29, Build260929 이후)

- ♡ **프리드라이브 자동 차단**: 프리드라이브를 켜 뒤 **1.5 초** 동안은 팔을 잡지 말고 지켜보세요. 이 사이 팔이 스스로 15 mm 이상 움직이면(페이로드·무게중심 불일치로 처짐) 또는 이후 TCP 150 mm/s · 관절 30 °/s 를 넘는 급한 움직임이 보이면 프로그램이 프리드라이브를 **즉시 끄고 서보를 다시 활성화**한 뒤 치명 알람("프리드라이브 자동 차단 ...")을 띄웁니다. 한도는 `config/app.yaml` `safety.freedrive_*`. 컨트롤러의 중력 보상 속도 자체를 프로그램이 늦출 수는 없으므로, Tool / Vision 의 페이로드(DG-3F-M $\approx$ 1.3~1.5 kg)·무게중심을 먼저 맞추세요.
- 📌 **지지면 하한(배지 평면) — Harvest Lab ▶ 파라미터**: 무앵커(3D 모델) 검출은 표고마다 카메라 깊이로 높이를 재므로 마커(앵커)가 높이를 더 정확하게 해 주지는 않습니다. 대신 촬영마다 점군에서 **배지 평면을 마커 없이(RANSAC) 구해** 파지 TCP 목표가 **평면 + 여유(기본 15 mm)** 아래로 내려가지 않게 올립니다(바닥 찌름 방지). '지지면 하한' 체크 · '지지면 여유(mm)' · '지지면 기울기 한도(°)'(기본 30 — 넘으면 벽/배경으로 보고 그 촬영에서는 적용 안 함, 갓 중심 과반이 평면 아래여도 적용 안 함). 🎓 Harvest Teaching 검출 뒤 진행 줄과 🏠 러너 동작 기록에 "지지면: z≈... · 기울기 ..." 와 "지지면 하한 적용 — ... mm 올림" 이 남습니다. 기본은 꺼짐. 표고보다 아래로 내려가는 원인이 **hand-eye-TCP 오차면** 이 옵션이 아니라 📐 위치/높이 보정(`grasp_offset_mm`)과 hand-eye 재캘리브레이션이 맞습니다.
- 📦 **3D Pick&Place 왼쪽 설정 패널 글자 잘림(FHD)**: ⑥ 실행의 '최대 픽 · 실행 시간 · 1개 제한 · 재시도' 한 줄이 새 폰트에서 패널 최소 폭(605 px)을 넘어 잘리던 것을 2×2 격자로 바꾸고, 카메라 콤보의 긴 이름이 최소 폭을 키우지 않게 했습니다. 그래도 패널이 내용보다 좁으면 이제 가로 스크롤이 생깁니다(잘림 없음). 🎯 3D Recognition · 🌱 Template Modeling 도 같은 규칙.
- 📄 **그리퍼 진단 로그 용어**: '관절 나이 / F/T 나이'(data age 직역) → '최신 샘플 경과', '통신주기값' → 'SDK 통신횟수/s'.
- 👉 **Gripper ▶ ⑤ 파지 프리셋 — 3지 방사(120°) 표고 파지 3종 추가**: 3지 방사 준비(갓 ⌀ + 여유)는 세 손가락 밑동을 120° 등간격($J1/J5/J9 = 0/-60/+60^\circ$)으로 벌리고 J2·J3·J4 는 세 손가락이 같은 각도가 되게 하며, 세 손끝이 그리는 **원의 지름 = 갓 지름 + 여유(기본 30 mm)** 가 되도록 J3(J4 = J3 × 비율)를 FK 로 정합니다. ⑤ 탭에서 파라미터를 바꾸면 계산된 12축 자세와 손끝 원 지름이 바로 보이고 '↗ 자세로 이동만' 으로 확인할 수 있습니다. **3지 방사 동기 파지**는 그 자세에서 세 손가락 J3·J4 를 같은 각도로 동기 닫아 갓을 중앙으로 모으며 F/T 목표 접촉력(1.5 N, 한계 4 N)에서 멈춥니다. **사이클: 3지 방사 감싸기**를 🏠 Lab ▶ 운영 사이클로 고르면 🏠 실수확에서 **검출된 갓 지름**으로 준비 자세를 대상마다 다시 계산하고, 방사 대칭이라 11자 방위 정렬은 생략(진입 기울기만)합니다. 손끝 캡 ⌀12 mm 만큼 실제 틈은 여유보다 좁습니다. 실장비 미검증 — 처음엔 드라이런.⑤ 검증으로 확인하세요.
- 👆 **숫자 입력 ▲▼ 화살표 오동작(전 화면 공통)**: Windows 11 기본 스타일에서는 스피ن 버튼의 클릭 영역이 좌우로 나란히 놓이는데 테마 QSS 는 위/아래로 그려, '올리기를 누르면 내려가고 내리기는 무반응' 이었습니다. 프로그램 스타일을 Fusion 으로 고정하고 화살표 영역을 QSS 로 위/아래 반반 배치해 그림과 클릭 영역이 일치합니다(assets/icons/spin_up-down.png).
- 📌 **F/T 자동 영점 — 준비 자세 도달 직후(허공)**: 지선단 F/T 센서는 파지 뒤 0.5~2 N 이 남는 이력 현상이 있어(현장 09-29: 방사 파지 뒤 준비 자세에서 0.59/1.74/1.50 N 잔류, ① F/T영점 버튼으로 0 복귀

확인) 접촉 판정(0.3 N)·목표 접촉력(1.5 N)이 오염될 수 있습니다. 🧠·🎓 수확 모션은 원래부터 pre-grasp 도달 뒤 하강 직전에 대상마다 자동 영점(harvest.ft_rezero, 기본 컴)을 하고 있었고, 이제 동작 로그에 "F/T 영점(준비 자세 도달 · 허공) — 잔류 ... N 제거" 로 보이며 🧠 Lab ▶ 파라미터 'F/T 영점' 체크로 끄고 켤 수 있습니다. 🧠 Gripper ▶ ⑤ 실동작 검증에는 '준비 자세 뒤 F/T 영점'(기본 컴)이 새로 생겨 자세·벌림·3지 방사 준비 단계 도달 직후 영점을 잡습니다(단일 실행·사이클 모두). 기존 '시작 전 F/T 영점' 은 손가락이 물체를 잡은 채 시작하면 오히려 오프셋을 만들므로 기본 꺼짐으로 바꿨습니다.

- 🧠 ⑤ 11자 대칭 수평 파지 3종 추가 (09-30):** 기존 11자 수평 파지 (hv_ready2_pinch-hv_pinch_h_grasp)를 FK 로 검토한 결과 두 가지가 확인됐습니다. ① 현장 티칭 준비 자세는 좌우가 비대칭입니다(J2 -81.2/+76.4, J3 25/30, 손끝 높이차 7.5 mm). ② 손끝 F/T 센서 패드 면은 tip 모듈의 두께 방향(+z)이며 J3·J4 가 같은 축이라 패드 방향은 J3+J4 로 정해집니다 — 종전 파지의 'J4 비율 +0.5' 로 오므리면 16° 만에 패드가 뒤집혀(cos -0.96) 센서가 대를 정면으로 누르지 못합니다. 새 프리셋: **11자 대칭 준비(패드 마주봄)** — F1·F2 의 높이(J2 干)·굽힘(J3·J4)을 같은 각도로 두고, 패드 틈 = 대 굽기(기본 13) + 여유(15)가 되게 J3 를 풀며, 끝마디 J4 는 두 패드가 정확히 마주 보는 값을 FK 로 고릅니다(cos 0.99, 예: J3 61° / J4 -6°). **11자 대칭 파지(패드 평행)** — J3 +Δ · J4 -Δ(비율 -1)로 패드를 평행하게 유지하며 프로파일 이동으로 오므리고, 힘/막힘에서 정지·예압, 두 손가락 접촉이면 성공. **사이클: 11자 대칭 파지.** 🧠·🎓 실수확은 🧠 Lab '대 굽기(11자 대칭 준비 기준)' 로 자세를 재계산하고 11자 방위 정렬(손끝 선)도 이 자세로 합니다. 3지 방사 파지는 그대로 두었습니다. 기존 11자 프리셋도 남아 있으나 위 이유로 새 프리셋 사용을 권합니다.
- 그리퍼 하드웨어 보호 3종 (09-30):** 2개월 3회 수리 이력과 09-30 F3 모듈 에러(423)를 계기로 프로그램이 장비를 밀어붙이는 경로를 막았습니다. ① **이동 중에도 스톱 보호** — 프로파일 이동 중이라도 어느 관절이 명령보다 15° 이상 뒤쳐진 채 3초 멈춰 있으면 명령을 실측 + 1.5° 로 되돌립니다(출발 지연 1.5초 감안). 2단으로 잔차 4° 이상이 20초 넘게 이어져도(예압·막힌 자세 방치) 실측 + 1° 로 완화합니다. ② **예압 유지 시간 제한** — ⑤ 에서 파지 프리셋을 단독 실행하면 막힘 접촉 예압을 '예압 유지(초, 기본 3)' 뒤 실측 자세로 풀어 밀착 상태로 방치되지 않게 합니다(사이클·실수확은 해제 단계가 처리하므로 제외, 0 = 유지). ③ **손끝 틈 3 mm 하한** — 3지 방사 파지 중 실측 손끝 틈이 3 mm 아래로 내려가면 최소 틈 설정과 무관하게 즉시 정지·고정합니다. 어제 발생한 손끝끼리 밀착은 이 하한과 틈 보정 실측으로 막습니다.
- 3지 방사 준비 — 끝마디(J4) 고정각 (09-29 17:30 현장 사진):** 준비 자세에서 끝마디가 J3 의 절반만 굽어 손끝이 아래를 향하면, 오므릴 때 중간 마디가 갓에 먼저 닿아 대(stem)까지 가지 못합니다. ⑤ '3지 방사 준비' 에 '**J4 고정각(끝마디, °)**' 파라미터를 추가했습니다. 0 이면 종전 비율 방식, **70~90** 을 주면 끝마디(J4/J8/J12)를 그 각도로 굽힌 채 J3 만 풀어 같은 손끝 원 지름(갓 Ø + 여유)을 맞추므로, 손끝이 갓 아래로 파고들어 대를 향한 채 진입합니다(갓 60 기준 J4 80° 이면 J3 48°). ⑤ 탭에서 값을 바꾸면 계산 자세와 손끝 원이 바로 보이고 '↗ 자세로 이동만' 으로 확인할 수 있습니다. 이 경우 '3지 방사 동기 파지' 의 '**J4 비율**' 은 **0** 으로 두어 오므릴 때 끝마디가 더 말리지 않게 하는 것을 권합니다. 🧠·🎓 실수확도 같은 파라미터를 씁니다(검출 갓 지름으로 재계산).
- 3지 방사 파지 — 접촉 판정 보강 · 조절 파라미터 (09-29 14:15):** 지선단 F/T 는 패드 정면 접촉만 잘 읽고 모서리로 닿으면 힘이 안 잡히므로, **막힘**(명령은 3° 이상 앞서 있는데 실측이 0.4 초 이상 안 움직임)도 접촉으로 봅니다(로그 "막힘 접촉(F/T 미감지) — F3"). 성공 판정은 "세 손가락 모두 목표 접촉력" 에서 "**접촉(힘 ≥ 유지 최소력 또는 막힘) 손가락이 '최소 접촉 손가락 수'(기본 2) 이상**" 으로 바뀌어, 표고가 중심에서 벗어나 한 손가락이 못 닿아도 잡은 것으로 인정합니다("부분 접촉 파지"). 조절: ① **접촉 간격** — 🧠 Lab '최소 손끝 틈(mm)'(기본 5, 줄이면 더 오므림, 0 = 끄). ② **힘** — ⑤ '3지

방사 동기 파지' 파라미터의 목표 접촉력(1.5 N, 낮추면 더 일찍 정지·성공), 유지 최소력(0.3 N, 접촉 인정 문턱), 부서짐 한계(4 N), 최소 접촉 손가락 수(2). ③ **관절 자체의 힘(토크)** 은 사용자 모드에서는 장비 내부 제어가 정하며 프로그램에서 바꿀 수 없습니다(DG Manager 설정, 🍌 Gripper ▶ ① 제어 패널 '토크 리밋' 토글). 개발자 모드일 때만 `smartfarm.yaml gripper.modes.developer` 의 `gain_p/d/i` 가 적용됩니다.

- **3지 방사 파지 — DG Manager 방식 프로파일 이동 (09-29 13:50):** "DG Manager에서는 정확히 그 각도로 가는데 왜 여기서는 안 되나" 의 답입니다. DG Manager 는 목표각 하나를 모션시간과 함께 `MoveJointAll` 로 보내고 장비 내부 제어가 정확히 도달합니다. 이 프로그램의 닫힘 루프는 사용자 모드에서 0.25 초마다 새 목표를 280 ms 모션시간으로 다시 보내는 스트리밍 흉내였고, 장비는 매번 꺾적을 새로 시작하느라 수렴하지 못해 명령보다 약 10° 뒤쳐졌습니다(로그 "잔차 10°"). 이제 3지 방사 파지는 **목표각(J3 +30°)을 8°/s 속도의 모션시간으로 한 번 보내고** 실측 틈·힘만 감시하다가, 손가락별로 틈이 '최소 손끝 틈'(5 mm)에 닿거나 접촉력이 오르면 **그 손가락을 실측 자세로 정지 명령**합니다(틈 25 mm 안에서는 2°/s 로 감속). 다른 파지(소프트·11자)는 종전 방식이며, 개발자 모드에서는 스트리밍이 정확해 종전 루프를 씁니다.
- **3지 방사 파지 — 손끝 틈 = 모듈 표면 거리 · 손가락별 균등 정지 (09-29 13:25 스크린샷):** "손끝 끝점 간 27/36/40 mm" 는 모듈 중심 사이 거리라 10 mm 가 될 수 없고, 보시는 것은 **모듈 표면 사이 틈** 입니다. 이제 틈을 지선단 모듈 상자(28.5×22.4×19 mm) 표면 사이 최소 거리로 직접 계산합니다(스크린샷 자세에서 5.2/15.8/14.2 mm — 한 쌍만 가깝고 둘은 넓던 실물과 일치). 그리고 닫힘 방식을 바꿔 **손가락마다 이웃과의 틈이 '최소 손끝 틈'(기본 5 mm)에 닿으면 그 손가락만 실측 자세로 고정** 하고 나머지는 계속 닫습니다. 표고 없이 실행하면 세 틈이 모두 약 5 mm 로 균등하게 멈추고, 표고가 있으면 접촉력에서 먼저 멈춥니다. 🍌 Lab '최소 손끝 틈(mm)' 로 간격을 정하고, 실물이 모델과 다르면 **'손끝 틈 보정' 실측 버튼**(손끝을 서로 닿게 한 뒤 누름)으로 0점을 맞춥니다. 안전망으로 어떤 손가락도 틈 2 mm(명령 1.5 mm) 아래로는 가지 않습니다.
- **3지 방사 파지 — 실측 관절각 기준 10 mm 정지 (09-29 13:16 스크린샷):** 손끝 기하를 고친 뒤에는 모델과 실물이 맞았지만(모델 개구 21 mm = 실물), 가드가 **명령 목표** 기준으로 멈추는 반면 실제 손가락은 명령보다 약 5°(≈17 mm) 뒤쳐져 멈춰 "표고가 있어도 못 잡을 만큼" 넓게 섰습니다. 이제 명령 목표는 최소 개구보다 20 mm 더 좁혀도 허용하고, **실측 관절각의 개구가 10 mm 에 닿는 순간** 그 실측 자세를 목표로 다시 명령해 손가락을 그 자리에 고정합니다(로그 "파지 중단(실측 관절각)"). 실측 개구가 25 mm 안으로 들어오면 접촉 전이라도 정밀 스텝(1°)으로 바꿔 넘침을 줄입니다. 표고가 있으면 접촉력(1.5 N)에서 먼저 멈춥니다.
- **3지 방사 파지 — 손끝 기하 수정 · 실측 관절각 가드 (09-29 13시 스크린샷):** 손끝이 겹친 실측 자세(J3 81°·J4 70°·J1 ±30°)를 FK 로 다시 계산하니 원인이 드러났습니다. 지선단 모듈은 tip 링크 원점에서 손가락 방향으로 28.5 mm 뺀 22.4 mm 폭의 상자인데, 지금까지 개구 계산에 쓴 '캡 중심'(원점 + 6 mm, hand-eye 캘리브용)은 모듈 밑동 근처라 J4 가 굽을수록 실제 손끝 끝이 모이는 것을 보지 못했습니다(그 자세에서 모델 개구 22 mm, 실물은 겹침). 이제 손끝 원·개구·접촉 판정은 **모듈 끝점(원점 + 25 mm)과 모듈 폭 22.4 mm** 로 계산하며, 같은 자세에서 -16 mm(겹침) 로 실물과 일치합니다. 방사 준비 자세의 "손끝 원 ø" 도 끝점 기준으로 바뀌어 같은 갓 지름에서 조금 더 벌립니다(갓 60 → J3 72.8°). 또 명령 J4 43° 인데 실측 J4 70° 였던 것처럼 명령과 실체가 다를 때를 대비해 **실측 관절각의 개구도** 매 주기 검사해 최소 미만이면 즉시 멈춥니다(로그 "파지 개구 가드(실측 관절각)"). 손끝 접촉 폭 기본값은 22.4 입니다.
- **3지 방사 파지 — 손가락 방향 유지 · J3 한계 실측 (09-29 오후):** ① 동기 닫힘이 시작할 때 실측 관절각으로 12축을 다시 보내 준비 자세 이동이 덜 끝난 관절(F2 J1 -30° / 목표 -60°)이 그 자리로 재

명령돼 **손가락 방향이 틀어지던** 문제를 고쳤습니다. 이제 J1·J2 는 마지막 명령값(준비 자세)을 유지하고 J3·J4 만 닫습니다. ② FK 손끝 모델이 실물과 어긋나(모델상 손끝 중심 간 70 mm 인 자세에서 실제 접촉) 개구 가드만으로는 부족하므로 **관절각 기준 상한**을 두었습니다: 📄 Lab ▶ '3지 동기 닫힘 J3 한계(°)' 와 '**실측(지금 닿은 J3 - 4°)**' 버튼. 절차: 표고 없이 ⑤ '3지 방사 동기 파지' 를 실행해 손끝이 서로 닿아 멈추면(잔차 경고가 나는 상태) 📄 Lab 에서 이 버튼을 누르세요. 세 손가락 실제 J3 최솟값 - 4°(≈10 mm)가 저장되어 이후 파지는 그 각도에서 멈춥니다(로그 "J3 한계 ... 도달(손끝 접촉 보호)"). 준비 자세·J4 비율을 바꾸면 다시 실측하세요. 0 = 한계 없음.

- **VTK 콘솔 경고:** "Resetting view-up since view plane normal is parallel" 등 VTK 메시지는 콘솔 대신 logs/vtk.log 로 갑니다(오류 이상만 콘솔).
- **손끝 접촉 폭 보정(09-29 실측):** 3지 방사 파지에서 계산 개구 9.8 mm 인데 손끝이 실제로 닿았습니다. 120° 배치에서는 손끝(F/T 모듈·캡 측면)이 **옆으로 먼저** 부딪히므로, 개구를 "잡을 수 있는 물체 지름" 과 "손끝 중심 간 거리 - 접촉 폭" 중 작은 값으로 판정하도록 바꾸고 접촉 폭 기본값을 실측 19 mm 로 두었습니다(📄 Lab ▶ '손끝 접촉 폭(mm)'). '**실측(지금 손끝이 닿은 상태)**' 버튼: 손끝 3개를 서로 닿을 때까지 오므린 뒤 누르면 현재 관절각으로 손끝 중심 간 최소 거리를 재어 접촉 폭으로 저장합니다(FK 모델 오차 흡수). 차단 문구에 손끝 중심 간 거리가 함께 남습니다.
- **최소 파지 개구(대 굵기) — 빈 파지에도 손끝이 닿지 않음:** 표고 수확용 대 굵기는 10 mm 이상이므로, 모든 파지(소프트·3지 방사 동기·갓 감싸기·11자·2지 동기)가 **손끝 개구(캡 표면 기준)를 최소값(기본 10 mm) 아래로 좁히지 않습니다**. 표고가 없는 채 파지를 실행하면 개구 10 mm 에서 멈추고 "파지 중단 — 손끝 개구 ... mm < 최소 10 mm(대 굵기): 표고/대 없음(빈 파지 보호)" 로 끝납니다. 값은 📄 Harvest Lab ▶ 파라미터 '**최소 파지 개구(대 굵기, mm)**'(harvest.min_grip_opening_mm, 0 = 끄)이며 🍌 단계별 창 로봇 측 수치에도 있습니다. 3지는 세 캡의 내접원 지름, 2지는 두 캡 사이 거리로 계산하고(FK), 역할 배정(Finger 교체)도 반영됩니다.
- **장시간 실행 시 느려짐 · 5 초마다 "COMM_LOST → Task stop" 반복 (진단 + 1차 대책):** 콘솔에 "[TelemetryBus] RB1 COMM_LOST 상태 채널 끊김(위치독 3 s)" 과 "Task stop (emergency)" 이 약 5 초 간격으로 반복되면 로봇이 끊긴 것이 아니라 **GUI 스레드가 몇 초씩 멈춰** 수신 샘플 처리가 밀린 것입니다(클릭이 늦게 먹는 증상과 같은 원인). 대책: ① 위치독이 **데이터 포트 수신 스레드의 실제 수신 시각**으로 끊김을 판정하고(GUI 지연은 끊김 아님), 정지 명령은 끊김 1회당 30 초에 한 번만, 별도 스레드에서 보냅니다. ② Alarms & Events · Fleet Dashboard · Robots & Drivers 가 알람/상태 변화마다 표를 통째로 다시 만들던 것을 300 ms 로 합치고 **화면이 보일 때만** 갱신합니다. ③ Robot 3D 뷰가 장면을 다시 만들 때마다 그리퍼 텔레메트리를 중복 연결하던 것을 막았습니다. ④ **GUI 정지 감시:** 이벤트 루프가 1.5 초 이상 멈추면 로그에 "GUI 이벤트 루프 정지 ... 메인 스레드 스택" 으로 **무엇이 붙잡고 있었는지** 함수 이름이 남고, 복귀 시 정지 시간이 기록됩니다. Robot 3D 렌더 1회가 150 ms 를 넘으면 OpenGL 렌더러 이름과 함께 경고를 남깁니다(소프트웨어 렌더링·원격 데스크톱 확인용). **원인 확정(12:39 로그):** GUI 정지 스택이 🍌 Gripper 3D 뷰의 힘 라벨 갱신(pyvista add_point_labels) 을 가리켰습니다. 이름 없이 부르면 pyvista 가 호출마다 점 액터를 하나씩 남기고(라벨 액터만 지워짐), 힘 값이 흔들릴 때마다 다시 그려 수천 개가 쌓이면 이름 조회가 전체를 순회해 호출 1번에 2 초 가 걸렸습니다. 고정 이름으로 교체하도록 고쳤고(📄 3D 점군 뷰의 후보/경로/강조 라벨도 같은 방식으로 정리), 재발 방지 테스트가 이름 없는 호출을 막습니다. 다른 정지가 보이면 같은 로그 줄을 보내 주세요.
- **Gripper 3D 뷰 콘솔 경고:** '위' 뷰에서 "Resetting view-up since view plane normal is parallel" 이 렌더마다 찍히던 것을 시선 2° 기울임으로 없앴습니다(Robot 3D 와 같은 방식).

v2.0 — 범용 3D Pick&Place (2026-09-16 · 현장 개선 09-17~26, Build260926)

카메라·물체·그리퍼·적재 방식에 독립적인 **범용 3D 픽앤플레이스**가 추가되었습니다. 표고 수확은 그 위의 한 "스킬+프로필"로 재배치되었고 기존 표고 화면·동작은 그대로입니다.

메뉴 재구성 (기능 삭제 없음)

- **RUN** — Job Runner · Shiitake Harvest(표고 수확) · Quality Dashboard · Fleet Dashboard · **Robot Monitoring**(구 Telemetry · 20 Hz: Live 차트 · **3D View**(3D 모델 + 관절/TCP/손끝/상태, 보기 전용) · Joint Status · Activity Stats — 09-21 Robots 의 Info·Activity Stats 탭 이동) (운영 화면만)
- **SETUP** — Robot Control(구 Robots: Overview·3D Model만) · Teaching · Gripper (DG-3F-M) · **Tool / Vision**(SYSTEM에서 이동) · Camera · Trajectory Capture
- **PICK & PLACE**(2026-09-24 재구성) — 학습 기반 일반 3D P&P 표준 흐름 순서:  **Object Model (Training)**(클래스 자유) →  Template Modeling (optional) →  3D Recognition · Comparison →  **3D Pick&Place** →  2D Pick Correction → Legacy(Bin Picking · PA Engine Jobs). 표고 화면은 **SHIITAKE HARVEST** 그룹(Harvest Lab · Harvest Teaching). 이전 나열(참고):  **템플릿 모델링**(신규) ·  표고 모델(학습) ·  **3D 인식 · 최신기술 비교**(신규) · Bin Picking · 2D Pick Correction(구 Pick Guidance) · 표고 수확 검증 · 표고 수확 티칭 ·  **3D Pick&Place 통합**(신규, 09-18 RUN에서 이동 — ②~⑦ 통합: 레시피 편집·단계별 검증·러너 실행) · **PA Engine 잡 (레거시)**(구 3D CAM - Jobs — 계획·검증 없이 그래프 포즈로 직접 이동하는 v1.3 런타임. 새 작업은  통합에서 인식 `pa_graph` + 스킬 `correct_trajectory` (인식 편차로 티칭 궤적 보정 재생) 또는 `simple_pick`). 그룹 헤더에 표시되듯 ①캘리브·③촬영은 SETUP ▶ Camera 에서 하며, 각 메뉴에 마우스를 올리면 담당 단계가 톨팁으로 보입니다.
- **INSPECT** — fineAi PA Engine · Object Detection · Anomaly Detection · Hole Inspection · Capture Quality · Model History
- **SYSTEM** — **Safety Vision**(GUIDE에서 이동) · Factory I/O · SmartFarm Link · Reports / SPC · Digital Twin · **Startup**(09-21: 시작 시 로봇/그리퍼 자동 연결 · 첫 화면 · Robot Monitoring 첫 탭)

3D Pick&Place 통합 (PICK & PLACE)

1. ① **레시피** —  편집: 타깃 소스(로컬 스캔 / SmartFarm claim / 외부 TCP / 픽플랜 재생), 인식(평면 군집 · 학습 모델 · 템플릿 PPF 매칭 · 원기둥/상자 · PA Engine 그래프), 그리퍼(시물 · DG-3F-M · DO 공압 · **OnRobot 2FG7** · 진공), 픽 스킬(단순 파지 · 스텝 하강+F/T · 비틀어 따기), 적재(지정 좌표 · 다점 순환 · 격자 팔레타이징 · 바구니 저녁차), 속도/파지력 한계, 작업영역, 도달·충돌 검증.
2. ② **스캔** —  촬영(관찰 자세 이동 옵션) 또는  PLY. hand-eye 캘리브가 있으면 베이스 좌표로 변환됩니다.
3. ③ **인식** — 대상 표(치수·점수). 3D 뷰에 대상 축 오버레이.
4. ④ **파지 계획·검증** — 대상마다 6DoF 후보(방위 12 + 상단 + 물체축 × 높이 × 롤 × 모드)를 만들고 폭 한계·지지면 관통·작업영역으로 걸러 점수를 매긴 뒤, 상위 후보를 도달성(IK·관절한계)·충돌(캡슐·

환경·지지면)로 지면 검증합니다. 우측 후보 표에서 선택하면 접근→파지→들어올림 경로가 3D에 표시됩니다.

5. ⑤ **적재·픽플랜** — 적재 요약(잡은 높이 보정·안전 고도). 📁 내보내기는 Robo3DV 호환 픽플랜 JSON(version 1) + latest.json, 📄 불러와 재생은 외부 트윈이 만든 픽플랜을 그대로 실행합니다.
6. ⑥ **실행** — 단일 배치 / 연속 / 계획만 · 드라이런. ▶ 시작 · ■ 중지 · || 일시정지 · ↺ 스킵 · ● E-STOP. 확인 게이트(레시피 옵션)에서 계속/이전/처음/중단. 🏠 선택 후보 1회 실행은 러너 없이 한번만 픽→적재합니다.

3D 뷰 ROI 맞춤 (09-23) — 3D 뷰 왼쪽 버튼 옆에 **ROI** 가 생겼습니다. 누르면 CAM(카메라 눈) 방향으로 보면서 ROI 상자만 화면에 딱 차게 줌인·중앙 정렬합니다 (🏠·Camera·🏠 공용 뷰어). 🏠 표고 수확 티칭은 인식 ROI 가 켜져 있으면 촬영+검출 뒤 3D 를 처음 보여줄 때 자동으로 ROI 에 맞춥니다. FIT 은 전체, CAM 은 방향만 바꿉니다.

🎓 **높이·대 길이 정확도 (09-23)** — ① 검출 경로(🎓·📏·🔴 러너)가 갓 축 사전지식(중력 방향, hand-eye 로 카메라 좌표 변환)을 받지 못해 PCA 축만으로 높이를 재던 문제를 고쳤습니다(현장: #6 높이 16 mm → 실제 30 mm; 수정 후 29 mm). 핸드아이가 없으면 예전처럼 PCA 로 폴백하며 로그에 경고가 남습니다. ② 세그 마스크의 배경 혼입 제거(앞면 층). ③ **대 길이 3D 직접 측정**: 갓 축을 따라 5 mm 빈마다 갓 옆면 띠(반경 0.75·r~r+2)와 대(반경 <0.5·r) 점을 세어 갓 두께를 재고 대 길이 = 배치면 거리 - 측정 갓 두께로 씁니다(표의 "대 mm" 열, 3D 측정이면 * 표시). 대가 가려져 측정이 안 되면 설정 갓 두께(20 mm)로 계산합니다. ④ 📏 **크기 보정(기준 표고)**: 후보를 선택하고 실제 갓 지름/높이를 넣어 + 를 누르면 지름 비율·높이 직선(기준 2개 이상) 보정이 smartfarm.yaml harvest.size_* 에 저장되어 모든 인식 경로에 적용됩니다(🔄 초기화로 해제). ⑤ 실수확 검증에 **속도 %** 를 두어 드라이런 포함 검증 이동 속도를 조절합니다(운영 러너는 harvest.speed_pct).

📺 **라이브 화면의 후보 위치 (09-23)** — 라이브(📺)는 2D 원본 영상(시야가 넓고 해상도가 다름)이고 검출 후보는 3D 정렬 영상 픽셀이라, 예전엔 비율만 맞춰 그려 원과 라벨이 표고에서 벗어났습니다. 이제 촬영+검출 때 두 영상의 특징점을 맞춰 픽셀 변환(호모그래피)을 구해 두고 라이브 위에 그 변환으로 후보·ROI 를 그립니다(표시 전용, 인식·좌표에는 영향 없음). 로그에 "라이브 오버레이 매핑(정렬 → 원본) 인라이어 n/m" 이 남고, 매칭이 부족하면 예전 방식으로 그립니다. 카메라나 로봇이 움직이면 촬영+검출을 다시 해야 위치가 맞습니다.

🎓 **실수확 검증의 드라이런 (09-22 정리)** — 드라이런은 **로봇 이동은 실제로 하고** 그리퍼 동작(벌림·파지·해제)·F/T 판정·단계 하강(갓 접촉 감지)·바구니 격자 오프셋/하강을 생략하는 "경로 리허설"입니다. 전체 수확 단계에서는 드라이런도 바구니 자세까지 이동합니다(해제는 생략, 결과에 "바구니 이동(드라이런: 해제 생략)" 표시). 드라이런 기록은 검증 이력에 남지 않습니다. 포인팅(+80 mm)은 드라이런 여부와 무관하게 대 길이 게이트를 통과합니다.

📷 **촬영+검출 진행 표시 · 빠른 후보 선택 (09-22)** — 📷 촬영+검출을 누르면 촬영이 끝나는 즉시 이미지(3D 보기면 점군)가 먼저 표시되고 상태줄에 "📷 3D 촬영 중... → 촬영 완료 N점 — 🗨 인식 중(모델 ...) → 인식 n건 — 픽셀 투영·베이스 변환 중... → 검출 n건 (베이스 변환 n건)" 순으로 진행이 보입니다. 후보 행을 클릭할 때는 기본 오버레이를 캐시해 선택 후보만 빨강으로 덧그리고(1420² 이미지 기준 약 70 ms → 15 ms), 3D 는 축·라벨을 유지한 채 선택 화살표만 갱신합니다(약 230 ms → 20 ms). 표에서 선택된 행은 빨간 배경·흰 글씨로 표시되며, 여러 행을 선택하면 이미지에서도 모두 빨간 원으로 강조됩니다. 보이지 않는 뷰(2D/3D)는 다시 그리지 않습니다.

🎓 **후보 여러 개 선택 → 순차 실수확 검증 · 3D 한글 라벨 (09-22)** — 후보 표에서 Ctrl/Shift로 여러 행을 고른 뒤 ▶ 실수확 검증을 누르면 표 순서대로 차례로 검증합니다(확인 대화 → 진행 중 "순차 검증 n/m — 후보 #k" 표시 → 결과 "순차 검증 m건 — 성공 n · #1✓ #3X"). 각 후보는 자기 구간의 티칭 규칙(화면 선택값 우선)으로 실행되고, 단계 확인에서 중단(X)하거나 취소하면 그 시점에서 멈춥니다. 한 개만 선택하면 예전과 같습니다. 3D 뷰 라벨은 VTK 기본 글꼴에 한글이 없어 클래스명이 비어 보였던 문제를 시스템 한글 글꼴(맑은 고딕)로 고쳤습니다.

🎓 **파지 티칭 규칙 → 운영 수확에도 적용 (09-22)** — 갓 지름 구간(소 <45 / 중 45~65 / 대 ≥65 mm, 클래스 전용 규칙 우선)별로 저장한 파지 프리셋·파지높이 보정·벌림 여유가 RUN ▶ Shiitake Harvest(관제·독자 모드)에서도 대상마다 자동 적용됩니다(검증 화면과 같은 함수). 규칙은 사이클 프리셋보다 우선하며 상태에 "티칭 규칙 [중] 적용 — 프리셋 ..." 이 표시됩니다. 규칙이 없는 구간은 운영 설정 그대로입니다. 검출 결과("검출 n건 (베이스 변환 n 건)")는 파란 굵은 상태줄에, 라이브 해상도·fps 는 그 아래 회색 줄에 따로 표시되어 서로 덮지 않습니다. 🏠 관찰 자세로 이동은 완료/실패와 소요 시간을 표시합니다.

🎓 표고 수확 티칭 — 📦 3D 보기 · ROI 직접 조정 (09-22) — 촬영+검출 뒤 📦 3D 버튼으로 2D 이미지와 3D 점군을 전환합니다. 3D 는 📦 /Camera 와 같은 뷰어(툴바: RGB/밝기/깊이 컬러맵·점 크기·CAM/ISO/FIT)이며 후보 위치가 축+구 마커로, **모든 후보에 라벨**("표시" 콤보의 번호/지름/클래스 내용·색·글자 크기를 2D 와 같이 따름)이, 선택 후보는 굵은 축과 접근 화살표로, 인식 ROI 는 상자로 표시됩니다. 🖱️ **3D 뷰에서** 조정을 하면 주황 상자를 드래그해 ROI 를 정할 수 있고 놓을 때마다 수확 설정(roi_box)에 저장됩니다(상자가 없으면 점군 가운데 60% 로 시작). 📌 **프리셋 저장**으로 이름을 붙이면 Camera ▶ 3D ROI · 📦 · 📌 수확 · 🔍 검증에서 같은 프리셋을 씁니다. ↗️ **맞춤**은 2D/3D 각각의 화면 맞춤입니다. 조정을 잘못했으면 ↶ **프리셋 복원**(또는 프리셋을 다시 선택)으로 저장 안 한 상자를 버리고 프리셋의 저장값으로 되돌립니다(조정 모드는 자동 종료).

🎓 표고 수확 티칭 — 시나리오 (09-22) — 화면 맨 위 "시나리오" 그룹에서 지금 설정 묶음을 이름으로 💾 **저장**(메모 가능, 같은 이름은 덮어쓰기 확인), 📁 **적용**(확인 후 현재 설정으로 되돌리고 화면을 다시 읽음), 🖱️ **이름** 변경, 🗑️ **삭제**할 수 있습니다. 시나리오에는 3D 카메라·인식 모드/모델/conf·인식 ROI· 관찰/바구니 자세·작업 방향·파지 파라미터·🚫 **제한·실행** 소스(관제/독자)와 독자 옵션·구간별 파지 규칙이 들어가며, 장비 고유값(핸드아이 장착 추정·모니터 포트)은 시나리오와 무관하게 유지됩니다. 적용하면 smartfarm.yaml harvest 절과 파지 규칙 파일에 써지므로 📌 표고 수확(관제·독자) · 🔍 검증도 같은 설정을 씁니다. 저장처: data/harvest_scenarios.json.

🎓 표고 수확 티칭 — 📺 **라이브** (09-22 수정) — 라이브 버튼이 카메라를 자동으로 열고 2D 프리뷰를 시작합니다(📌 표고 모델 ① 라이브와 같은 구현). 2D 프리뷰가 실패하면 전체 촬영(3D 포함, 느림)으로 자동 폴백하고, 상태줄에 "라이브 W×H · 2D 프리뷰/3D 촬영 폴백 · ms · fps" 와 실패 원인이 표시되며 포기하지 않고 재시도합니다. ↗️ **맞춤** 버튼은 휠로 확대해 일부만 보일 때 화면에 맞춤니다(라이브 시작 시 자동 맞춤). 촬영+검출 이미지는 3D 정렬 영역(깊이 그리드, 예 1420×1420)이라 Camera 2D 원본(예 2000×2000)보다 시야가 좁은 것이 정상입니다. 예전에는 카메라가 아직 열려 있지 않으면 5초 뒤 "촬영을 먼저 실행하세요" 로 멈춰 라이브가 켜지지 않았습다. 실패하면 원인(카메라 open 실패·빈 프레임·미선택)이 상태줄에 표시됩니다.

🎓 표고 수확 티칭 — 🔍 **검출 오버레이 표시 옵션** (09-22) — 촬영 버튼 아래 "표시" 행에서 라벨 내용(번호만 / **번호+지름**(기본, 예 "#1 34") / 번호+클래스+지름), 글자 크기(px, 0 = 자동: 이미지 폭/50), 색(노랑 기본·청록·연두·자홍·흰색·주황), 원 표시 여부를 고를 수 있습니다. 라벨은 검정 테두리로 배지색 위에서도 읽히고, 홀수/짝수 후보를 원 위·아래로 번갈아 배치해 겹침을 줄였습니다. 선택 후보는 항상 빨강. 설정은 기억됩니다.

🎓 표고 수확 티칭 — 3D 카메라 선택 · 인식 ROI (09-22) — 라이브 화면 아래에 3D 카메라 콤보(현재 수확 설정 camera_id 표시, 바꾸면 즉시 저장)와 인식 ROI 적용 체크 + 프리셋 콤보가 생겼습니다. ROI 프리셋은 SETUP ▶ Camera ▶ 3D ROI(3D 뷰에서 상자 드래그) 또는 📦 P&P 에서 저장한 것을 그대로 쓰며, 프리셋 만드는 순서: Camera ▶ 3D ROI 에서 3D 촬영 → [3D ROI] 체크(클라우드 범위로 상자 생성) → [클라우드 맞춤]·[❖ 박스 편집]·수치 조정 → 이름 입력 → 📁 저장. 상자가 0 크기인 채 저장하면 거부됩니다(예전에 "표고버섯" 이 0,0,0,0,0,0 으로 저장돼 티칭에서 "상자 없음" 이 됐던 문제). 커면 상자 안의 점·이미지만으로 인식합니다(3D 좌표는 그대로). 촬영+검출 후 이미지에 청록 ROI 사각형이 그려지고 상태줄에 "ROI 적용: 점 N/M" 이 표시됩니다(상자가 없거나 비어 있으면 "⚠ ROI 미적용"). 라이브 영상은 인식 대상이 아니므로 ROI 로 잘리지 않습니다. 이 설정은 smartfarm.yaml harvest 절(camera_id · roi_enabled · roi_box · roi_preset)에 저장되어 RUN ▶ Shiitake Harvest(관제·독자 모드)와 📡 표고 수확 검증의 인식에도 그대로 적용됩니다. 빈 상자(0 크기) 프리셋은 적용되지 않고 안내만 표시됩니다.

티칭 자세 저장 위치 이동 (09-22) — 관찰 자세·바구니 자세·작업 방향 저장(현재 로봇 자세 캡처)과 "🏠 관찰 자세로 이동"은 📡 표고 수확 검증에서 🎓 표고 수확 티칭 화면(라이브 화면 아래 "티칭 자세" 패널)으로 옮겼습니다. 저장처는 그대로 smartfarm.yaml harvest 절 이라 📡 표고 수확(관제·독자 모드)·📡 검증이 같은 값을 쓰고, 검증 화면에는 현재 저장값 요약만 표시됩니다. 러너의 수확 대기(관찰) 위치가 생각과 다르면 이 패널에서 다시 저장하세요.

📡 표고 수확 — 독자 모드 (09-22) — RUN ▶ Shiitake Harvest 상단 실행 소스에서 독자 모드(카메라 인식만으로 순회)를 고르면 SmartFarm 관제 없이 동작합니다: 관찰 자세 → 3D 촬영 → 학습 모델 인식 → 신뢰도가 가장 높은 후보부터 1개씩 수확(검증·계획·수확·적재는 관제 모드와 같은 흐름) → 재스캔. 옵션: 최대 수확 수(0 = 제한 없음) · 대상이 없을 때 재스캔 대기 시간 · 대상 없으면 종료(끄면 상시 대기). 실패한 대상 주변(30 mm)은 이번 실행에서 다시 시도하지 않고, 우측 "인식 후보" 표에 최근 스캔 후보와 제외(X)가 보입니다. 결과는 이력·통계에만 남고 SmartFarm 으로 보내지 않습니다. 설정은 smartfarm.yaml harvest.source(local/smartfarm)·local_* 에 저장됩니다. 관찰 자세·핸드아이·카메라 준비 조건은 관제 모드와 같습니다.

표고 수확은 기본적으로 📡 표고 수확 화면(기존 러너)이 담당합니다. 📦 화면의 shiitake_harvest 레시피(smartfarm.yaml harvest: 자동 변환)로 ▶ 시작하면 범용 러너가 claim → 모델 인식 → 비틀어 따기(기존 수확 모션 위임) → 바구니 적재로 병행 운전합니다.

🗨 ④ 3D 인식 · 최신기술 비교 (PICK & PLACE)

한 장면(📷 촬영 / 📁 스냅샷 PLY)에 여러 인식 방법을 **동시에** 돌려 비교합니다: **학습 모델**(YOLO-seg) · **SAM 분할**(FastSAM everything + 3D 높이 씨앗 MobileSAM — 학습 불필요, 모델이 놓친 물체를 잡음) · **오픈 보케블러리**(YOLO-World 텍스트 프롬프트 "box, bottle..." — 학습 불필요) · 평면 군집 · 템플릿 PPF. 방법별 건수·처리시간 표, 2D 색 오버레이(파랑 모델·초록 SAM·주황 YOLO-World·☉ 융합), 융합 표(동의 방법 수). 정책 **union**(하나라도) / **vote**(2개 이상 동의) / **priority**(풀백). ✓ **레시피에 적용**을 누르면 선택 레시피의 인식 계층이 **ensemble** (멤버·가중치 포함)로 저장되어 📦 3D Pick&Place 운영에 그대로 쓰입니다. 표고는 SAM 멤버에 **profile: shiitake** 를 주면 학습 모델과 같은 갓 축·대 길이 규칙으로 변환됩니다(데모 레시피 **shiitake_ensemble_offline**). 가중치 파일은 **assets/foundation_models/** .

현장 강건화 (레시피 🗨 편집 ▶ ⑥)

- **손가락 자리 간섭 검사** — 파지 자세에서 손가락이 들어갈 자리에 장면 점군(이웃 물체·상자 벽·배지 프레임)이 있으면 그 후보를 버리고 다음 후보를 씁니다. 여유 mm(0 = 끼기).
- **촬영 누적 프레임 · 홀 채움** — 같은 자세에서 N장 찍어 합치고(홀 상호 보완·노이즈 억제), 작은 홀만 이웃 보간합니다(큰 홀은 지어내지 않음).
- **대안 관찰 자세** — 인식이 부족하면 지정한 관절 자세들로 순서대로 재촬영.
- **파지 실패 복구 사다리** — 빈손/힘 초과 시 같은 대상의 다음 검증 후보로 재시도(횟수), 후보 소진 시 재스캔·재인식 후 같은 물체를 한 번 더. 시도 이력은 사이클 결과에 남습니다.

✳ 템플릿 모델링 (PICK & PLACE)

점군 색(② ROI 그룹) — 촬영/조각화 PLY 소스는 **텍스처(카메라 RGB)**로 표시되어 물체를 눈으로 찾기 쉽습니다. 정제·정규화 후에는 점 대응이 사라져 **높이(Z 컬러맵)**로 자동 전환되며, 단색도 선택할 수 있습니다. 다른 3D 뷰의 xyz 점군도 단색 대신 높이 컬러맵으로 표시됩니다(v2.0). 촬영/PLY 소스는 Camera 화면·3D Studio 와 같은 **카메라 눈 방향**(2D 이미지와 같은 상하좌우)으로 열리며, 3D 뷰 왼쪽의 **CAM** 버튼으로 언제든지 그 방향으로 되돌립니다. 정제·정규화·파지점 단계에서도 시점은 바뀌지 않습니다 — 정규화는 내부 데이터만 물체 좌표(중심 원점·축 정렬)로 바꾸고 화면에는 카메라 좌표로 되돌려 그리므로 물체가 제자리에 보이며, 파지점 축도 같은 자리에 표시됩니다(저장되는 템플릿은 물체 좌표). ⑤ **파지점**은 표에서 편집하는 것 외에 🗨 **화면에서 지정**으로 점군을 직접 클릭해 만듭니다 — **1점**(파지점, 위에서 접근) · **2점**(접근 시작 → 파지점) · **표고**(갓 상면 중심 → 대의 파지 지점, 접근축 = 갓 축). 파지점은 주황 구와 축(선택 행은 노란색·굵게), 번호 라벨로 표시되고 정규화 후에도 유지됩니다. 정제(③)는 KD-tree 이상치 제거로 수십만 점도 수 초에 끝나며, 실행 중 로그와 완료 점수가 기록됩니다. 정제를 여러 번 눌렀거나 잘못 잘랐으면 🔄 **되돌리기**(Ctrl+Z)로 ROI 자르기·정제·정규화·원본 복원을 한 단계씩(최대 12단계) 취소합니다. 새 소스를 열면 되돌리기 이력은 비워집니다.

매칭용 물체 템플릿(점군 + 파지점)을 만듭니다: ① 소스(촬영/PLY/기존 템플릿) → ② ROI 박스 자르기 → ③ 정제(복셀·이상치·바닥 제거) → ④ 정규화(중심/기준면→+Z/PCA) → ⑤ 파지점(👉 자동 제안, 표에서 편집) → ⑥ 저장 · 📌 현재 장면에서 테스트 매칭(PPF/군집/FPFH). 저장 위치 `data/pick_place/templates/<이름>/`. 레시피 인식 타입 `ppf_match` 의 templates 에 이름을 추가해 사용합니다.

외부 연동 · 헤드리스

- **비전 TCP 서버(:6000)** — `config/pick_place.yaml external_tcp.enabled: true`. 줄단위 JSON `ping / status / models / scan` (robot_base 좌표 반환) + `pick / harvest / home / abort / state` (accept_robot_cmds 가 켜져 있고 external_tcp 레시피가 돌 때만).
- **헤드리스** — `fineai-robodk-run --pick-place <레시피> [--mode continuous] [--dry-run] [--sim]`. 모니터 `:8770` 의 `/api/status` 에 `pick_place` 키로 상태가 실립니다.
- **결과 DB** — Reports / SPC 에서 레시피 `pp:<이름>` 로 사이클 행(판정·사유·힘·시간)을 조회합니다.

현장 개선 — 그리퍼 동작 설정 · 단계 확인 · 카메라 사용 규칙 · EpicEye LED (2026-09-24~26, Build260926)

- **EpicEye Laser V3 야간 LED·노출** — 원인 수정: 라이브 바의 📖 LED·노출·게인과 ⚙ Settings 의 3D 해상도 등이 카메라에 "저장만 되고 적용되지 않던" 문제는 펌웨어가 아니라 앱이 **촬영 모드 번호를 잘못 매핑**(모드 1 = 구조광 섹션인데 레이저 스윙 섹션에 기록)한 것이었습니다. 정정 후 LED ON 에서 2D 평균 밝기 3 → 121, 3D 해상도 모드 0 에서 2840² 격자가 즉시 반영됩니다. 카메라 웹 UI(`:5000/app`) 와 무관하게 앱만으로 LED 가 동작하며, 설정 창 촬영 모드 표기는 **0 비디오 스트림 · 1 구조광(기본) · 2 패시브 양안 · 3 레이저 스윙** 으로 바뀌었습니다(docs/26 §5).
- 🤖 **그리퍼 동작 세부 설정 — 단계별 (09-25)**: 🏠 Harvest Lab ▶ 파라미터의 🤖 **그리퍼 동작 세부 설정(단계별)**..., 🍄 Shiitake Harvest 실행 소스의 🤖 **그리퍼 동작**..., 🎓 Harvest Teaching 의 🤖 **그리퍼 동작**... 이 같은 창을 엽니다. 운영 사이클(⑤ 프리셋)의 단계가 순서대로 나오고(pre-grasp 진입 자세 → 11자 파지 → 비틀어 따기 → 유지 확인 → 해제), 진입 자세는 **F1·F2·F3 × J1~J4 12축**을 직접 고치거나 **현재 그리퍼 자세 가져오기**로 채우고 **그리퍼를 이 자세로 보내** 확인할 수 있습니다. 파지 목표 접촉력·부서짐 한계, 비틀기 각도·횡수·속도·힘 한계, 유지 시간과 🤖 로봇 측 수치(진입 높이·파지 높이 보정·벌림 여유·인출·힘 한계·속도)를 한 화면에서 바꾸고 💾 **저장**하면 바뀐 프리셋은 `data/harvest_grip_presets.json`(★오버라이드 표시, 내장값 복원 가능), 로봇 측 수치는 `smartfarm.yaml` 에 저장되어 🍄·🎓·🤖 에 공통 적용됩니다. 우선순위는 🎓 티칭 규칙/검증값 > 사이클 프리셋 > Lab 수치이며, 비틀기 각도는 갓 지름에 따라 자동 보정(×0.6~1.5)됩니다.
- **단계별 확인 대기 — 드라이런에서도 동작 (09-25)**: 🍄 Shiitake Harvest 의 '단계별 확인 대기' 가 드라이런에서는 무시되고, 집중 보기에서는 확인 바가 접힌 왼쪽 패널에 숨어 있어 "동작하지 않는" 것처럼 보였습니다. 이제 체크가 켜져 있으면 드라이런에서도 진입/파지/비틀기/인출/적재 전에 화면 **가운데 위** '△ 단계 확인 대기' 바가 나타나 계속·이전·처음·중단을 고릅니다(그리퍼 없이 이동 자세를 단계마다 확인할 때 유용). 🎓 Harvest Teaching 에도 드라이런 옆에 같은 설정의 **단계 확인** 체크를 두었습니다(🍄·🤖·🎓 모두 `harvest.confirm_steps`).
- **카메라는 보이는 화면만 사용**: Camera·Teaching 등 라이브 페이지를 떠나도 공용 프리뷰가 뒤에서 계속 카메라를 촬영해, 🍄 Shiitake Harvest(RUN) 에서 라이브를 끄고 수확이 끝난 뒤에도 EpicEye LED 가 계속 번쩍이던 문제를 고쳤습니다. 메뉴를 옮기면 공용 프리뷰가 멈추고, 라이브 페이지로 돌

아오면 자동 재개됩니다(떠 있는 라이브 창은 유지). 🏠·🎓·📦·🔍 화면의 🖥️ 라이브도 같은 규칙입니다 — 다른 메뉴로 가면 꺼지고(버튼 해제), 돌아오면 켜져 있던 경우 자동으로 다시 켜집니다. 카메라 연결 자체는 유지되어 다음 시작이 빠릅니다.

현장 개선 · P&P 재구성 · RB5 제어 SDK (2026-09-24, Build260924)

- **제어박스 SW 버전:** Robot Control ▶ Overview 에 **Controller SW** 행 — 연결 직후 `get_control_box_info` 로 읽은 컨트롤러 소프트웨어 버전·박스 종류를 표시하고 운영 로그에 남깁니다('—'면 펌웨어 미응답).
- **TCP 오프셋 · 페이로드:** Tool / Vision ▶ 🤖 **로봇 TCP 오프셋 · 페이로드** 그룹 — 체크한 항목만 `robots.yaml` (tcp_mm_deg / payload)에 저장되고 **재생 준비(제어권 획득·런 시작)** 때마다 로봇에 자동 적용됩니다. '로봇에 지금 적용' 은 연결된 로봇에 즉시 전송.
- **충돌 감도 레시피 파라미터:** Harvest Lab ▶ 파라미터 ▶ **충돌 감지 적용 / 감지 켜기 / 감도(0~1) / 충돌 시 동작(일반·회피 정지) / 충돌 후 처리(일시정지·중지)** — '적용' 을 켜면 🏠 수확. 🔍 검증. 🎓 티칭의 첫 이동 전에 로봇에 1회 보냅니다(드라이런 제외). 끄면 컨트롤러 설정 그대로.
- ⚠️ **프리드라이브(직접 교시) — 09-24 사고와 보호 절차:** 첫 사용 때 '켜짐 — 끄기' 를 누르는 순간 **로봇 전 관절이 풀려 쓰러졌습니다**(중력 보상은 컨트롤러에 등록된 페이로드/툴과 실제 장착물이 맞고 서보가 활성화일 때만 성립). 이후 켜기는 **전제 조건 검사**(Real Mode · 이동 중 아님 · E-STOP 래치 아님 · **페이로드 등록**(Tool / Vision ▶ 🤖) · Lock)를 통과하고 **운영자 확인** 창에 동의해야 하며, 켜기 직전 서보 활성화(Activate)과 페이로드/TCP 재전송을 자동으로 하고, **끄는 즉시 서보를 다시 활성화**합니다. 그래도 티치펜던트에서 직접교시가 정상인지 먼저 확인하고, 팔을 받칠 수 있는 자세에서 E-STOP 을 손에 두고 쓰세요. 완전히 막으려면 `robots.yaml freedrive_enabled: false` (버튼 숨김).
- ⚠️ **도달 한계(09-24 사고):** 도달 검사가 스펙 최대(RB5-850 850 mm)까지 허용해 #3·#4 처럼 먼 표고에서 팔이 완전히 뻗으며 관절이 뒤집혔습니다. 이제 Harvest Lab ▶ 파라미터의 **도달 한계(mm, 기본 700)** 보다 먼 TCP 목표와, 현재 자세 대비 한 관절이라도 **관절 급변 한계(°, 기본 90)** 이상 돌아야 하는 IK 해(팔 배치 변경 = 뒤집힘)는 '도달 불가' 로 이동하지 않습니다. 850 mm 는 한쪽으로 완전히 뻗은 값이므로 700 이하를 권장합니다.
- 🎓 **검증 중 후보 실시간 표시:** 실수확 검증이 도는 동안 2D/3D 화면의 해당 후보 위에 주황 화살표(2D 는 0.5 s 깜빡임)와 "#3 🤖 파지중" 같은 단계 문구가 붙고, 끝나면 초록(✓ 완료)/빨강(X 실패)으로 바뀝니다. 다음 검증을 시작하면 지워집니다.
- 🎓 **실수확 검증 진행 표시:** 검증 그룹 아래 진행 목록에 후보별 단계가 시간과 함께 쌓입니다 — "#1 ▶ 검증 시작 — 전체 수확 · 드라이런 · 수확 ø45 · 프리셋", "#1 pre-grasp 이동", "#1 진입(하강)", "#1 소프트 파지 F/T ...", "#1 비틀어 따기", "#1 인출", "#1 후퇴", "#1 바구니 담는 중", "#1 ✓ 성공 — ... (12.3 s)" / "#2 X 실패 — 사유". 상태줄에도 같은 문구가 나옵니다.
- **드라이런 속도 50%:** 실수확 검증의 '드라이런' 을 켜면 속도가 50% 로, 끄면 수확 설정 speed_pct(기본 20%) 로 바뀝니다(직접 고친 값은 유지).
- **검증 동작 로그(시간대별 작업 내역):** 진행 목록의 모든 줄(세션 시작·후보별 단계·성공/실패)이 `data/harvest_log/harvest_teach_steps_YYYYMM.csv` 에 자동 기록됩니다(드라이런 포함, 열: ts-session-cand-kind-stage-dry-text). 진행 목록 아래 📄 **진행 CSV** 는 이번 세션분만 따로 저장합니다(Excel 용 UTF-8 BOM).
- 🎓 **시나리오 → 🏠 러너 그대로 실행:** Harvest Teaching 에서 실수확 검증까지 마친 상태를 📄 시나리오로 저장하면 **검증 화면 선택값(프리셋·파지 높이 보장·벌림 여유, 드라이런이 아닐 때의 속도**

%)까지 함께 저장됩니다. RUN ▶ Shiitake Harvest ▶ 실행 소스에 **시나리오** 선택이 생겨 📁 적용 또는 ▶ 시작 때 그 설정(카메라·ROI·모델·티칭 자세·파지 파라미터·구간 규칙·검증 선택값)으로 바뀝니다. 러너는 화면과 같은 우선순위(검증 선택값 > 구간 규칙 > 사이클 프리셋)로 파지합니다.

- 📦 **레시피 편집기 도움말·제로샷 프롬프트**: 편집기 하단 ? **도움말**(각 그룹의 ? 버튼은 해당 절로 이동)에 항목별 뜻·권장값·"처음 하는 순서"를 정리했습니다. ② 인식의 **프롬프트(제로샷)** 칸에 물체 이름(영어, 심표 구분)을 적고 🔍 **제로샷으로 먼저 시험**을 누르면 JSON 편집 없이 YOLO-World(openvocab)·프로필 "일반"으로 설정됩니다. 저장 후 📦 ② 스캔 ▶ 🔍 시작 전 확인으로 인식을 확인하고, 불안정하면 Object Model 에서 학습해 model3d 로 바꾸는 순서입니다.
- **PICK & PLACE 메뉴 재구성 — 일반 3D P&P 표준 흐름**: 순서가 **Object Model (Training)**(구 Shiitake Model — 클래스 자유, 제품·부품·표고 공용 YOLO-seg 학습) → **Template Modeling (optional)**(정밀 정합이 필요한 강제만) → **3D Recognition · Comparison** → **3D Pick&Place**(레시피 티칭·🔍 시작 전 확인·1회 실행·러너) → 2D Pick Correction → Bin Picking (Legacy) · PA Engine Jobs (Legacy) 로 바뀌었고, 표고 전용 화면(Harvest Lab · Harvest Teaching)은 **SHIITAKE HARVEST (P&P extension)** 그룹으로 옮겼습니다(구현·저장 위치·RUN ▶ Shiitake Harvest 는 그대로, 저장된 첫 화면 설정도 유지). 절차는 docs/27 §7.
- **Object Model (Training) — 클래스 자유화**: 라벨 도구 위 **클래스 목록**(심표 구분, 예 freshener,cap,box)을 적고 적용하면 클래스 버튼이 바뀌고, 새 데이터셋에 그 클래스가 저장되어 모델 names 가 됩니다. 기존 데이터셋을 고르면 그 클래스가 들어오고, 📌 표고 프리셋 버튼으로 harvest/immature/blocked 로 되돌립니다.
- **model3d 로케이터 일반 프로필**: 레시피 편집기 ② 인식의 **인식 프로필**을 "일반 제품·부품"으로 두면 학습 모델의 인스턴스 마스크를 표고 파생값 없이 PCA 상자·상면 중심·법선 접근으로 바꿉니다 (sam_segment·openvocab 와 같은 경로). 흐트러진 규격 제품(방향제 등)을 템플릿 없이 학습 모델만으로 인식·파지하는 표준 경로입니다. "표고" 프로필은 갓 축·대 길이·비틀어 따기 확장을 그대로 씁니다.
- 📦 **3D Pick&Place(강제)에도 📌 러너 기능 이식**: ② 스캔 그룹의 🔍 **시작 전 확인**(촬영→인식→계획·검증을 러너와 같은 경로로 한 번에, 로봇 이동 없음, 결과는 대상 표·후보 순위·3D·`pickplace_detections_YYYYMM.jsonl`, 📦 기록은 `data/3d_pnp_log/` 에 수확 기록과 분리)과 🖥️ **라이브**(3D 뷰 자리에 2D 프리뷰, 실행 중에도 계속·SCAN 단계만 잠시 대기); 3D 뷰 위 **핵심 상태 줄**(FSM·대상·단계·성공/실패/스킵·드라이런)과 실행 중 대상 **화살표**(주황 진행 → 초록 완료/빨강 실패); ⑥ 실행의 **최대 픽 · 실행 시간 · 1개 제한 · 재시도** 스피ن(레시피 run/retry 에 저장 — 편집기 ⑥ 에도 동일 항목 + ⑤ 관절 급변 한계 flip_deg 90°: 도달 검증에서 현재 자세 대비 팔 배치가 바뀌는 IK 해 탈락, reach_max_mm 은 700 권장); 모든 단계 문구가 `data/3d_pnp_log/pickplace_steps_YYYYMM.csv` 에 자동 기록되고 로그 아래 📄 **진행 CSV** 로 세션 분 저장. 위치/높이 보정은 기존 ④ 🎯 **보조 보정(현재 로봇 위치 = 선택 대상)** 이 같은 역할입니다 (표본 평균을 모든 대상에 적용).
- 📌 **위치/높이 보정 (검출 높이 ↔ 로봇 목표 맞추기)**: 실수확 검증 그룹의 📌 **위치/높이 보정 dx/dy/dz(mm)** 는 파지 TCP 목표에 더하는 베이스 좌표 보정입니다(harvest.grasp_offset_mm, lift·파지중심 보상 뒤에 적용, 📌 러너·🔍 검증·📌 공용, 시나리오 포함). **측정 절차**: 후보 선택 → 단계 'pre-grasp 까지'(또는 '파지 확인까지')로 검증해 로봇이 멈추면 → 조그/펜던트로 손끝을 실제 파지 위치에 맞춘 뒤 📌 **현재 TCP = 파지점** → 차이(현재 TCP - 마지막 파지 목표)를 확인하고 저장. 값을 직접 입력해 📄 저장하거나 0 으로 초기화할 수도 있습니다. 동작 로그의 "파지점 → TCP 목표" 줄에

보정값이 함께 찍힙니다. 카메라 전체가 일정하게 어긋나는 경우(캘리브 잔차·TCP 정의)에 쓰고, 표고 크기에 따라 달라지는 오차는  크기 보정(높이 offset)으로 맞추세요.

-  **러너 — 재시도 · 실행 시간 · 1개 동작 제한:** 독자 모드는 원래 **최대 수확 수만큼 성공할 때까지** 스캔→수확을 반복합니다("대상 없으면 종료"를 끄면 빈 스캔도 기다렸다 반복). 이번에 **재시도(회):** 실패한 표고를 이 횟수까지 재스캔 후 다시 시도하고, 다 쓰면 후보 표에 "**실패 주변(재시도 제외)**" (반경 local_skip_radius_mm 30 mm 안의 같은 자리)로 표시해 더 시도하지 않습니다(예전엔 1회 실패로 바로 제외). **실행 시간(분):** 지나면 진행 중인 1개를 마치고 종료(0 = 제한 없음). **1개 동작 제한(s):** 표고 1개의 접근~적재가 이 시간을 넘기면 안전 중단(해제·후퇴)하고 '시간초과' 실패로 기록·재시도 규칙 적용(0 = 없음).
-  **러너 — 수확 중 2D 라이브:**  라이브를 켜 두면 수확이 진행되는 동안에도 카메라 2D 영상이 실시간으로 계속 보입니다(로봇이 따는 장면을 그대로 관찰). 러너가 3D 촬영하는 SCAN 단계에서만 카메라를 양보해 잠시 멈췄다가 자동 재개하고, 라이브 줄에 현재 FSM 단계·대상이 함께 표시됩니다. 후보 번호·ROI 오버레이는 라이브를 끄면(스캔 이미지) 또는  3D 에서 봅니다.
-  **러너 — 수확 속도 슬라이더:** 실행 소스 그룹의 **수확 속도** 슬라이더/숫자(5~100 %)는 harvest.speed_pct 입니다. 시나리오를 적용하면 그 값이 들어오고, 여기서 바꾸면 수확 설정에 저장됩니다( Harvest Lab 속도와 같은 값). 수확 중에 바뀌도 다음 이동 구간(접근/하강/인출/바구니 이동 각각)부터 바로 적용됩니다(진행 중인 한 구간만 이전 속도). 비틀기 속도는 ⑤ 사이클 프리셋 값을 따릅니다.
-  **러너 —  촬영+검출 (시작 전 확인):** 러너와 완전히 같은 경로(촬영 → 인식 ROI → 모델 인식 → 대 길이 게이트 → 클래스·도달성 자동 선택)를 로봇 이동 없이 한 번 실행합니다. 후보 표(클래스·신뢰도·도달mm·제외 사유), 2D 스냅샷(선택 후보 주황 "◀ 수확"), 3D 라벨이 실제 수확과 같게 채워지고, 검출 기록 파일에 source=precheck 로 남습니다. 지금 로봇 자세에서 촬영하므로 관찰 자세에서 눌러야 러너와 같은 시야입니다( 티칭 자세 패널 ' 관찰 자세로 이동'). 로봇이 연결되지 않았으면 도달성은 계산하지 않고 표시합니다.
-  **러너 — 수확 시작 전 3D · ROI 확인:**  3D 를 켤 때 러너 스캔이 없으면 자동으로  **3D 촬영** (러너와 같은 카메라·조명 경로)을 한 번 해 점군과 인식 ROI 상자를 3D 로 보이고, 2D 이미지에도 촬영본과 ROI 상자를 그립니다.  3D 촬영 버튼으로 언제든지 다시 찍을 수 있습니다(러너 실행 중에는 러너 스캔을 표시). 이미지 아래 줄에 촬영 시각·점 수·ROI 적용/꺼짐·카메라가 표시됩니다.
-  **러너 — 집중 보기 · 패널 접기:** 최근 스캔 위에 **핵심 상태줄**(FSM · 대상 · 최근 단계 · 수확/실패/스킵 · 드라이런, 실행 중 초록 / 복구 빨강)이 항상 보이고,  **집중 보기**를 누르면 좌(제어)·우(후보/이력) 패널이 접혀 최근 스캔·라이브·3D 와 핵심 상태만 크게 봅니다(다시 누르면 복원). 좌·우 패널은 경계를 끝까지 드래그해 접을 수도 있고, 오른쪽의 인식 후보/시도 이력 사이도 세로로 조절됩니다(모두 기억됨).
-  **러너 — 드라이런 · 화면 크기 조절 · 수확 후 확인 촬영:** 실행 소스 그룹의 **드라이런** 체크 (harvest.dry_run)는  검증의 드라이런과 같은 동작(접근·하강·비틀기·인출·바구니 이동만, 파지·해제·F/T 판정 없음)으로 러너를 돌립니다. 결과는 성공이 아닌 **스킵**으로 집계되고 관제에 보고하지 않으며 검증 기록에는 남지 않습니다(동작 로그·운영 결과 파일에는 dry=1). 독자 모드에서 드라이런 시도도 실패로 세어 '재시도' 횟수까지 반복한 뒤 제외됩니다(09-24 수정: 예전엔 1회 만에 제외). 후보 표에는 제외 전까지 "재시도 n/한도" 가 표시됩니다. 좌(제어)·중(최근 스캔)·우(후보/이력) 폭은 경계를 드래그해 조절할 수 있고(기억됨), 1개 수확이 끝날 때마다 2D 프리뷰로 현재 장면을 찍어 최근 스캔 이미지를 갱신합니다(harvest.snapshot_after_harvest, 이미지 아래에 "수확 후 확인 촬영 #n" 표시).

-  **러너 화면 — 라이브 · 3D · ROI 상태 · 단계 확인 옵션**: 가운데 표시가  티칭 화면과 같아졌습니다.  **라이브**(수확 설정 카메라의 2D 프리뷰, 러너가 촬영하는 동안은 자동 일시정지),  **3D**(최근 스캔 점군 + 후보 번호/클래스/지름 + 인식 ROI 상자, 선택 후보 주황 화살표), **ROI 맞춤**. 스캔 이미지는 인식 ROI(청록 상자), 후보 원(선택=주황 "◀ 수확", 가능=노랑, 제외=회색+사유)이 그려지고, 아래 파란 상태줄에 지금 러너가 따르는 설정(카메라·모델·ROI·사이클·속도·검증값·도달 한계·클래스·단계 확인)이 보입니다. 실행 소스 그룹의 **단계별 확인 대기** 체크는  Harvest Lab ▶ 파라미터 '단계별 확인' 과 같은 설정(harvest.confirm_steps)입니다 — 무인 연속 수확이면 고세요.
-  **독자 모드 자동 선택 · 연속 수확**: 후보를 사람이 고르지 않습니다. 스캔마다 후보 전부를 베이스 변환·도달성(IK) 계산해 **수확 가능 클래스**(harvest 설정 local_classes, 기본 harvest) 이고 **도달 가능** 한 후보만 신뢰도 순으로 1개씩 수확하고 재스캔합니다. 실패한 대상 주변, 다른 클래스, 도달 불가 후보는 인식 후보 표에 **제외 사유**(클래스·도달mm 열 포함)로 표시됩니다.
-  **러너 기록 =  화면과 같은 파일**: 스캔마다 검출 후보 전부(source=runner, 선택 가능 수·제외 사유 포함)를 harvest_teach_detections_YYYYMM.jsonl, 단계마다 동작 로그("#3 ▶ 수확 시작", "#3 pre-grasp 이동", ... "#3 ✓ 성공")를 harvest_teach_steps_YYYYMM.csv (session = 러너 시작 시각), 결과를 검증 기록 형식(harvest_teach_log_YYYYMM.jsonl, source=runner)에도 남깁니다. 관제 모드에서도 단계·결과는 같은 파일에 기록됩니다.
-  **운영 러너 결과 파일**: RUN ▶ Shiitake Harvest(관제·독자 모드 모두)의 수확 시도 결과(성공/실패/스킵, 사유, 사이클·검출 시간, 갓 지름, 수확량 g, 파지 자세, 그리퍼 상태, 모드·로봇·클래스·누적 통계)를 data/harvest_log/harvest_runner_YYYYMM.jsonl 에 1건 1줄로 남깁니다(관제 전송과 같은 내용).
-  **검증 중 화살표 자동 제거**: 실수확 검증(순차 검증 포함)이 끝나면 후보 위의 화살표·단계 문구가 지워집니다. 결과는 진행 목록과 기록 파일에 남습니다.
- 수확 기록 폴더 data/harvest_log/ (월별 파일 · 20 MB 분할)**: 수확 관련 기록 파일(운영 러너 결과 jsonl · 검증 결과 jsonl · 검출 후보 jsonl · 동작 로그 csv · ⑤ 파지 프리셋 검증 csv)은 모두 이 폴더에 이름_YYYYMM.확장자 로 한 달에 한 파일씩 쌓이고, 한 파일이 20 MB 를 넘으면 같은 달 안에서 _p02, _p03 ... 로 이어 씁니다. 365 일 수확해도 종류당 12개(+분할) 파일이라 백업·삭제가 간단하고, 오래된 달 파일을 지워도 프로그램은 영향이 없습니다(아무것도 자동 삭제하지 않음). 예전 위치(data/ 바로 아래)의 파일은 첫 실행 때 이 폴더로 옮기며 이름에 _legacy 가 붙습니다. 화면 표·요약은 모든 파일을 최신부터 이어 읽습니다.
- 후보·검증 기록 파일**: 촬영+검출 1회마다 후보 전부(클래스·크기·카메라/베이스 좌표·접근 벡터·도달성·티칭 규칙)를 data/harvest_log/harvest_teach_detections_YYYYMM.jsonl 에 1줄로 남기고, 실수확 검증(드라이런 제외) 결과는 data/harvest_log/harvest_teach_log_YYYYMM.jsonl 에 후보 번호(cand)·촬영 시각(det_ts)·베이스/카메라 좌표·도달성 사유를 더해 기록합니다. 두 파일은 det_ts 로 이어집니다.
- 수확 도달성 사전 계산**:  Harvest Teaching 후보 표에 **도달** 열(✓/X + 베이스 원점 거리 mm, 툴팁에 사유)이 생겼고, 실수확 검증  운영 러너는 **이동 전에** RB5-850 URDF 역기구학( P&P 와 같은 reach_check)으로 pre-grasp·파지점(포인팅은 +160/+80 mm 상공) 도달 가능 여부를 계산합니다. 불가면 로봇을 움직이지 않고 '도달 불가 — 사유' 로 실패 기록(예전엔 이동을 보내고 펌웨어 거부/타임아웃 뒤에야 도달불가). 자세 기준은 작업 방향(work_rpy), 없으면 현재 TCP 자세이며, 기준이 없으면 '?' 로 표시하고 진행합니다.
-  **티칭 자세 패널 —  바구니 자세로 이동 ·  작업 방향으로 이동**: 저장 버튼 줄 아래에 이동 버튼 줄을 두었습니다. 바구니 자세는 관찰 자세와 같은 관절 이동(move_j), 작업 방향은 **현재 TCP**

위치를 유지한 채 자세(rx,ry,rz)만 저장값으로 맞추는 직선 이동(move_l)입니다. 모두 로봇 연결·Lock 필요, 완료/실패와 소요 시간을 패널에 표시합니다.

- **조명 제어와 3D 카메라:** 링조명 제어기(lighting.yaml)는 이제 **2D 카메라 촬영에만** 적용됩니다. 3D 카메라(자체 광원)는 촬영 때 제어기에 접속하지 않아, 제어기가 꺼져 있어도 3D 촬영마다 나오던 "조명 연결 실패 ... timed out" 경고·2 s 지연이 사라집니다. 3D 카메라에도 조명을 쓰려면 lighting.yaml의 `controllers[].cameras`에 카메라 id를 적으세요.
- **수정:** Robot Control 속도 바  저장에 robots.yaml에 실제로 기록되지 않던 문제(설정 필드 누락) 수정. SYSTEM ▶ Startup의 **그리퍼 연결**을 꺼도 앱 시작 때 그리퍼 연결을 시도하던 문제(옛 smartfarm.yaml gripper.enabled 자동 시작 경로 2곳: 앱 진입·SmartFarm Link 화면 생성) 제거 — 이제 Startup 설정만 따릅니다.

현장 개선 (2026-09-23, Build260923)

-  **표고 수확 티칭:** 라이브 위 후보 위치 보정(2D 원본↔3D 정렬 픽셀 매핑), 작은 표고 후보 누락 수정(마스크 앞면 층 필터)·제외 사유 로그, 높이·대 길이 정확도(검출 경로에 중력 축 자동 공급 — 16→29 mm 사례), 대 길이 3D 직접 측정·'대mm' 열,  크기 보정(기준 표고), 실수확 검증 속도 %, 3D 뷰 ROI 맞춤 버튼·자동 맞춤.

현장 개선 (2026-09-22, Build260922)

- **Job Runner:** 시작 시 Lock 확인·획득, Startup '연결 후 Lock 자동 획득'. 시작 배너·창 포커스.
-  **표고 수확 독자 모드**(관제 없이 카메라 인식만으로 순화)·티칭 규칙(구간별 프리셋/파지높이/별림 여유)이 운영 러너에도 적용.
-  **표고 수확 티칭:** 티칭 자세 패널(검증 화면에서 이동), 3D 카메라 선택, 인식 ROI(프리셋·3D 뷰 드래그 조정·복원·저장), 라이브 수정,  3D 보기·한글 라벨, 오버레이 표시 옵션, 시나리오 저장/적용/관리, 후보 멀티 선택 순차 검증, 촬영 즉시 표시·진행 문구, 클릭 성능, 포인팅/드라이런 게이트 경고, 드라이런 바꾸니 이동.
- **그리퍼:** F/T·관절 실시간 수신(SDK 콜백 캐시, 20 Hz), 두 손가락 평균 판정 + 미세 정착(목표 ± 0.1 N), ⑤ 프리셋 지연 제거(로봇 준비는 비틀기만·미도달 즉시 판정), 명령 지연 진단 로그.
- **기타:** Camera ▶ 3D ROI 빈 상자 저장 거부, Bin Picking 테스트 파지 자세 유지·/■, Job 재생 그리퍼 동작·촬영 규칙·조명 중지.

현장 개선 (2026-09-18~21, Build260921)

실장비 표고 수확 운전(09-18~21)에서 나온 항목입니다. 각 기능의 상세는 해당 절에 있습니다.

- **인식·접근:** 갓 축(중력 우선)·3D 지지면 높이·배지 법선 직각 진입, 11자 손끝 방향 선택(배지 좌→우/우→좌/평행/자동), 파지점→TCP 로그,  보조 보정(현재 로봇 위치 = 선택 대상), 3D 뷰 실시간 TCP·손끝 표시.
- **그리퍼:** ⑤ 파지 프리셋을 레시피·Teaching에서 사용, 11자 파지 F1/F2 굽힘 동일, 소프트 파지 떨림·팅김 제거(적응 스텝·스톨 감지·예압 램프), 자세 명령은 이전 상태와 무관하게 정의된 모양, 관절 진단 오탐 정리,  스톨 보호,  재초기화, F/T 센서 정지 감지, 손끝 힘 라벨, DG Manager Pose#1~#6, 스톨/의심 메시지에 전류·온도·3 s 추이.

- **메뉴·시작:** 영문 메뉴명, RUN ▶ Robot Monitoring(Live-3D View-Joint Status-Activity Stats), SYSTEM ▶ Startup(자동 연결·첫 화면·상태 표시), 로고 클릭 = 첫 화면, Robot Control 속도바, 2D Pick Correction, 표고 모델 ① 라이브 미리보기.
- **Bin Picking:** 촬영 후 CAM 시점, 인식 ROI, 순위표 X/Y/Z-접근°, 베이스 변환 단일 출처(고정 카메라), 테스트 파지 자세 유지·II/■.
- **재생·Job:** 같은 자세 waypoint 이동·대기 생략, Job Runner 그리퍼 동작 실행, 로봇 미이동 waypoint 촬영 생략(첫 지점 예외), 레시피 조명 미사용 시 조명 접속 생략.

현장 개선 (2026-09-17, Build260917)

실장비(RB5-850 + DG-3F-M + EpicEye Laser) 표고 수확 운전에서 나온 문제를 고치고, 새 현장 설치를 위한 기능을 더했습니다. 세부 변경 이력은 [docs/25_pick_place_v2_changelog.md](#) §1-21 ~ §1-28.

- 📦 ① 레시피 — 어댑터 복사본: 🔒 표시된 📄 표고 수확 (smartfarm.yaml 이 원본, 직접 저장 불가)에서 ✎ 편집을 누르면 편집 가능한 독립 복사본(📄 표고 수확 · 복사본)을 만들어 저장합니다. 어댑터 자체 설정은 📄 표고 수확 검증(smartfarm.yaml)에서 바꿉니다.
- 📦 ③ 🚫 제한(수확 제외 게이트) 패널: 대 길이 $\geq N$ mm(0 = 끄) · 갓 $\varnothing$ 하한~상한 · 지지면 관통 여부를 화면에서 보고 📄 적용으로 현재 레시피(어댑터면 smartfarm.yaml)에 저장. 게이트 메시지에 근거 수치(배지면 기준 갓 높이 - 갓 두께)가 함께 표시됩니다.
- 지지면을 평면으로: 옆을 보는 카메라(배지 벽면)에서 측면 파지가 전부 "지지면 관통"으로 탈락하던 문제 — 지지면을 높이 하나가 아니라 점·법선 평면으로 판정합니다. 레시피 `gripper.params.grip_mode` (비틀어 따기 + 2지면 핀치 우선)로 파지 모드를 지정할 수 있고, ④ 후 보를 고르면 3D 뷰에 ● 파지점과 파지 설명(모드·폭·힘·접근 방향·점수·검증)이 표시됩니다.
- 3D 뷰 공용 툴바(📦 · 🗨️ · Bin Picking · Camera · 템플릿 · Studio 동일): 표시(RGB · 밝기(회색·대비 강조, 기본) · 깊이 컬러맵 · 혼합 · 단색) · 컬러맵 · 점 크기 · 밝기 · 자동 대비 · 가까움=빨강 · 어두운 배경. 설정은 기억되고 재촬영 없이 즉시 반영됩니다. 왼쪽 CAM 버튼 = 카메라 눈 방향(2D 이미지와 같은 상하좌우).
- 같은 기능은 같은 규칙: 표고 프로필 수치(갓 두께·갓 $\varnothing$ ·대 길이·파지점 깊이)는 한 곳 (`smartfarm.yaml harvest` ← 🚫 제한 패널)에서 정해 🗨️ 3D 인식·Bin Picking · 📦 · 📄 수확 검증이 모두 같은 값을 씁니다.
- 📄 관절 풀기(레시피 ✎ 홈/관찰 자세 옆): J4 $\approx$ 188°·J6 $\approx$ 191°처럼 감긴 자세를 같은 TCP 의 덜 감긴 관절 해로 바꿉니다. 실제 이동은 move_l 이라 시작 관절 구성이 사이클 전체의 진입 방향을 정하므로, 진입이 이상하면 홈/관찰 자세부터 확인하세요(▶ 이동으로 실제 확인).
- Robots ▶ 설치 기준(작업 좌표계): 바닥/벽/천장 프리셋 + yaw, 또는 3점 티칭(원점+X·XY면). 다른 장소에 설치한 로봇(RB10-1300 등)에서 '위' 방향·작업 좌표계를 정해 계획·이송 고도·작업영역·적재 높이가 따라갑니다 → Robots.
- Robots/Pendant ▶ 저장 자세 — 누르고 있는 동안 이동(Rainbow-RB 유틸리티 ▶ 자세 대응): 관절 별 순차(J1→J6) 또는 동시 이동, TCP 정기구학 표시, 속도 슬라이더 → Robots. 이 과정에서 RB 재개 명령 오류(`task resume_b` → `resume_a`)를 고쳐 Pendant II/▶·Safety Vision 재개도 정상화되었습니다.
- 🌱 템플릿 모델링: 텍스처(RGB) 표시·CAM 시점 유지·정제 KD-tree 가속(수 분 → 수 초)·↻ 되돌리기 (Ctrl+Z)·🖱️ 화면에서 파지점 지정(1점 / 2점 / 표고 갓→대)·파지점 구 마커·번호.

-  ④ 3D 인식 · 최신기술 비교 화면과 **현장 강건화**(손가락 자리 간섭·다중 프레임/홀 채움·대안 관찰 자세·재시도 사다리)는 위 항목 참조. PICK & PLACE 메뉴는 3D CAM 절차 순서로 정렬되고 각 항목 툴팁에 담당 단계가 표시됩니다.

Object Model (Training) — 물체 모델 학습 (제품·부품·표고 공용)

PICK & PLACE 첫 화면. 딥러닝 인스턴스 분할(YOLO-seg) 모델을 만드는 곳이며, 표고 모델(구 Shiitake Model)도 여기서 합니다.

1. **촬영** — 왼쪽 촬영 목록에서 3D 카메라로 장면을 찍습니다(2D+3D 조직화 스냅샷). 기존 촬영본은 목록에서 고릅니다.
2. **클래스 목록** — 라벨 도구 위 **프리셋 콤보**에서 고릅니다:  표고(harvest,immature,blocked) ·  제품 예시(product / freshener,cap,box) · 기존 데이터셋별 클래스. 직접 적으려면 입력칸에 싹표로 적고 **적용**. 클래스 버튼이 바뀌고 새 데이터셋에 그 클래스가 저장됩니다(모델 names).
3. **라벨** — 도구(원/상자/다각형)로 물체마다 라벨을 그리고 데이터셋(예 방향제_v1)에 저장. 기존 데이터셋을 고르면 그 데이터셋의 클래스가 자동으로 들어옵니다.
4. **학습** — 오른쪽 YOLO-seg 학습 그룹에서 데이터셋을 고르고 학습 시작(GPU 자동). 끝나면 모델 목록에 나타나며 **활성**으로 지정합니다.
5. **레시피에 연결** —  3D Pick&Place ▶ ① 레시피  ▶ ② 인식: 타입 model3d, 학습 모델 콤보에서 모델 선택 후 **모델 지정** →, 인식 프로필 **일반 제품·부품**(표고면 표고).

학습 전에 먼저 시험하려면 레시피 편집기 ② 인식의 **프롬프트(제로샷)** 칸에 영어 물체 이름을 적고  **제로샷으로 먼저 시험** — YOLO-World 로 학습 없이 찾습니다.

3D Pick&Place — 운영 절차 (2026-09-24 갱신)

화면은 좌(①~⑥ 설정·실행) · 중(3D 뷰 /  라이브) · 우(파지 후보 순위 · 사이클 이력 · 로그) 3분할입니다. 3D 뷰 위 **핵심 상태줄**에 FSM·대상·단계·성공/실패/스킵·드라이런이 항상 보입니다.

1. ① **레시피** — + 새로 만들거나  편집. 편집기 하단  **도움말**(그룹별 ? 버튼)에 항목마다 뜻·권장 값·처음 하는 순서가 있습니다. 모르는 파라미터 키(오타)는 저장 시 "혹시 'conf'?" 로 알려줍니다.
2. ② **스캔** — 3D 카메라 선택 ·  촬영 /  PLY / 관찰 자세 ·  **시작 전 확인**(촬영→인식→계획·검증을 한 번에, 로봇 이동 없음) ·  **라이브**(2D 프리뷰, 실행 중에도 계속·SCAN 단계만 잠시 대기) · ROI 인식(프리셋/ 뷰어에서 조정).
3. ③ **인식** —  인식 실행 → 대상 표(#·라벨·치수·점수), 3D 뷰 번호 라벨.  제한(대 길이·갓 지름·관통 여유)은 표고 프로필용.
4. ④ **파지 계획·검증** —  계획(전체): 후보 6DoF 생성 → 지지면·작업영역 → 도달성(IK, 도달 범위 상한 · **관절 급변 한계 90°**) → 충돌·손가락 간섭 → 순위.  보조 보정: 로봇을 실제 대상 위치에 맞춘

뒤 "현재 로봇 위치 = 선택 대상"(표본 평균을 모든 대상에 적용 — 위치/높이 보정).

5. ⑤ **적재 · 픽플랜** — 놓을 곳(격자/고정/포인트) · 픽플랜 JSON 내보내기/불러와 재생.
6. ⑥ **실행** — 모드(단일/연속/계획만) · 드라이런 · ⚠ 단계 확인 대기 · **최대 픽 · 실행 시간(분) · 1개 제한(s) · 재시도(회)**(레시피에 저장) · 🏠 준비위치 · ▶ 시작 / ■ 중지 / || / ~ 스킵 / ● E-STOP / 🔒 E-STOP 해제 · 🔄 **선택 후보 1회 실행**(러너 없이 픽→적재 한 번).
- **실행 중 표시** — 3D 뷰에 실행 중 대상 화살표(주황 → 완료 초록 / 실패 빨강), 우측 사이클 이력·로그. 러너 단계 문구는 `data/3d_pnp_log/pickplace_steps_YYYYMM.csv` 에 자동 기록되고 로그 아래 📄 **진행 CSV** 로 세션분 저장. 인식 결과는 `pickplace_detections_YYYYMM.jsonl`.
- **실패 처리** — 빈손/힘초과/시간초과 → 같은 대상의 다음 후보(파지 실패 → 다음 후보 재시도) → 재스캔 재시도 라운드 → 실패 확정. 1개 동작 제한을 넘기면 안전 중단 후 motion_failed(timeout).
- **권장 순서** — 제로샷으로 인식 확인 → 학습 모델로 교체 → 드라이런 1회 실행 → 실제 1회 실행 → 단일 배치 → 연속.

🍄 Shiitake Harvest (RUN) — 운영 화면 (2026-09-24 갱신)

좌(실행 소스·제어·상태) · 중(최근 스캔 / 📹 라이브 / 📦 3D) · 우(인식 후보 · 시도 이력). 패널 경계를 드래그해 폭을 바꾸고(기억됨), 📊 **집중 보기**로 양쪽 패널을 접어 큰 화면으로 봅니다.

1. **실행 소스** — 독자 모드(카메라 인식만) / SmartFarm 관제. **시나리오** 콤보에서 🎓 Harvest Teaching 에서 저장한 시나리오를 고르면 ▶ 시작 때 그 설정(카메라·ROI·모델·티칭 자세·파지 파라미터·구간 규칙·검증 선택값)이 적용됩니다. **수확 속도** 슬라이더/숫자는 수확 중에도 다음 이동 구간부터 적용.
2. **독자 모드 옵션** — 최대 수확 수 · 빈 스캔 대기 · 대상 없으면 종료 · **재시도(회)**(실패한 표고를 n회까지 재스캔 후 재시도, 다 쓰면 "실패 주변(재시도 제외)") · **실행 시간(분) · 1개 동작 제한(s) · 단계별 확인 대기**(드라이런에서도 진입/파지/비틀기/인출/적재 전에 화면 가운데 위 ⚠ 단계 확인 대기' 바에서 계속·이전·처음·중단 — 무인 연속 수확이면 끄기) · **드라이런**(파지·해제 없이 동작만, 스킵 집계·관제 미보고) · 🖱 **그리퍼 동작...**(단계별 그리퍼 세부 설정 창, 아래 📖 절 참조).
3. **수확 시작 전 확인** — 가운데 📷 **3D 촬영**(📦 3D 를 켜면 자동)으로 점군·ROI 상자를 보고, 🔍 **촬영 +검출**로 러너와 같은 경로(촬영→인식→클래스·도달성 자동 선택)를 로봇 이동 없이 돌려 후보 표(클래스·신뢰도·도달mm·제외 사유)와 2D/3D 표시를 확인합니다. 지금 로봇 자세에서 촬영하므로 관찰 자세에서 누르세요.
4. ▶ **수확 시작** — 관찰 자세 이동 → 스캔 → 수확 가능 클래스(harvest)·도달 가능 후보를 신뢰도 순으로 1개씩 수확 → 재스캔 반복(최대 수확 수/실행 시간까지). 핵심 상태줄·라이브 줄에 진행이 보이고, 1개 수확 뒤 확인 촬영으로 최근 스캔 이미지가 갱신됩니다. || 일시정지 · ~ 현재 대상 스킵 · ● E-STOP.
5. **파라미터 상태줄** — 이미지 아래 파란 줄에 지금 따르는 설정(카메라·모델·ROI·사이클·속도·검증값·도달 한계·클래스·단계 확인)이 보입니다. 값을 바꾸려면 📖 Harvest Lab(파라미터) 또는 🎓 Harvest Teaching(규칙·자세·검증값).

🎓 Harvest Teaching — 티칭 화면 (2026-09-24 갱신)

좌(시나리오 · 카메라/3D · 촬영+검출 · 표시 옵션 · 티칭 자세 패널) · 우(후보 표 · 크기 보정 · 파지 티칭 · 실수확 검증 · 검증 기록). 좌우 경계를 드래그해 폭을 바꿉니다(기억됨).

1. **티칭 자세 패널** — 관찰/바구니/작업 방향 저장(현재 로봇 자세) · 🏠 관찰 자세로 이동 · 🛒 바구니 자세로 이동 · 🌀 작업 방향으로 이동(위치 유지, 자세만) · 🖱️ 프리드라이브(전제 조건 검사·운영자 확인 뒤 ON, 끄면 서보 재활성).
2. **촬영+검출** — 3D 카메라 선택, 인식 ROI(프리셋/3D 뷰에서 조정), 모델·conf. 후보 표에 #·클래스·갓·높이·대mm·위치·도달(✓/× 거리, 툴팁 사유)·티칭 규칙.
3. **파지 티칭** — 후보 선택 → 프리셋·파지 높이 보정·벌림 여유 → "이 구간에 티칭 저장"(구간×클래스 규칙).
4. **실수확 검증** — 단계(포인팅/pre-grasp/파지/전체) · 드라이런(커면 속도 50%) · 단계 확인(🔴 · 🟢) 과 같은 설정, 드라이런 포함 단계마다 확인 창) · 🖱️ **그리퍼 동작...**(단계별 세부 설정 창) · 속도 · 📏 **위치/높이 보정**(dx/dy/dz, "현재 TCP = 파지점": pre-grasp 검증 후 조그해서 측정) · ▶ 실수확 검증(다중 선택 순차). 진행 목록에 "#3 ▶ 검증 시작 ... #3 ✓ 성공"이 쌓이고, 화면의 후보 위에 주황 화살표/단계 문구(완료 초록·실패 빨강, 종료 시 제거). 바구니까지 검증하려면 단계 '전체 수확' + 드라이런.
5. **시나리오** — 📁 저장(검증 선택값·속도 포함) / 📁 적용 / 이름 / 삭제. 저장한 시나리오는 RUN ▶ Shiitake Harvest 에서 그대로 사용.
6. **기록** — 진행 목록은 `data/harvest_log/harvest_teach_steps_YYYYMM.csv`, 후보는 `harvest_teach_detections_YYYYMM.jsonl`, 검증 결과(드라이런 제외)는 `harvest_teach_log_YYYYMM.jsonl`. 📁 진행 CSV 로 세션분 저장.

🔬 Harvest Lab — 파라미터 · 검증 (2026-09-24 갱신)

좌(촬영·앵커·좌표·모션 검증·그리퍼 핸드아이) · 중(검출 오버레이) · 우(후보 · **파라미터**). 경계 드래그로 폭 조절(기억됨).

- **파라미터(harvest 설정 공용)** — 카메라·앵커·모델·운영 사이클 프리셋·소프트 파지·적응형 벌림·단계별 확인·F/T 한계·파지 높이 보정·비틀기·인출·속도 · **도달 한계(mm, 기본 700)** · **관절 급변 한계(°, 90)** · 충돌 감지 적용/켜기/감도/동작/후처리(RB5 set_collision_*). 📁 파라미터 저장 → 🏠 러너 · 🎓 티칭 공용. 🖱️ **그리퍼 동작 세부 설정(단계별)...** 버튼으로 아래 창을 엽니다.
- **그리퍼 hand-eye** — 보드 없이 손끝 3점/공으로 카메라↔로봇 변환. 그리퍼 재부착 후 다시 실행 권장.

🖱️ 그리퍼 동작 세부 설정 (단계별) — 🔬 Lab · 🏠 러너 · 🎓 티칭 공용 창 (2026-09-25)

표고 수확 한 사이클에서 DG-3F-M 손가락(F1·F2·F3)이 단계마다 어떤 자세·힘으로 움직이는지 한 화면에서 보고 바꿉니다. 맨 위 **운영 사이클(⑤ 프리셋)** 콤보가 단계 순서와 값의 출처를 결정합니다(기본

`hv_cycle_human2` 사람 손 2지 수확).

동작 로 그의 단 계	그리퍼가 하는 일	이 창에서 바꾸는 값	저장 위치
pre- grasp 이 동	진입 자세 프리셋(기본 <code>hv_ready2_pinch</code> , 11자·F3 후굴)으로 손가락 이동, 로봇은 파지점 위 진입 높이로	F1·F2·F3 × J1~J4 12축 각도 (현재 그리퍼 자세 가져오기 / 그리퍼를 이 자세로), 📷 진입 높이	프리셋 오버라이드 / smartfarm
진입(하강)	손가락 유지, F/T 영점, 파지점까지 하강	📷 파지 높이 보정	smartfarm
파지	<code>hv_pinch_h_grasp</code> : F1·F2 를 같은 각도로 좁혀 손가락별 목표 접촉력에서 정지, 한계 초과 금지, F3 유지	목표 접촉력(N) · 부서짐 한계(N) · 접근/정밀 스텝(°) · 시간 제한	프리셋 오버라이드
유지 확인	<code>hv_hold_check</code> : 잡은 채 대기 뒤 F/T 재확인(빈손 판정)	유지 시간(s), 📷 F/T 최소	프리셋 / smartfarm
비틀어 따기	<code>hv_twist_pick</code> : 손가락 유지, 로봇이 대 축둘레로 +0.0~−0.0 회전	비틀기 각도(°) · 반복 · 속도(%) · 힘 한계(N) · 안정 시간	프리셋 오버라이드
인출 · 후 퇴	접근축 반대로 당기고 pre-grasp 로 복귀	📷 인출 거리 · 인출 중 유지 최소 힘 · 힘초과 한계 · 이동 속도	smartfarm
바구니 적재	<code>hv_release</code> / <code>hv_basket_release</code> : 해제(대체 포즈)	대체 포즈 번호	프리셋 오버라이드

- ★오버라이드 표시가 있는 단계는 현장에서 바꾼 값이 내장값을 덮고 있다는 뜻입니다. **내장값으로 복원**으로 되돌립니다. 파일: `data/harvest_grip_presets.json`.
- 우선순위: 🎓 티칭 규칙/시나리오 검증값(파지 프리셋·파지 높이·벌림 여유) > 이 창의 사이클 프리셋 값 > 📖 Lab 의 소프트 파지 힘·비틀기(°) 수치(사이클을 쓰는 동안은 무시됨).
- 비틀기 각도 자동 보정: 프리셋 각도 × clamp(50 / 갓 ømm, 0.6~1.5). 예: 20° 설정, 갓 ø83 mm 이상 → 12°, ø33 mm 이하 → 30°. 이 배율은 고정입니다.
- 드라이런에서는 그리퍼 명령을 보내지 않으므로 pre-grasp 손가락 자세·파지·유지 확인 줄이 동작 로그에 나오지 않습니다(로봇 이동만).
- 그리퍼가 분리된 상태에서는 **현재 그리퍼 자세 가져오기·그리퍼를 이 자세로**가 실패 문구를 보이고 값은 그대로 둡니다. 재부착 후 실검증하세요.

🔧 Tool / Vision · 🤖 Robot Control (2026-09-24 갱신)

- Tool / Vision ▶ 📷 로봇 TCP 오프셋 · 페이로드 — 체크한 항목만 robots.yaml 에 저장되고 재생 준비 때 로봇에 적용. **페이로드** = 플랜지에 달린 전부(그리퍼+브라켓+카메라)의 무게(kg)와 플랜지 좌

표 무게중심(mm). **TCP 보내기는 끈 채로** 두세요(핸드아이·티칭 자세가 컨트롤러 TCP 기준). 좌측 열은 세로 스크롤.

- **Robot Control ▶ Overview — Controller SW** 행(제어박스 SW 버전·박스 코드), 속도 바  저장 (robots.yaml), Teach: Freedrive(전제 조건·확인 뒤 ON, OFF 시 서보 재활성).
- **Startup** — 그리퍼 자동 연결은 SYSTEM ▶ Startup 체크 하나로만 결정(다른 자동 연결 경로 제거).

Alarms & Events · Robots & Drivers · 역할 (v2.1, 2026-09-28)

세션 상태와 버튼 활성화

상태	뜻	가능한 동작
미연결 / 연결 중	컨트롤러 연결 전	연결
모니터링(Connected)	상태만 수신	제어권 획득, 비상정지
제어권 보유 (Controlled)	Lock 보유, 서보 OFF	서보 ON(Activate), TCP·페이로드·충돌 설정, DO
대기 · 서보 ON(Ready)	모션 가능	Jog · Move · 저장 자세 · 재생 · 프로그램 실행 · 프리드라이브
실행 중 / 일시정지	재생·프로그램 진행	일시정지/재개/정지, 속도 조절, 비상정지
Fault	비상정지·충돌·통신 끊김·제어권 상실	알람 확인 → 리셋(모니터링으로 복귀)

Alarms & Events 사용

1. 상단 KPI(미확인 Critical/Error/Warning, Fault 로봇)와 Fault 배너를 확인합니다. 배너의  **로봇 리셋**은 E-STOP 래치를 풀고 상태형 알람을 정리합니다.
2. 표에서 로봇·레벨·미확인 필터로 좁히고, 처리한 알람은 ✓ **선택 확인** 또는 ✓ **전체 확인**. 확인자(역할)와 시각이 기록됩니다.
3. **감사 로그(명령)** 탭에는 비상정지·Fault 리셋·로봇 등록/삭제·연결 테스트 등 명령이 역할·로봇·파라미터·결과와 함께 남습니다(삭제 불가).
4. 알람 레벨을 바꾸려면 `config/alarm_map/rainbow.yaml` 의 `codes` (정확한 코드) 또는 `patterns` (부분 일치)를 편집합니다.

Robots & Drivers 사용 (관리자)

1. 역할을  Admin 으로 전환(PIN) → SYSTEM ▶ Robots & Drivers.

2. + **새 로봇**: 벤더 → 모델을 고르면 제어/상태/Modbus 포트·DOF·관절 한계·속도 한계가 채워집니다. ID·이름·IP 를 넣고 필요하면 TCP 오프셋(x,y,z,rx,ry,rz), 페이로드(kg,cx,cy,cz), 설치 기준, vendor_params(JSON)를 적고 **저장**(config/robots.yaml).
3. 표에서 로봇을 선택하고 **연결 테스트 (5단계)**: ① Ping ② 제어 포트 ③ 상태 포트 ④ 컨트롤러 펌웨어(Rainbow: get_control_box_info) ⑤ 드라이버·Capability. 결과와 요약이 아래에 표시되고 감사 로그에 남습니다. 드라이버가 없는 벤더는 ⑤에서 "미구현"으로 표시됩니다.
4. **드라이버 Capability** 표: 벤더별 예상(● 지원 · △ 검증 필요 · × 미지원)과 선택 로봇 드라이버의 실제 지원을 비교합니다. 미지원 기능의 버튼은 화면에서 비활성화되고 사유가 툴팁에 나옵니다.

역할 전환 — 상단 바 오른쪽 역할 버튼 ▼: 👤 Operator(확인만) · ⚙️ Engineer(엔지니어 PIN) · 🛡️ Admin(관리자 PIN, 기본 9999 — 설치 후 `app.yaml ui.admin_pin` 을 바꾸세요). 운영자는 RUN 그룹만 보이고 Robot Control 속도 바가 50% 로 제한됩니다.

📁 기록 파일 (2026-09-24)

폴더	파일(월별 · 20 MB 분할 · 자동 삭제 없음)	내용
<code>data/harvest_log/</code>	harvest_runner_YYYYMM.jsonl	🔧 러너 수확 시도 결과 전부(성공/실패/스킵, 관제 보고와 동일 + mode/robot/cls/stats/dry)
	harvest_teach_log_YYYYMM.jsonl	실수확 검증 결과(드라이런 제외) + 러너 결과(source=runner)
	harvest_teach_detections_YYYYMM.jsonl	촬영+검출 1회 = 1줄 후보 전부(좌표·도달성·규칙), source=teach/runner/precheck
	harvest_teach_steps_YYYYMM.csv	시간대별 동작 로그(🏠 검증 · 🔧 러너, 드라이런 포함)
	harvest_grip_verify_log_YYYYMM.csv	Gripper ⑤ 파지 프리셋 검증 로그
<code>data/3d_pnp_log/</code>	pickplace_steps_YYYYMM.csv	🏠 러너 동작 로그
	pickplace_detections_YYYYMM.jsonl	🏠 인식 결과(러너 스캔·시작 전 확인)

예전 위치(data/ 바로 아래)의 파일은 첫 실행 때 `_legacy` 를 붙여 옮겨집니다. 오래된 달 파일은 지워도 프로그램에 영향이 없습니다.

v1.3 업데이트 (2026-07-16)

이번 버전은 **DG-3F** 그리퍼 티칭, **3D 뷰 회전 개선**, **SmartFarm 수확 연동**이 핵심입니다.

Teaching → Gripper 탭 — DG-3F 그리퍼 티칭 v1.3

- **공식 SDK 연동** — Tesollo DGSDK 기반 위치제어·힘 파지·수동 티칭. 장비 모델(DG-3F-B/M)은 자동 감지되며, 연결이 끊겨도 수 초 내 자동 재연결됩니다.
- **수동 티칭** — 티칭 모드를 켜면 손으로 손가락을 직접 움직여 포즈를 잡고,  **현재 포즈 저장**으로 이름을 붙여 보관합니다.
- **조그 티칭** — 3핑거 × 4관절(12축) 슬라이더로 미세 조정(관절 한계 반영), 라이브 이동 옵션.
- **포즈 목록** — 프리셋(열기/준비/핀치/파지/삼각)과 사용자 티칭 포즈를 더블클릭/버튼으로 적용, 삭제 관리.
- **파지 힘 판정 = 두 손가락 평균 · 미세 정착** 09-22 — 11자 수평 파지와 2지 동기 파지(운영 수확 · ⑤ 프리셋·Teaching 모두)가 두 손가락이 모두 닿았을 때는 **평균 힘**으로 목표 도달을 판정하고(한쪽만 닿으면 큰 쪽, 부서짐 한계는 항상 큰 쪽), 잠근 뒤 **미세 정착**으로 목표 ± 0.1 N 안에 들어올 때까지 0.1° 이하 스텝으로 한 방향만 조정합니다(떨림 방지: 방향이 바뀌면 그 값으로 수용, 최대 12회). ⑤ hv_pinch_h_grasp 파라미터: 판정=평균, 정착 허용(N), 정착 스텝($^\circ$), 정착 최대(회). 결과 문구에 "판정 x.xx N" 이 표시됩니다.
- **F/T·관절 실시간 표시** 09-22 — 그리퍼 데이터를 DG Manager 와 같이 SDK 수신 콜백으로 받아 (폴링 폴백 포함) 20 Hz 로 갱신합니다. Gripper 3D 뷰의 힘 라벨/표는 50 ms, 화면 표는 100 ms 마다 갱신됩니다(예전: 5 Hz 폴 + 100~250 ms 타이머로 지연). 로그에 10 초마다 수신 방식(콜백/폴링)·데이터 나이가 남습니다.
- **DG Manager Pose#1~#6** 09-21 — TESOLLO DG Manager 가 PC 에 저장한 레시피 파일 (`C:\WTESOLLO\DGManager\resources\config\DG-3F-MW\Operator\GripperRecipeData.json`)의 Pose#1~#6 관절값이 목록에 **DG Manager Pose#n** 으로 표시됩니다(시작 자세 표시, 툴팁에 J1~J12·모션시간). 적용 = 그 관절값으로 이동, **↓ DGM→저장** 으로 앱 포즈 `dgm_pose_n` 에 복사(레시피/티칭에서 이름으로 사용). 편집은 DG Manager 에서 하며 앱은 읽기 전용, 파일 위치는 app.yaml `gripper.dg_manager_dir` 로 변경.
- **힘 제어 파지** — 파지 모드(3지/2지/평행/감싸기)와 파지력(N)을 정해  파지 /  해제. 3D 뷰 손가락도 실시간 연동.
- **Waypoint 그리퍼 동작** — Teaching 궤적의 각 지점에 파지/해제/포즈를 지정해 **Trajectory에 저장·재생**합니다. Waypoint 편집기 **Grip 열**, Pendant 수집 중 지정, ▶ **Go to row** 단일 이동 시에도 도달 후 실행.
- **실시간 관절 표시·고급(SDK)** — 12축 각도 실시간 표시(티칭 중 강조), 모션시간·게인·PID/PD·토크 리미트·로우패스 필터·GPIO·F/T 영점 등 SDK 기능 직접 조작.
- **시작 자세 저장·자동 적용** — Tool/Vision에서 프리셋 적용/Ready 시 **시작 자세로 저장**되고, 프로그램 시작 시 그리퍼가 자동 연결되어 그 자세로 이동합니다. Tool/Vision 3D 미리보기는 로봇 팔의 **실시간 자세**를 반영합니다.

주의 — 그리퍼는 **단일 연결 장치**입니다. DG Manager(벤더 프로그램)와 본 앱을 동시에 연결하지 마세요. 연결 경쟁이 반복되면 그리퍼 통신이 멈출 수 있으며, 이때는 그리퍼 전원을 재인가해야 합니다.

3D 뷰 — 턴테이블 회전

- 모든 3D 화면에서 **왼쪽 드래그 좌우 = 로봇 주위 360° 궤도 회전**(바닥 수평 유지), **상하 = 올려보기/내려보기**. ISO 각도를 유지한 채 빙 둘러보기가 쉬워졌습니다. 휠 줌·Shift+드래그 이동은 동일합니다.

SmartFarm 수확 연동

- SmartFarm 서버로 수확 세션/시도/로봇 상태를 전송(오프라인 시 아웃박스 재전송)하고, Tesollo 12축 그리퍼 텔레메트리를 표시합니다.

v1.2 업데이트 (2026-07-07)

이번 버전은 **티칭 자동 정렬·캘리브레이션, 곡면 제품 합성(스티치), Job Runner 운영, 리포트/SPC, Operator** 편의가 크게 보강되었습니다.

Teaching — 체커보드 평면 정렬 · 오토포커스 · 자동 캘리브레이션

-  **평면 정렬(무보정)** — 제품 위 체커보드를 카메라가 **정면(수직)**으로 바라보도록 로봇이 스스로 기울기·방향·거리를 맞춥니다. 로봇을 살짝 움직여보며 **로봇↔영상 관계(야코비안)**를 학습하므로 hand-eye/내부파라미터 오차에 영향받지 않습니다. 카메라 장착 방향이 회전돼 있어도 **3축 회전 프로브**로 올바른 축을 자동 발견하고, **3축 병진 재중심**으로 회전 중 보드가 화면을 벗어나지 않게 유지합니다.
- Live 기하 오버레이** — 정렬 과정이 실시간으로 표시됩니다(노란=실제 외곽 / 시안=이상 정면 외곽 / 좌상단 수평계 / 왜곡·방향·칸크기 수치). 인라인 뷰와 **확대 창** 모두 표시되며 해상도에 맞춰 글자가 커집니다.
- 초점(작업거리)** — 광축을 **자동 검출**해 그 축으로 이동하며, 내부파라미터+목표거리가 있으면 체커보드 겹보기 크기로 **정확한 작업거리(mm)에 착지**, 없으면 선명도로 최적 초점을 찾습니다. 기본 칸 크기 24mm, 목표 작업거리 500mm.
- 캘리브레이션 자동 촬영** — Intrinsic-Hand-Eye 캘리브레이션에서 **9~15자세를 로봇이 자동으로 이동·촬영**합니다(검출 성공 자세만 누적, 완료 후 원위치 복귀).
- 실시간 품질 지표** — Camera-Teaching의 Live/Snapshot에 선명도·밝기·대비·과노출·노이즈·SNR 등을 실시간 표시.

합성(Stitch) — 곡면 언랩 · 정밀 이음 정합

- **곡면 언랩(기본 방식)** — 곡핏처럼 완만한 곡면 제품을 **로봇 포즈로** 표면을 복원해 **호길이대로 펼침**. 평면 투영의 원근 왜곡을 제거하고 배율을 균일화합니다. 갤러리와 촬영 설정 화면 "**합성 방식**"에서 선택하며(기본값 곡면 언랩, 설정 유지), hand-eye+ 내부파라미터가 있고 **조밀 촬영(겹침 클수록)**일 수록 정확합니다.
- **정밀 이음 정합(옵션)** — 곡면에서 프레임별 배율차로 접합면이 벌어질 때 겹침을 측정해 보정합니다. 보정이 실제로 정합도를 높일 때만 적용(**절대 악화 없음**).
- **hand-eye 대소문자 무관** — 로봇 ID를 rb1/RB1 등 어떻게 표기해도 동일한 캘리브레이션을 사용하여, 궤적 합성이 축이 틀어지거나 배율이 어긋나던 문제를 해결했습니다.

Job Runner — 레시피 확장 · 실행 모드 · 결과 표시

- **레시피 설정 확대** — Hole Inspection 화면과 동일한 **분석 상세 파라미터**(Hough 민감도, 직경/위치 허용오차, 기대 개수, 최대 결함수, 누락 시 NG, CAD 도면/위치허용)와 **로봇 촬영옵션**(Stop&Shoot/Fly-by, settle, 촬영주기, N번째 waypoint, 모든 waypoint), 그리고 **카메라 설정**(노출·게인·현재값 읽기)·**조명 설정**(채널별 capture 강도)을 레시피에 저장합니다.
- **실행 모드** — 반복횟수 / 시간(분) / 무한 중 선택. 무한 실행 시 로그·화면이 상한 처리되어 장시간 운영에도 가벼움을 유지합니다.
- **결과 표시** — 프레임별 **Hole 검출 오버레이 이미지**와 실시간 **Stitched(전체 패널)**를 탭으로 전환해 봅니다(사이클 종료 시 Hole Inspection과 동일한 정확 합성으로 저장: 세션 폴더 `stitched_holes.png`). 버튼 아래 **큰 판정 배너**(상태: 검사 대기/중/완료)와 좌측 **사각 램프**(결과: **OK 녹색** / **NG 빨강**)로 구분 표시합니다.

Reports / SPC

- **상세 검사 데이터 탭** — 사이클별(시각·판정·시리얼·택트·프레임·NG·사유·측정치) 테이블과 **OK/NG·불량률·평균 택트 요약 통계**를 함께 표시.
- **Cp/Cpk 자동 규격한계** — 측정치를 고르면 레시피 공차에서 **LSL/USL**을 자동 채워 Cp/Cpk를 바로 계산합니다(예: 홀 직경 = 기대 $\pm$ 직경공차). 직접 입력도 가능합니다.
- **자동 화면 갱신** — 끄/2/5/10/30/60초 주기로 관리도·상세 데이터를 자동 새로고침(설정 유지).

Operator 편의

- **왼쪽 메뉴 Hide/View** — Operator 모드에서 상단 **≡ 버튼** 또는 **F9 / Ctrl+M**으로 좌측 메뉴를 접었다 펼 수 있습니다(키오스크형 전체 결과 화면). 메뉴가 숨겨진 동안에는 복원용 **≡**가 항상 보입니다.

저장 위치 — 판정 결과 DB `dataWresults.db`, 캡처/오버레이 `dataWcapturesWcapture_<궤적>_<시간>_rNNW` (원본 `frame_XXXX.png` + 홀 오버레이 `frame_XXXX_holes.png` + `stitched_holes.png`), 레시피 `dataWrecipesW<이름>.yaml`.

화면 구성

- 좌측 사이드바 — 페이지 이동 메뉴. 맨 아래 버전 정보 및  **User Manual** 링크.
- 중앙 — 선택한 페이지.
- 창 크기/위치 — 종료 시 자동 저장되어 다음 실행에 복원됩니다. 처음 실행은 FHD 기준 크기로 열립니다.

WORKSPACE

▣ Fleet Dashboard

○ Robots

⌘ Telemetry · 100 Hz

AUTOMATION

✦ Teaching

⦿ Camera · 2D / 3D

◎ Trajectory Capture

VISION

⌘ fineAi PA Engine

▶ 3D CAM - Jobs

QUALITY

▣ 품질 대시보드

✓ Capture Quality

↻ Object Detection

Startup (SYSTEM)

프로그램을 켤 때 자동으로 할 일과 첫 화면을 정합니다(`config/app.yaml` `ui.default_view / ui.default_tab / startup`). 저장 후 다음 시작부터 적용됩니다.

- **시작 배너·창 포커스** (09-22) — 시작 절차 문구가 상단 **파란 배너**에 시각과 함께 누적 표시되고(마지막 문구 뒤 12초 후 자동 숨김, 클릭하면 바로 닫힘), 프로그램 창이 시작 직후 앞으로 올라와 포커스를 받습니다(Windows 포그라운드 제한 우회 포함).
- **시작 절차 상태 표시** (09-21) — 하단 상태창과 운영 로그(Startup)에 단계별 문구가 뜹니다: "🚀 지정된 첫 화면 'Robot Monitoring ▶ 3D View' 으로 이동합니다" → "👉 로봇 'RB1' 을(를) 연결합니다 — 192.168.1.7" → "✅ 로봇 'RB1' 연결 완료" → "👉 그리퍼를 연결합니다" → "✅ 그리퍼 연결 완료 · 시작 절차 완료". 실패하면 ⚠ 문구와 수동 조치 안내가 표시됩니다.
- **연결 후 컨트롤 Lock 자동 획득** (09-22) — 커면 자동 연결 직후 Lock 을 잡아 Job Runner·Teaching 을 바로 실행할 수 있습니다. 꺼져 있어도 Job Runner ▶ 시작 때 Lock 이 없으면 "지금 Acquire 하시겠습니까?" 로 물어보고 획득합니다(예전에는 "시작 실패: Lock 미보유" 로 끝나 Robot Control 에서 잡아야 했음).
- **시작 시 로봇 자동 연결** — 연결할 로봇(등록 목록)과 시뮬레이션 여부. 창이 뜬 뒤 백그라운드에서 연결하고 성공하면 활성 로봇으로 설정합니다(실패해도 앱은 계속, 로그 `[Startup]`).
- **시작 시 그리퍼 자동 연결** — `smartfarm.yaml` `gripper.enabled` 가 켜져 있어야 하며 연결 후 시작 자세로 이동합니다.
- **첫 화면(메뉴)** — 모든 메뉴 항목 중 선택. RUN ▶ Robot Monitoring 이면 **첫 탭**(Live · 3D View · Joint Status · Activity Stats)도 고를 수 있습니다. 운영자 모드에서는 항상 Job Runner 로 시작합니다.
- **왼쪽 상단 로고 클릭** — 언제든지 설정된 첫 화면(+탭)으로 돌아갑니다.

Robot Control (구 Robots)

Speed bar (09-21) — Overview ▶ Robot control 아래 **Speed bar %**(스핀 + 슬라이더 연동): 컨트롤러 전역 속도. 각 동작의 속도(레시피 `speed_pct`, 티칭 %)는 이 바에 곱해지므로 100% 가 기준값(레시피 20% → 50 mm/s 그대로)이고, 50% 로 두면 모든 동작이 절반 속도가 됩니다. **적용**은 연결된 로봇에 즉시, **저장**은 `robots.yaml(speed_ratio)`에 기록되어 재생 준비(E-STOP 해제·러너 시작·1회 실행)마다 다시 적용됩니다. Robot Monitoring ▶ 3D View 의 State ▶ Speed 가 이 값입니다.

메뉴명 영문 통일 (09-21) — 메뉴 라벨은 영문으로 통일했습니다(향후 Language 선택 기능에서 한글 표시 예정). 대응: Shiitake Harvest=표고 수확 · 2D Pick Correction=2D 픽 보정(구 Pick Guidance: 2D 골든 샷 기준 픽 위치 보정) · Robot Control=로봇 제어 · Gripper (DG-3F-M)=그리퍼 · Template Modeling=템플릿 모델링 · Shiitake Model (Training)=표고 모델(학습) · 3D Recognition · Comparison=3D 인식·최신기술 비교 · Harvest Lab=표고 수확 검증 · Harvest Teaching=표고 수확 티칭 · 3D Pick&Place=3D Pick&Place 통합 · PA Engine Jobs (Legacy)=PA Engine 잡(레거시) · SmartFarm Link=SmartFarm 연동. Robots 의 **Info(Joint Status)·Activity Stats** 탭은 RUN ▶ Telemetry 탭으로 옮겼고, 여기에는 Overview(연결·Lock·모드·전원·E-STOP·설치 기준·저장 자세)와 3D Model 만 남았습니다.

Controller SW · TCP/페이로드 · Freedrive (09-24) — Overview 의 **Controller SW** 행은 연결 직후 읽은 제어박스 소프트웨어 버전/박스 종류입니다. Robot control ▶ **Teach: Freedrive ON** 은 프리드라이브(직접 교시) — 09-24 사고(OFF 순간 전 관절 풀림·전도) 이후 전제 조건 검사(Real·서보·**페이로드 등록**·Lock)와 운영자 확인을 거쳐야 켜지고, 끄는 즉시 서보를 재활성합니다(What's new 09-24 참고). TCP 오프셋·페이로드는 Tool / Vision 화면에서 저장하며 재생 준비 때마다 로봇에 적용됩니다. 충돌 감도는 Harvest Lab 파라미터(레시피)에서 켵니다.

설치 기준 (작업 좌표계) v2.0 — Overview 탭 ▶ Robot control 아래. 로봇을 다른 장소(바닥/벽/천장, 다른 방향)에 설치하면 컨트롤러의 베이스 좌표는 그대로지만 **중력 방향과 작업대 방향**이 달라집니다. 여기서 정한 기준을 📦 3D Pick&Place 의 파지 계획(위/옆/아래 접근 분류·수평 방위·지지면)·이송 고도·작업영역·적재 높이가 모두 사용합니다.

1. **프리셋** — 바닥 설치(베이스 Z 가 위) · 벽 설치(베이스 +X 가 위, Z 는 벽에서 수평으로 나옴) · 천장 설치(베이스 Z 가 아래) + **yaw**(위 축 둘레 회전, 현장 방향 정렬). 직접 입력(rpy)도 가능.
2. **3점 티칭**(권장, 어떤 설치든 동일) — 로봇을 연결한 뒤 TCP 를 ① **원점** → ② **+X 방향 점(≥50 mm)** → ③ **작업면 위 +Y 쪽 점**에 차례로 대고 각각 캡처 → **계산**. 세 점이 만드는 면의 법선이 '위'가 되고, 원점·X 방향이 작업 좌표계가 됩니다(RB10-1300 을 새 장소에 설치했을 때 이 세 점만 찍으면 됩니다).
3. **📁 저장** — `config/robots.yaml` 의 해당 로봇 `mount` 에 기록. 상태 라벨에 up(베이스) 벡터가 표시됩니다. 저장 즉시 새 계획부터 반영됩니다.

주의: 홈/관찰/바구니 **관절 자세**는 로봇 모델·설치가 바뀌면 다시 티칭해야 합니다(레시피 ↘ ▶ 이동으로 확인). 작업영역(레시피 제한)은 기준이 바닥·yaw 0·원점 0 이 아니면 작업 좌표계에서 판정합니다.

저장 자세 — 누르고 있는 동안 설정된 자세로 이동 v2.0 — Overview 탭 ▶ Robot control 아래(Teaching ▶ Pendant 에도 같은 패널). Rainbow-RB 유틸리티 ▶ 자세와 같은 사용법입니다: 저장 번호(기본 **Pose-0 = 전 관절 0°**)를 고르고 Base/Shoulder/Elbow/Wrist1~3 값을 넣거나 **현재 자세 정보 가져오기**로 채운 뒤 **설정 저장**. **↓ 설정된 자세로 이동** 버튼을 **누르고 있는 동안** 로봇이 그 자세로 부드럽게 이동하고, **손을 떼면 정지**해 그 자리에 멈춥니다 (감속 정지가 듣지 않는 펌웨어면 0.7초 뒤 자동으로 확실한 정지를 보냅니다). 다시 누르면 현재 위치에서 다시 이동하며 도달하면 자동으로 완료됩니다. 첫 누름에서는 재생 준비(서보 활성화·속도바, 약 2초)를 자동으로 한 번 합니다. **이동 방식**은 기본 **관절별 순차(J1→J6)** (Rainbow-RB 와 같이 한 관절씩 차례로) 또는 동시 이동(move_j 한 번)이며, 설정 자세와 현재 자세의 **TCP(X,Y,Z,Rx,Ry,Rz)**가 정기구학으로 계산되어 함께 표시됩니다. 속도는 슬라이더/입력(%)으로 정하고, 연결·제어권(lock)·E-STOP 해제·**Real Mode** 상태여야 실제 로봇이 움직입니다(Sim Mode면 3D만 움직이며 상태 라벨에 표시). 1.5초 동안 움직임이 없으면 상태 라벨이 원인 점검 항목을 안내합니다. 저장 위치 `config/saved_poses.yaml` (로봇별).

로봇 컨트롤러를 등록·연결하고 상태(연결/락/모드)를 확인합니다. 다른 페이지의 모션·텔레메트리는 여기서 연결한 활성 로봇을 사용합니다.

상태 색상 표시 — Overview 상단의 **연결 상태 배지**(CONNECTED 초록 / DISCONNECTED 회색 / SIMULATION 하늘 / ERROR 빨강)와 Connect(초록)·Disconnect(빨강) 버튼 색으로 현재 연결을 한눈에 봅니다. **Robot control**에서는 텔레메트리로 **현재 Mode**(활성 모드 버튼 채움: Real 파랑 / Sim 하늘)와 **Power/안전 상태**(정상=전원ON 초록, E-STOP=빨강, 충돌=주황)를 버튼·「현재 상태」알약으로 표시합니다. (창 제목표시줄에 버전 `fineAi RoboVision v1.2` 표기.)

Teaching

궤적(Trajectory)을 만들고 편집합니다. 세 가지 소스 탭을 제공하며, 현장 조작 빈도가 가장 높은 **Pendant**가 기본(첫) 탭입니다.

- **Pendant (수동 조작)** — 소프트웨어 펜던트로 jog 하며 waypoint 수집(기본 탭). Bookmarks 아래 **저장 자세** 패널(Robots 와 동일): **↓** 버튼을 누르고 있는 동안 저장 자세로 이동, 떼면 정지.
- **Record (주행 기록)** — 로봇을 수동으로 움직이며 자세를 기록.
- **Generate (패턴)** — Line/Raster 등 파라메트릭 궤적 생성.
- **Gripper (그리퍼)** v1.3 — DG-3F 그리퍼 포즈 티칭·파지 제어(아래 참조).

현장(터치스크린·장갑) 조작을 위해 Jog 키패드와 Capture·Play·Record 등 핵심 버튼을 터치하기 좋은 크기로 키웠습니다. ■ **E-STOP**은 접이식 영역 밖 좌측 하단에 **항상 노출**되어 어떤 상태에서도 즉시 누를 수 있

습니다. 속도/가속 등 **Motion 파라미터**는 접이식(기본 접힘)으로 정리되어, 펼쳐서 조정합니다(Play-Lock은 항상 보임).

카메라 Live

우측에 **Camera** 토글 바와 **Camera (live)** 뷰가 있어, 티칭 중 카메라 영상을 보며 작업할 수 있습니다.

 버튼으로 라이브 화면을 별도 창(비모달)으로 크게 띄워 펜던트 조작과 동시에 볼 수 있습니다.

Trajectory 목록 (복사)

-  **복사** — 좌측 Trajectories 목록에서 선택한 궤적을 **waypoint까지 통째로 복제**합니다(새 이름 입력 → 새 ID로 저장, 복사본 자동 선택). 원본과 완전히 독립이라 복사본을 수정해도 원본은 그대로입니다.

Waypoint 목록 편집

- **On** 체크 — 해제 시 재생/캡처에서 그 waypoint를 건너뛵니다.
- **Cap** 체크 — 해제 시 Trajectory Capture에서 그 지점은 **항상 촬영 제외**(어떤 옵션으로도 덮어쓰지 않음).
- **+** **현재 위치 삽입** — 로봇의 **현재 자세(TCP+관절)**를 선택한 waypoint **바로 다음**에 새 waypoint로 삽입합니다(선택이 없으면 맨 끝에 추가). 인덱스 자동 재정렬·새 행 선택, 저장 전까지 편집/취소가 가능. 로봇이 연결되어 현재 위치가 들어와야 활성화됩니다.
- **▲ Up / ▼ Down** — 선택한 waypoint(들)를 한 칸 위/아래로 **순서 이동**(# 인덱스 자동 재부여).
- **Duplicate / Delete** — 선택 행 복제(맨 끝 추가) / 삭제. 저장 전까지 Discard로 복원 가능.
- **Grip** 열  — 로봇이 그 waypoint에 **도달한 뒤** 실행할 그리퍼 동작을 지정합니다: —(없음) /  **파지**(힘 제어) /  **해제** / **포즈 ...**(열기·준비·파지 등 프리셋과 사용자가 티칭한 포즈). 콤보에서 바꾸거나 —로 되돌리면 되고(추가/변경/삭제),  **Save** 하면 Trajectory에 함께 저장됩니다. 점대점 재생(Pendant/Generate 궤적)과 Sim 재생에서 실행되며, 블렌드(수동 dense 기록) 재생에서는 실행되지 않습니다.

새 궤적 수집 중(저장 전)에도 지정할 수 있습니다 — Pendant 탭의 waypoint 수집 박스에서 **그리퍼** 콤보로 동작을 고른 뒤  **Capture pose** 하면 그 지점에 함께 기록되고(캡처 후 콤보는 —로 복귀), 이미 캡처한 지점은 목록에서 선택 → **선택 wp에 적용**으로 변경/삭제합니다. 목록에  /  /  표시가 붙습니다.

▶ **Go to row**로 단일 waypoint에 이동할 때도, 그 행에 그리퍼 동작이 있으면 **도달 후 자동 실행**됩니다.

재생 (Play)

- **단일 waypoint 이동**(행 더블클릭 / Go to)과 **전체 Play** 모두 **관절 기준**으로 교시 자세를 정확히 재현합니다(같은 자세에 도달). 수동 기록(관절 dense)은 관절 블렌드(move_jb), 그 외(Generate/Pendant 등)는 **Trajectory Capture와 동일한 점대점(move_l/move_j)**으로 재생됩니다.

- 이동 완료 판정은 시간이 아니라 실제 모션 상태(정지 감지) 기준입니다 — 첫 지점이 멀거나 속도가 느려도 로봇이 실제로 멈출 때까지 기다린 뒤 다음 점으로 갑니다(고정 타임아웃으로 인한 “첫 지점 건너뛸” 없음).
- **Move Speed** 가 점대점 이동에, **Blend Speed** 가 관절 블렌드(수동 기록)에 적용됩니다.
- **특이점 보호정지 자동회피(속도 자동제한)** — 긴 직선이동(move)에서 양끝 관절각 차이가 크면(특이점 근처) TCP 속도가 조금만 높아도 순간 관절속도가 한계를 넘어 보호정지(램프 깜빡임)가 납니다. 이동 방식은 **직선(move)** 그대로 유지하고, 구간별로 평균 관절속도가 한계 이하가 되도록 **TCP 속도/가속을 자동으로 제한**합니다(관절이동이 작은 구간은 슬라이더 속도 그대로). 한계는 `app.yaml` 의 `robot.max_joint_speed_dps` (기본 8°/s)로 조정 — 멈추면 낮추고, 여유 있으면 올립니다.

그리퍼 티칭 — Gripper 탭 v1.3

DG-3F 3핑거 그리퍼(12축)의 포즈를 티칭하고 파지를 제어합니다. 상단 🔌 **연결**을 누르면 그리퍼에 연결되고(모델 자동 감지), 연결이 끊겨도 수 초 내 자동 재연결됩니다.

- 🖐️ **수동 티칭** — **티칭 모드 시작**을 누르면 모터가 풀려 손으로 손가락을 직접 움직일 수 있습니다. 원하는 포즈를 잡은 뒤 📌 **현재 포즈 저장**으로 이름을 붙여 저장하고, **티칭 모드 종료**로 제어에 복귀합니다. 티칭 중 잡은 각도는 조그 슬라이더에 실시간으로 표시됩니다.
- **포즈 목록** — 내장 프리셋(open/ready/pinch/close/tripod 등)과 사용자 티칭 포즈를 한 목록에서 ▶ **적용(이동)**(또는 더블클릭)📄 **삭제**로 관리합니다. 포즈는 `data/gripper_poses.json` 에 보관됩니다.
- **조그** — 핑거별(F1~F3) 관절 J1~J4 슬라이더로 각도를 지정하고 ▶ **슬라이더로 이동**. **라이브 이동**을 켜면 슬라이더 조작이 즉시 반영됩니다. **현재각** → **슬라이더**로 실제 자세를 불러옵니다.
- **파지 (힘 제어)** — 파지 모드(3지/2지 조합/평행/감싸기)와 파지력(N)을 정한 뒤 🖐️ **파지**. 물체에 힘이 걸리면 자동으로 유지되고 🖐️ **해제**로 놓습니다.
- **3D 연동** — 티칭/이동/파지 중 그리퍼 손가락 움직임이 3D 로봇 뷰에 실시간 반영됩니다.
- **실시간 관절 각도** — 12축 각도가 항상 실시간 표시되며, 수동 티칭 중에는 강조색으로 바뀌어 지금 저장될 값을 눈으로 확인할 수 있습니다. **라이브 이동** 조그는 서보 스트리밍으로 즉시 반응합니다.
- **고급 (SDK)** — 접이식 그룹에서 SDK 기능을 직접 조작합니다: **모션시간(ms)**, **게인 P/D/I/I한계**, **PID/PD 제어 전환**, **토크 리밋**, **로우패스 필터**(α — 낮을수록 부드러움), **GPIO 출력 3ch**, **F/T 센서 영점**.

주의 — DG Manager(벤더 프로그램)와 동시에 연결하지 마세요(단일 연결 장치). 연결 설정은 `config/smartfarm.yaml` 의 `gripper` 섹션(driver `dgsdk`, 포트 502)입니다.

Camera (2D/3D)

카메라를 열고 라이브를 보며 스냅샷, 캘리브레이션을 합니다.

카메라 추가 / 선택

- **Device** 콤보에서 등록된 카메라 선택.
- **+ Add 3D (EpicEye)** — IP 입력 또는 네트워크 검색으로 EpicEye 카메라 등록 (`config/cameras.yaml` 에 저장).
- **Open / Close /  Snapshot** — 연결/해제/촬영(스냅샷은 PNG, 3D는 PLY 포인트클라우드도 저장 — 아래 “3D ROI / PLY 저장” 참고).
- **⚙ Settings** — Basler/EpicEye 카메라 파라미터 설정.

화면 구성 (좌: Live, 우: 적응형)

- 좌측 = 라이브 2D (마우스 휠 줌 / 드래그 이동 가능).
- 우측 = **3D 카메라**면 포인트클라우드, **2D 카메라**면 마지막 Snapshot 정지화면.
- **PC색** 콤보 — 포인트클라우드 색상: 의사색채(깊이)/텍스처/회색조.
- **🎯 2D방향** — 포인트클라우드를 2D 영상과 같은 방향으로 정렬.
- **🔄 3D 자동 + 주기(초)** — 커면 주기적으로 풀 3D를 촬영해 포인트클라우드를 갱신.

2D 먼저 표시: 3D 카메라를 열면 라이브 2D가 즉시 보이고, 우측 3D는 백그라운드로 수집되어 **3D 자동 On/OFF와 무관하게 최초 1회는 반드시** 표시됩니다(실패 시 자동 재시도). 수집 중에는  **3D 데이터 수집 중...** 이 표시됩니다.

3D 뷰 조작

- 좌측 버튼 **±X / ±Y / ±Z** — 해당 축 정렬 뷰로 즉시 전환, **ISO** 등각, **FIT** 화면 맞춤.
- 마우스: 왼쪽 드래그 **회전**, 휠 **확대/축소**, 중간버튼 이동(pan).

3D ROI / PLY 저장

- **3D 카메라일 때만** 표시되는 ROI 바: **제품** 콤보 + **3D ROI** 체크 + X/Y/Z min·max 스펠박스 + **클라우드 맞춤** + **✂ 박스 편집(드래그)**.
- **3D 박스 ROI** — X/Y/Z 범위(mm, 카메라 좌표)로 관심영역을 지정. 숫자 입력과 3D 뷰에서 마우스 드래그(box widget) 양방향 동기. **클라우드 맞춤**으로 현재 클라우드 범위에 자동 맞춤.
- **제품별 ROI 프리셋** — 제품 이름으로 ROI를  **저장**/불러오기/🗑 **삭제**. `data/roi_presets.json` 에 영구 저장되어 다음 실행에도 유지.
- **PLY 저장 형식** —  스냅샷의 3D는 **EpicEye 호환 organized 바이너리 PLY**로 저장됩니다(헤더 `format binary_little_endian` + `obj_info ... (Width=...; Height=...)` + `element vertex = W×H`, 전체 그리드 유지). ROI를 켜면 박스 밖 점은 invalid(NaN)로 기록되되 정점 수는 유지됩니다. (앱의 **모든 PLY 출력은 바이너리**이며, PA Engine Load Cloud는 바이너리·organized·NaN을 모두 읽습니다.)

3D 캡처 타이밍 표시

3D 프레임을 받을 때마다 상태줄에 총시간·수집·전송·계산·점수·전송속도가 표시됩니다 (예: 총 1000ms · 수집 200 · 전송 650 · 계산 50ms · 120,000pts · 2.8MB/s). 3D Studio에서도 동일 표시. 연속 수신 실패 시 △ 카메라 3D 응답 없음 - 다른 PC 동시 접근 확인 으로 안내합니다.

캘리브레이션

하단 Calibration 줄의 [Intrinsic...] / [Hand-Eye...] / [2D 평면...] 로 진행하며 결과는 config/hand_eye.yaml 에 저장됩니다.

- **보드 종류 선택** — 체커보드 / 원형 그리드(대칭·비대칭). cols×rows·간격(mm) 입력.
- **ROI 지정** — 체크 후 영상에서 **드래그로 검출 영역**을 지정하면 그 안의 보드/원만 검출합니다(주변 패턴·잡음 무시). **ROI X**로 해제(전체).
- **검출 확인** — 현재 프레임에서 보드 검출을 즉시 시험(성공 시 초록 점 오버레이). Capture는 검출된 프레임만 누적되어 “검출 0” 상황을 방지합니다.
- **카메라값 불러오기** — 카메라가 내부파라미터를 제공하면(EpicEye 등) 보드 촬영 없이 Intrinsic을 바로 저장.

Hand-Eye (3D 6DoF)

- **Mode** — **Eye-in-Hand**(카메라=로봇 장착, 보드 고정) / **Eye-to-Hand**(카메라 고정, 보드를 그리퍼에 부착). 각각 T_tcp_cam / T_base_cam 으로 저장되며 3D CAM 런타임이 모드에 맞게 좌표를 변환합니다.
- **자동 다자세 수집** — 연결·Lock된 로봇을 자동으로 여러 자세(기준 자세 주변 회전·이동)로 이동하며 매 자세에서 자동 촬영(검출된 것만 누적). 수동으로 옮겨가며 Capture할 필요가 없습니다. 끝나면 기준 자세로 복귀.

2D 평면 (평면 위 픽)

부품이 일정 높이의 평면에 있을 때, 보드를 **1회 촬영**하면 픽셀→로봇 XY 변환(호모그래피)을 즉시 산출합니다(다자세 불필요).

- 보드를 작업 평면에 두고 **보드의 로봇 배치**를 입력: 원점 X·Y(마커 #0의 로봇 XY), 축각(보드 가로 방향이 로봇 +X와 이루는 각), 평면 Z(작업면 높이). 현재 **TCP로 원점 채우기** 버튼으로 로봇을 보드 원점에 대고 자동 입력 가능.
- **촬영 & 계산** → RMS(mm) 확인 → **Save**. 결과는 config/hand_eye.yaml 의 **planar** (호모그래피 H + 평면 Z)로 저장됩니다.

EpicEye 3D 카메라

SDK 4.0(Laser V3) 설정이 촬영에 반영되지 않을 때 — Apply 가 "카메라가 설정을 거부했습니다"로 나오면 장비가 직전 설정/프레임을 처리 중인 것이니 잠시 후 다시 Apply 합니다 (앱이 3초간 자동 재시도). 촬영이 계속 실패하거나 같은 프레임만 반복되면 ⚙ Settings 의 **세트 적용**(활성 매개변수 세트 재선택)으로 복구되고, 그래도 안 되면 카메라 웹 UI(<http://<IP>:5000/app>)에서 재부팅합니다. 실장비 검증 결과와 벤더 문의 항목은 [docs/26](#) 참조. **09-24**: "설정이 저장만 되고 촬영에 반영되지 않는다"던 문제는 앱의 촬영 모드 번호 오매핑으로 확인·수정되었습니다(모드 1 = 구조광 섹션). 야간에는 라이브 바 💡 LED ON + 노출 100~150 ms 로 2D 가 밝아집니다.

Transfertech EpicEye(Pixel Pro 등) 3D 카메라를 HTTP API로 연결합니다. IP는 `host` 또는 `host:5000` 형식(포트 생략 시 5000).

구분	파라미터	범위/의미
2D	노출(ExpTime2D), Gain	0.1–1000 ms / 0–24 dB
3D	노출(ExpTime3D), Gain	2.5–200 ms / 0–24 dB
일반	프로젝터 밝기 / 패턴 / 해상도 / LED 보조조명	Low·Med·High / 모드 / 모드 / ON-OFF
후처리	평활화·구멍보완·노이즈필터	각 5단계(끄기~매우강)

라이브 화면에서도 **노출/Gain/💡 LED**를 즉시 조절할 수 있는 컨트롤 바가 제공됩니다. 카메라 촬영은 **보이는 화면만** 요청합니다 — 다른 메뉴로 옮기면 이 페이지의 라이브가 멈추고(카메라 연결은 유지), 돌아오면 자동 재개됩니다. 🏠 · 🎓 · 📦 · 🔍 의 📺 라이브도 같은 규칙입니다(09-26).

3D Studio

Camera 페이지의 📷 **3D Studio** 버튼으로 엽니다. 촬영 후 다양한 방식으로 보고 설정·저장합니다.

표시 (하단 버튼)

- 📦 3D — 포인트 클라우드(자유 회전). 색상: 텍스처/기본/의사색채/회색조 + 의사색채 축(X·Y·Z).
- 🖼 2D 이미지 — 원본 텍스처.
- 🌈 3D 이미지 — 2D와 동일 해상도의 평면 영상을 깊이로 채색(의사색채/회색조/혼합).

3D 이미지의 원리: EpicEye는 텍스처와 깊이가 픽셀 단위로 정렬된 organized 센서입니다. "3D 이미지"는 같은 2D 격자에 깊이값을 색으로 칠한 평면 깊이맵(회전 불가)이고, "3D"는 그 깊이를 3차원 점으로 펼친 포인트 클라우드(회전 가능)입니다.

3D 뷰 프리셋 / 줌

- 2D(카메라)/Front/Top/Side/Iso/Fit — 시점 즉시 전환(카메라 좌표 기준).
- 2D 줌    — 2D/3D 이미지 확대·축소(마우스 휠, 드래그 이동, fit보다 작게도 축소).

깊이 셰이더 / 포인트클라우드

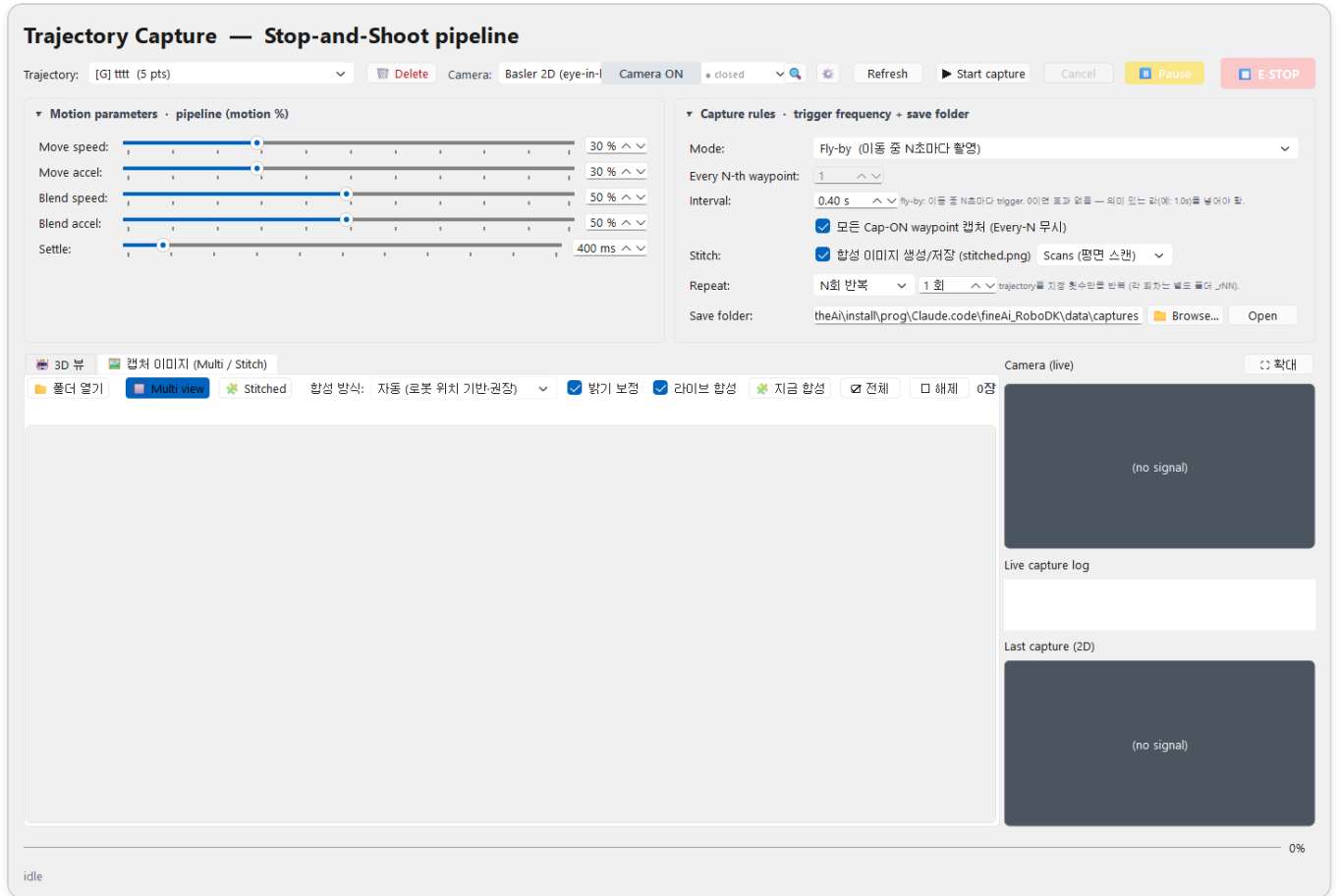
- **Depth shader** 컬러맵(JET/Turbo/Rainbow 등). 어떤 컬러맵이든 **가까움=빨강, 멀=파랑**으로 자동 정렬됩니다.
- **Z min/max** — 깊이 색상 범위, **점 크기** / **불투명도** / **다운샘플**.

설정 / LED / 다운로드

- 2D/3D 노출·Gain, 프로젝터, 후처리(5단계), 💡 **LED 보조조명**을 구조화 UI로 조절 → **Apply**.
Advanced JSON으로 전체 config 편집.
- **다운로드**: 2D 이미지(PNG) / Depth 의사색채(PNG) / Depth raw(NPY) / 포인트클라우드(PLY). 한글 폴더 경로 지원. PLY는 **EpicEye organized 바이너리** 형식(전체 그리드)으로 저장됩니다.
- 창 크기·위치는 **닫을 때 저장되어 다음에 그대로 복원**됩니다(첫 실행만 왼쪽 메뉴가 다 보이는 크기로 자동). 최대화 가능.

Trajectory Capture

Teaching에서 만든 궤적을 따라 로봇을 움직이며 자동 촬영합니다.



Trajectory Capture — 모션 파라미터 · 촬영 규칙 · 캡처 이미지(Multi / Stitch) 갤러리(폴더 열기·합성 방식·밝기 보정·라이브 합성·결함 맵)

모드

모드	동작
Stop-and-shoot	각 waypoint에 정지·정착 후 촬영.
Fly-by	이동 중 N초마다 촬영(촬영 수 = 주행시간/간격). 카덴스가 leg 간 연속.

반복 (Repeat)

- 없음(1회) / N회 반복 / N분간 반복. 한 워커 스레드가 전체를 처리해 안정적이며 Cancel이 항상 동작합니다.
- 회차마다 별도 폴더(_r01, _r02 ...)에 저장.

촬영 대상 규칙

- Cap OFF는 항상 제외** — Teaching 편집기에서 Cap을 끈 waypoint는 모드/옵션과 무관하게 절대 촬영되지 않습니다(stop-and-shoot-fly-by 공통). On(enabled) 해제도 재생·촬영에서 제외.
- 모든 Cap-ON waypoint 캡처(Every-N 무시)** 체크 — 커짐: Cap이 켜진 모든 점에서 촬영. 꺼짐: Cap-ON 중 N번째마다(Every N-th) 촬영.
- Interval(초)** — 연속 촬영 최소 간격 throttle.

제어 (Start / Pause / E-STOP)

- **II Pause / ► Resume** — 캡처 파이프라인과 로봇 모션을 **함께** 일시정지/재개합니다(N분 모드 포함; 일시정지한 시간만큼 지속시간 보정). 시뮬레이션에서도 동작.
- **■ E-STOP** — 워커 취소 + 펌웨어 비상정지(task stop)를 즉시 송출(Lock 무관). 3D CAM Jobs·Weld 검사의 정지 버튼도 동일하게 실제 비상정지를 수행합니다.
- **Move/Blend Speed** 등 모션 파라미터가 사전이동을 포함해 실제 반영됩니다.

저장 위치

기본 `data/captures/` 아래 `capture_<궤적 이름>_<날짜_시각>[_rNN]/` 폴더로 저장됩니다. 궤적 이름의 파일명 불가 문자는 자동으로 안전하게 변환되고 한글 이름도 지원합니다. 3D 점군은 **바이너리 PLY**로 저장됩니다.

우측 **Camera (live) + Last capture** 뷰,  **확대로** 큰 창 표시. 캡처 중에는 촬영 프레임이 라이브 뷰에 갱신됩니다.

합성(Stitch) — 캘리브레이션 연동 · 자동 면분할

- **로봇 위치 기반 합성** — 매니페스트(manifest.json)의 로봇 이동거리로 프레임을 배치하고, 각 출력 픽셀은 중심이 가장 가까운 **한 프레임**에서 취득(단일소스)하여 개별 이미지의 선명도를 그대로 유지합니다(접합부만 영향).
- **캘리브레이션 연동** — 해당 카메라가 캘리브레이션(내부 파라미터+Hand-Eye)되어 있으면 **왜곡 보정(undistort)** 후 **카메라 프레임으로 배치**해, 카메라 장착 방향과 무관하게 합성 좌우/상하가 실제 시야와 일치합니다(`+undist+cam`).
- **배율(px/mm) 학습·재사용** — 촬영 매수·속도가 달라도 같은 제품이 같은 크기로 합성되도록, 충분한(밀집) 촬영에서 측정한 px/mm를 카메라별로 학습해 저장하고, 희소 촬영(매수 적음)에서는 그 값을 재사용합니다.
- **자동 면(face) 분할** — ㄱ자 필릿 용접처럼 카메라 방향이 바뀌며 여러 면을 촬영한 스캔은 방향(자세) 변화를 감지해 **면별로 분리 합성**(`stitched_face1.png` , `stitched_face2.png` + 합본 `stitched.png`)합니다. **단일 대면적(방향 일정)** 스캔은 기존처럼 1장으로 합성됩니다.

Stitched 뷰 — 배율 설정 · 길이 표시 · 거리 측정

- **배율(px/mm)** — 합성 시 추정된 값이 자동으로 채워지며, 값을 바꾸면 아래 길이 표시가 즉시 갱신됩니다.  이 배율로 재합성으로 설정한 배율로 다시 합성하고, **기준 배율 저장**으로 이 카메라의 기준 px/mm로 저장합니다.
- **길이 표시** — 합성 이미지의 가로/세로 픽셀 ÷ px/mm 로 제품 길이(mm)를 표시합니다.
-  **거리 측정** — 버튼을 켜고 합성 이미지에서 **두 지점을 클릭**하면 그 사이 실제 거리(mm)를 현재 px/mm로 계산해 표시합니다. 휠로 확대해 정밀하게 찍을 수 있고, '지우기'로 초기화합니다.

조명 제어 (LED 조명 제어기)

Camera 페이지와 Trajectory Capture에서 **공통**으로 쓰는 LED 조명 제어(LAN/TCP) 기능입니다. 각 페이지의 접이식 “💡 조명 제어” 패널에서 설정하며, 설정은 한 곳(`config/lighting.yaml`)에서 공유됩니다. 현재 **Leimac IDGC**와 **CCS** 제어기를 지원합니다. 등록된 `vendor` 에 맞는 드라이버로 제어하며 UI 문구는 브랜드 중립입니다.

제어기 검색 / 등록

- 🔍 **조명 검색** — 서브넷을 스캔해 응답하는 제어기를 찾아 목록에 표시하고(채널 수 자동 식별), 선택해 등록합니다. 기본 서브넷은 `192.168.0`. **Leimac(포트 1000)**은 `.30 → .31 → .32 ...`, **CCS(포트 40001)**는 `.2 → .3 ...` 부근을 먼저 시도한 뒤 서브넷 전체를 스캔하며, 발견 즉시 목록에 나타냅니다.
- + **이 IP 연결·등록** — 제어기 IP를 알면 스캔 없이 바로 연결·등록합니다(기본 `192.168.0.30`). 무응답이면 채널 수를 직접 지정해 등록할 수 있습니다.

조명 제어기는 대개 **동시 TCP 연결 1개**만 허용합니다. 본 프로그램은 명령마다 연결을 짧게 열고 닫아(즉시 RST 해제) 슬롯을 점유하지 않으므로, 검색·라이브 제어·캡처가 서로 막지 않습니다.

채널 설정 · Live / Capture 세기

- 채널별로 **사용** 체크, **Live·Capture** 세기를 지정합니다. 세기 범위는 제어기별로 다릅니다(**Leimac IDGC 0~255** · IDMU 등 확장형은 최대 **1000**, **CCS 0~999**) — 패널 머리글에 각 제어기의 유효 범위가 표시됩니다. 확장형은 `config/lighting.yaml` 의 해당 제어기 `max_intensity` 를 `1000` 으로 지정하세요.
- Live 유지 / Live 적용 / 모두 OFF** — 미리보기 중에는 Live 세기를 유지하고, 즉시 적용하거나 모두 끌 수 있습니다. 채널 **테스트** 버튼으로 Capture 값으로 잠깐 켜다가 Live로 복귀해 확인합니다.

촬영 시 동작 (Snapshot · Capture)

스냅샷과 Trajectory Capture 촬영은 자동으로 **Capture 세기로 ON → 촬영 → Live 세기로 복귀** 순서로 동작합니다(조명 미설정 시 무동작). 노출 순간에만 밝게 커므로 조명 수명·발열에 유리합니다.

fineAi_PA_Engine · Vision Graph

fineAi_PA_Engine은 비전 알고리즘을 코드 없이 **노드 그래프**로 구성하는 엔진입니다. 연산자(노드)의 출력 포트를 다음 노드의 입력 포트에 연결하면, 데이터가 그 방향(출력→입력)으로 흐르는 방향성 그래프(DAG)가 됩니다. 점군/이미지 전처리 → 인식 → 6DoF 포즈까지 한 화면에서 설계합니다.

화면 구성

영역	역할
좌측 — 연산자 팔레트	카테고리별 연산자 트리. 더블클릭 또는 선택 후 + 추가 버튼으로 캔버스에 노드 생성. 노드가 화면 중앙으로 자동 포커스됩니다.
중앙 — 캔버스	노드 배치/연결. 빈 캔버스에는 안내문이 표시됩니다.
우상 — 파라미터	선택한 노드의 파라미터가 자동 생성 폼 (슬라이더/콤보/체크/파일선택 등)으로 표시·편집됩니다. Load Cloud (PLY) 노드를 선택하면 그 아래  세부정보 패널에 PLY 정보(포맷 ascii/binary, 점 개수, organized W×H, 속성-RGB, 유효점, XYZ 범위(mm), 파일 크기)가 표시됩니다(새로고침 버튼).
우하 — 미리보기	2D 탭(이미지·마스크) / 3D 탭(점군 + 포즈 축). 실행 후 Preview2D / Preview3D 노드 결과가 표시됩니다.
상단 — 톨바	새 그래프 · 열기 · 저장 · 검증 · ▶ 실행 · ⏮ 선택까지 · ■ 취소 · Fit · 카메라 선택.

기능 그룹 (팔레트 구성)

연산자는 비전 Pick&Place 파이프라인 순서(카메라 → 인식 → 분할 → 검사 → 자세 → 파지)에 맞춰 6개 기능 그룹으로 정리되어 있습니다. 위에서 아래로 따라가면 하나의 그래프가 됩니다.

그룹	역할	주요 연산자
1. 카메라 영상	이미지/점군 취득·전처리	CameraSource, Load(Image/PLY), Split, ToGray, GaussianBlur, CropROI, Voxel, SOR/Radius, PassThrough, CropBox, Transform, Normals
2. 인식	물체 검출·매칭	TemplateMatch2D, FeatureMatch2D, OnnxDetect2D, SurfaceModelMatch3D, Contours
3. 영역 분할	평면/군집/마스크 분할	PlaneSegmentRANSAC, RemovePlane, EuclideanCluster, KMeans, MaskToCloud, Threshold, Morphology, Canny
4. 검사	판정·필터·결과 확인	FilterDetections, ThresholdDecision, Preview2D/3D, OutputScalar/Detections
5. 자세 추정	6DoF 포즈·좌표 변환	ClusterToPose, ICPRefine, PnPPOSE, PlaneFrameToPose, PickBestPose, Apply/Invert/Compose Transform, HandEyeToBase
6. 파지 경로	그랩 선정·접근·출력	GraspRank, SortPoses, OffsetPose, OutputPose

참조: 일반적인 3D 솔루션은

Camera→Detection→Segmentation→Inspection→Pose→Action 파이프라인 구성을 따릅니다.

에디터 조작

동작	방법
노드 추가	팔레트 항목 더블클릭 / 선택 후 + 추가 / Enter
노드 이동	노드 본체를 드래그
연결	출력 포트(우측 점)를 드래그해 입력 포트(좌측 점)에 놓기. 타입이 호환되면 선이 초록, 불가하면 빨강 . 입력 포트당 1개만 연결(새로 연결하면 교체).
삭제	노드/엣지 선택 후 Del
화면	마우스 휠 = 줌, 중간버튼 드래그 = 팬, Fit = 줌 초기화

“**새 그래프**”를 누르면 캔버스를 비우고 **CameraSource** 시작 노드를 자동 배치합니다. 여기서부터 연산자를 이어 붙이세요.

빠른 시작 — 빈피킹 그래프 만들기 (예)

1. **새 그래프** 클릭 → **CameraSource** 가 생깁니다.
2. 팔레트에서 **CropBox** → **VoxelDownsample** → **RemovePlane** → **StatisticalOutlierRemoval** → **EuclideanCluster** → **ClusterToPose** → **GraspRank** → **OutputPose** 를 차례로 추가합니다.
3. **CameraSource.cloud** → **CropBox.cloud** 부터 순서대로 포트를 드래그해 연결합니다.
(**ClusterToPose** 는 **cloud** 와 **labels** 두 입력 필요: 점군은 SOR에서, labels는 EuclideanCluster에서.)
4. **OutputPose** 노드를 선택하고 우상 파라미터의 **출력 이름**을 **grasps** 로 둡니다.
5. 상단에서 카메라를 선택하고 **검증** → 통과하면 ▶ **실행**. 우하 3D 탭에서 점군·포즈를 확인합니다.
6. **저장**으로 이름을 주면 **data/atom_graphs/** 에 JSON으로 저장됩니다. Epic Pro에서 이 그래프를 잡에 연결합니다.

처음이라면 Epic Pro 화면의 “**샘플 템플릿 생성**”으로 위 3개 그래프를 자동 생성한 뒤, fineAi_PA_Engine에서 **열기**로 불러 구조를 살펴보는 것을 권장합니다.

포트 타입

연결은 출력→입력 방향이며 타입이 같아야 합니다(예외: Pose6D→PoseSet 자동 허용).

타입	데이터
Image2D	HxWx3 BGR 이미지
Depth	HxW 깊이맵(mm)
PointCloud	Nx3(xyz mm) 또는 Nx6(xyz+rgb)
Mask	HxW 이진 마스크
Labels	점별 군집 ID(-1=노이즈)
Pose6D / PoseSet	4x4 포즈 / 포즈 집합(+점수)
Transform	4x4 강체변환
Detections	검출 목록(박스·점수·포즈)
Intrinsics · Scalar · ROI	내부파라미터 · 스칼라 · 관심영역

연산자 레퍼런스 (주요 파라미터)

연산자	설명 · 핵심 파라미터
Source / IO	
CameraSource	입력 프레임을 image/depth/cloud/intrinsics로 분해. 프레임이 없고 카메라가 선택되면 1회 촬영.
LoadImage / LoadCloudPLY	디스크에서 이미지/PLY 점군 읽기(유니코드 경로 안전). <code>path</code>
SplitFrameBundle	FrameBundle을 4개 포트로 분리.
점군 전처리	
VoxelDownsample	복셀 평균 다운샘플. <code>voxel_mm</code>
StatisticalOutlierRemoval	k-이웃 평균거리 이상점 제거. <code>k</code> , <code>std_ratio</code>
RadiusOutlierRemoval	반경 내 이웃 부족점 제거. <code>radius_mm</code> , <code>min_neighbors</code>
PassThroughFilter / CropBox	축 범위 / 박스 영역으로 필터. <code>axis,min,max</code> / <code>x · y · z min · max</code>
TransformCloud / EstimateNormals	4x4 변환 적용 / k-이웃 PCA 법선 추정.
세그멘테이션 / 군집	
PlaneSegmentRANSAC / RemovePlane	지배 평면 추정(평면프레임 출력) / 평면 제거. <code>threshold_mm</code> , <code>iterations</code>
EuclideanCluster(DBSCAN)	점군 군집화 → labels + 검출. <code>eps_mm</code> , <code>min_samples</code>
KMeansCluster / MaskToCloud	고정 k 군집 / 2D 마스크로 픽셀정렬 점군 추출.
인식 / 매칭	
TemplateMatch2D / FeatureMatch2D	NCC 다중스케일 / ORB+호모그래피. <code>template_path</code>
SurfaceModelMatch3D	PCA 전역정합으로 model→scene 초기 포즈(ICP의 init).
OnnxDetect2D	ONNX 검출모델(onnxruntime). <code>model_path</code> , <code>score_thr</code>

포즈	
ClusterToPose	군집 PCA OBB → 6DoF 포즈집합. <code>min_points</code>
ICPRefine	점-점 ICP(cKDTree+Kabsch) 정밀 정합. <code>max_iter</code> , <code>max_corr_mm</code>
PnP Pose / PlaneFrameToPose / PickBestPose	2D→3D PnP / 평면프레임→포즈 / 최고점수 포즈 선택.
지오메트리 / 결정 / 출력	
Apply-Invert-Compose Transform / OffsetPose	포즈·변환 연산 / 로컬축 오프셋(접근·이탈).
HandEyeToBase	카메라프레임 포즈 → 로봇 베이스(hand-eye + 현재 TCP).
FilterDetections·SortPoses·GraspRank·ThresholdDecision	필터·정렬·그래프순위·임계판정.
OutputPose·Detections·Scalar / Preview2D·3D	명명 출력 발행 / 에디터 미리보기.

실행 결과 포즈는 **OutputPose**의 “출력 이름”으로 발행되며, Epic Pro가 잡의 **포즈 출력 이름**으로 이 값을 읽습니다.

PA Engine 잡 (레거시 · 구 3D CAM - Jobs)

레거시 안내(09-18) — 이 화면은 파지 계획·도달성/충돌 검증·그리퍼 드라이버 없이 그래프 포즈로 직접 이동합니다. 같은 일을 검증·재시도·적재·결과 DB 와 함께 하려면  **3D Pick&Place** 통합에서 인식 타입 `pa_graph` (PA Engine 그래프 포즈)를 고르고 스킬을 `correct_trajectory` (locate-correct: 기준 궤적 id + 기준 물체 포즈 → 인식 편차만큼 궤적을 보정해 재생, 적재 없음) 또는 `simple_pick` 으로 두세요. 기존 Job 은 그대로 실행됩니다.

3D CAM - Jobs는 비전을 로봇 동작으로 연결하는 **Job** 런타임입니다. 한 사이클은 **촬영** → **fineAi_PA_Engine** 그래프 → **6DoF 포즈** → **Hand-Eye 변환**(카메라→로봇 베이스) → **로봇 동작** 순으로 수행됩니다. 로봇은 컨트롤러 추상 인터페이스로만 제어하므로 **RB5-850**를 비롯한 기종에 그대로 적용됩니다. (메뉴에서는 “잡” 대신 **Job** 용어를 사용합니다.)

사전 준비 (중요)

1. **카메라 등록·연결** (Camera 메뉴) — EpicEye 등.

2. **Intrinsic + Hand-Eye 캘리브레이션** (Camera 메뉴) — 로봇 베이스 좌표로 포즈를 변환하려면 해당 (로봇, 카메라) 쌍의 **eye-in-hand** 캘리브레이션이 필요합니다. 결과는 `config/hand_eye.yaml` 에 저장됩니다.
3. **fineAi_PA_Engine 그래프** — 대상 포즈를 `OutputPose` 로 발행하는 그래프를 저장.
4. **로봇 연결·Lock 획득** (Robots 메뉴) — 실제 동작 시.

잡 편집 탭 — 필드

필드	의미
이름 / 템플릿	잡 식별자 / 응용 종류(custom·locate_correct·bin_pick·depalletize).
로봇 / 카메라 / 그래프	사용할 로봇(robots.yaml) · 카메라 · fineAi_PA_Engine 그래프.
포즈 출력 이름	그래프의 어떤 OutputPose 이름을 소비할지(예 <code>grasps</code> , <code>boxes</code> , <code>pose</code>).
로봇 동작	none (포즈만)· pick (집기)· place-correct (보정 포즈)· sequence (순서 픽).
접근/이탈 거리(mm)	포즈 로컬 -Z 방향 standoff. 집기 전 접근, 집은 후 들어올림.
속도/가속(%)	이동 속도·가속 비율.
최대 사이클	0 = 장면이 빌 때까지(검출 없음 시 종료), N = N회.

샘플 템플릿 생성 버튼으로 아래 3개 응용의 그래프+잡을 `data/` 에 한 번에 생성할 수 있습니다.

응용 템플릿

응용	그래프 파이프라인	로봇 동작 · 비고
위치보정 locate- correct	CropBox → RemovePlane → SOR → SurfaceModelMatch3D → ICPRefine → OutputPose	correct — 대상의 현재 6DoF 포즈를 베이스 좌표로 산출. (모델 PLY 경로를 <code>LoadCloudPLY</code> 에 지정해야 매칭 가능.) 조립/검사 위치 보정에 사용.
랜덤 빈피킹 bin-pick	CropBox → Voxel → RemovePlane → SOR → EuclideanCluster → ClusterToPose → GraspRank → OutputPose("grasps")	pick — 가장 도달성 좋은(가까운) 물체부터 접근→집기→이탈→플레이스. 매 사이클 재촬영, 군집이 없으면 종료.
디팔레타이징 depalletize	CropBox → PlaneSegmentRANSAC(상단층) → EuclideanCluster → ClusterToPose → SortPoses(z↓) → OutputPose("boxes")	sequence — 가장 높은 층의 박스부터 순서대로 픽. 층이 비면 재촬영해 다음 층.

템플릿은 시작점입니다. CropBox 범위(작업영역), 군집 `eps_mm` / `min_samples`, 모델 PLY 경로 등을 현장에 맞게 조정하세요.

위치보정(correct) 사용법

대상물이 기준 위치에서 틀어졌을 때, 그 차이만큼 **기준 궤적 전체를 보정**해 재생합니다.

1. 잡 편집에서 동작을 **correct**로, **기준 궤적**을 조립/검사 궤적으로 지정하고 저장합니다.
2. 부품을 **기준 위치**에 두고 런타임의 **기준 포즈 설정**을 누릅니다 → 1회 촬영·산출한 베이스 좌표 포즈가 잡에 저장됩니다("기준 포즈: 설정됨").
3. 이후 부품이 이동한 상태에서 ▶ **실행** → 현재 포즈와 기준 포즈의 차이 $\Delta = T_{\text{현재}} \cdot \text{inv}(T_{\text{기준}})$ 를 계산해 기준 궤적의 모든 **Cartesian** 웨이포인트에 적용하고 `move_l`로 재생합니다.

보정은 **Cartesian(TCP) 웨이포인트**에 적용됩니다. 관절(joint)만 있는 웨이포인트는 보정 대상에서 제외됩니다. 기준 포즈가 없거나 로봇이 미연결이면 **보정량만 계산**하고 재생은 생략합니다(안전).

런타임 탭 — 실행

1. 잡을 선택하고 ▶ **실행**.
2. 각 사이클: 촬영 → 그래프 실행 → 포즈 산출 → **Hand-Eye 변환** → 로봇 동작.
3. 오른쪽 3D 뷰에 베이스 좌표 포즈가 **RGB 축(triad)**으로, 아래 표에 사이클별 **검출 수·판정·메시지**가 누적됩니다.
4. ■ **정지**는 사이클 사이에 즉시 반응합니다. 종료 시 실행 로그가 `data/job_runs/`에 저장됩니다.

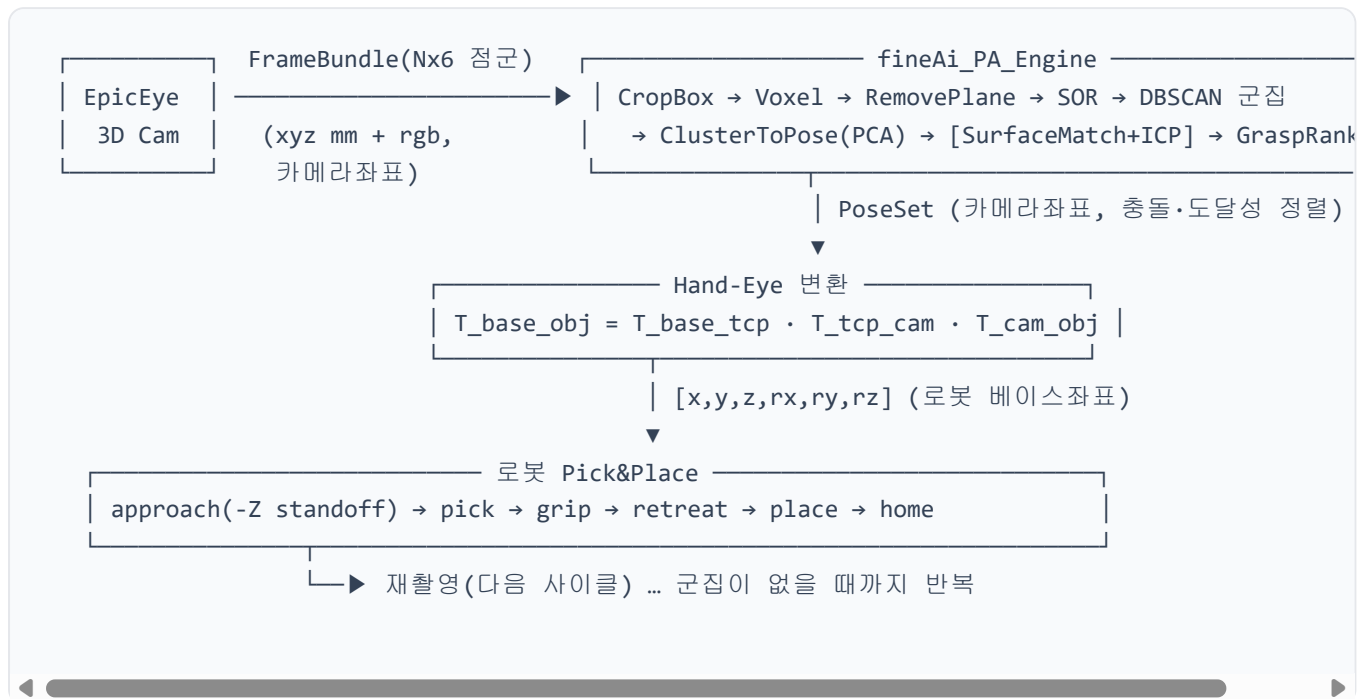
판정	의미
ok	정상(포즈 산출/동작 완료).
no_detection	검출 없음 → 장면 소진으로 보고 루프 종료.
unreachable / motion_failed	도달 불가 / 모션 거부·타임아웃.
error	촬영 실패·그래프 오류·캘리브레이션 없음 등(메시지 참조).

로봇 동작(pick/place/sequence)은 **Hand-Eye 캘리브레이션 + 연결·Lock 보유 로봇**이 필요합니다. 캘리브레이션이나 로봇이 없으면 동작 없이 **포즈만 산출**합니다(안전 동작).

`correct`는 현재 보정 포즈만 산출합니다.

Pick&Place 동작 원리

3D 카메라 → fineAi_PA_Engine(인식) → Hand-Eye 변환 → 로봇 동작이 한 **사이클**로 반복됩니다. (외부 로봇 시뮬레이션 SDK 없이 자체 구현입니다.)



1) 인식 — 점군에서 집을 물체 찾기

단계	알고리즘	의미
CropBox	AABB 범위 필터	빈(bin) 내부만 남기고 배경 제거
VoxelDownsample	복셀 격자 평균	점 수 축약 → 가속
RemovePlane	RANSAC 평면적합 $ n \cdot p + d < thr$	바닥/팔레트 평면 제거 → 물체만
SOR	kNN 평균거리 > mean+σ·std 제거	부유 노이즈 제거
DBSCAN	밀도 군집(eps, min_samples)	물체 인스턴스 분리(labels)
ClusterToPose	PCA: 중심 + 공분산 고유벡터	군집별 6DoF 포즈(OBB)
(옵션) Surface+ICP	전역정합 + 점-점 ICP(Kabsch)	CAD 모델 기반 mm급 정밀 포즈
GraspRank	선호 + 접근통로 충돌 + 도달성	집기 순서 결정·위반 후보 제외

2) 전달 — 카메라좌표 → 로봇 베이스좌표 (Hand-Eye)

PA_Engine 포즈는 카메라 좌표이므로 로봇 베이스로 변환합니다:

$T_{base_obj} = T_{base_tcp} \cdot T_{tcp_cam} \cdot T_{cam_obj}$
 T_{tcp_cam} : eye-in-hand 캘리브레이션 (config/hand_eye.yaml)
 T_{base_tcp} : 촬영 순간의 현재 TCP 자세 (controller.latest_sample)
 T_{cam_obj} : PA_Engine이 인식한 물체 포즈 (카메라좌표)
 $\rightarrow matrix_to_pose() \rightarrow [x, y, z, rx, ry, rz]$ (mm, deg) $\rightarrow$ 로봇 명령

3) 로봇 Pick&Place 시퀀스

1. **Approach**: 그랩 포즈의 로컬 -Z로 `approach_mm` 후퇴한 지점 $p = T \cdot (0, 0, -approach, 1)$ 으로 이동.
2. **Pick**: 그랩 포즈로 직선 이동(`move_l`) $\rightarrow$ `wait_motion_complete` 로 도달 확인.
3. **Grip**: 그리퍼 닫기(디지털 출력, 있을 때).
4. **Retreat**: 로컬 -Z로 `retreat_mm` 들어올림.
5. **Place**: `place_pose` 로 이동 $\rightarrow$ 그리퍼 열기.
6. **Home**: 홈 복귀 $\rightarrow$ 다음 사이클 재촬영. 군집이 없으면(`no_detection`) 종료.

매 이동은 도달 핸드셰이크로 확인하며, 사이클 사이 **정지**가 즉시 반영됩니다. 그리퍼 IO가 없는 컨트롤러에서는 파지 단계를 건너뛰니다(동작 계속).

GraspRank — 충돌 회피 · 도달성

거리/크기 선호에 더해, 장면 점군(`cloud` 입력)이 연결되면 **그리퍼 접근 통로**의 충돌을 점수화하고, 워크스페이스가 설정되면 **도달 가능 범위**를 검사합니다.

파라미터	역할
collision_check	cloud가 있을 때 접근 통로 충돌 검사 on/off
gripper_radius_mm	접근 통로(원기둥) 반경 $\approx$ 그리퍼 스윙
approach_mm	충돌을 검사할 통로 길이(접근 standoff)
object_radius_mm	대상 자신을 충돌에서 제외하는 반경
collision_weight	충돌 점수의 페널티 가중치
reject_collision_pts	통로 충돌점이 이 값 초과 시 후보 제외 (0=off)
reach_center / reach_min_mm / reach_max_mm	워크스페이스 구(중심·반경). 범위 밖이면 제외(0=off)

충돌 검사는 포즈가 표현된 좌표계 기준의 기하 연산이라 카메라좌표에서도 유효합니다. 단, **도달성(reach)**은 로봇 베이스 기준이 자연스러우므로, 베이스 도달성을 쓰려면 포즈를 베이스로 변환한 뒤 적용하세요(또는 빈피킹 템플릿처럼 충돌 검사 위주로 사용).

부품별 튜닝 가이드

부품의 표면 특성에 따라 점군 품질이 달라집니다. 아래 기준값에서 시작해 ▶▶ **선택**까지로 단계별 점군을 보며 조정하세요.

표면 특성별 촬영(EpicEye) 설정

부품	주의점	권장 설정
일반(무광 플라 스틱/금속)	표준	기본 노출, LED 보조광 on, 1회 촬영으로 충분
반짝이는/반사 (광택 금속)	스페큘러 → 구 명/노이즈	노출 ↓ + LED 각도 분산, HDR/다중노출, 편광 권장. SOR <code>std_ratio</code> ↓(엄격)
검은색/저반사	반사광 부족 → 점군 희박	노출 ↑·LED ↑, ProjectorBrightness ↑. <code>min_samples</code> ↓, Voxel ↓(점 보존)
투명/반투명	깊이 누락·왜곡 큼	3D 단독 어려움 → 2D 인식 병행 (TemplateMatch/Onnx) 또는 코팅/배경 대비. Pixel Pro류 투명대응 모드 활용

형상별 인식 파이프라인 선택

형상	전처리·군집	포즈 추정
박스/직육면체 (디팔레)	PlaneSegmentRANSAC로 상단층 → DBSCAN <code>eps</code> 크게(박스 간격), <code>min_samples</code> 크게	ClusterToPose(OBB)면 충분. SortPoses(z↓)로 층 순서
봉지/부정형(빈 피킹)	RemovePlane → SOR 강하게 → DBSCAN <code>eps</code> 작게(밀착 분리)	ClusterToPose. 형상 일정치 않으면 모델매칭 부적합 → 중심/법선 그래프
동일 규격 부품 (정밀)	RemovePlane → SOR	SurfaceModelMatch3D → ICPRefine (CAD/기준 PLY) 로 mm급. 위치보정(correct)에 적합
평면 인쇄/마커	2D 경로(ToGray→Threshold→Contours 또는 FeatureMatch2D)	PnP Pose(2D→3D)

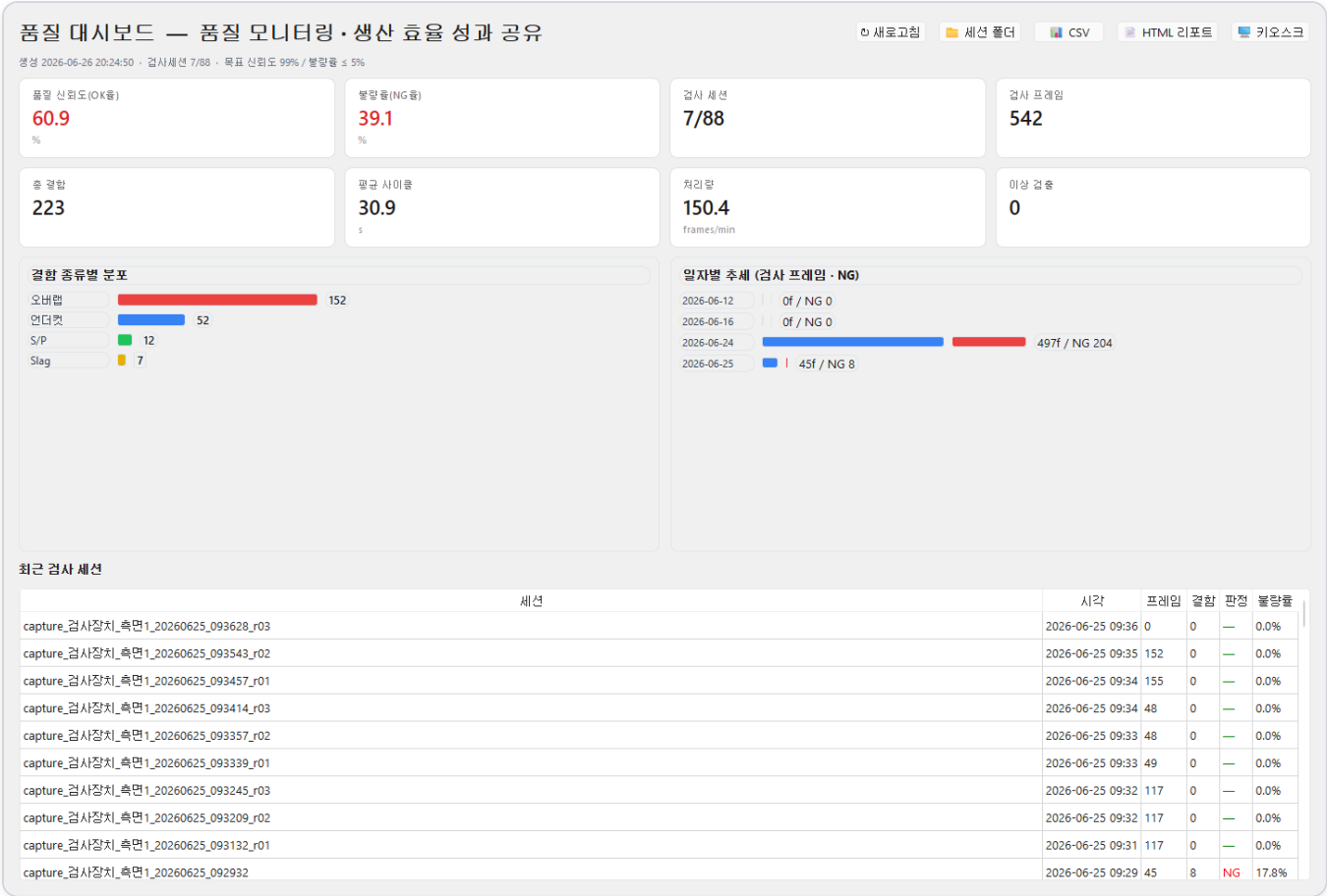
핵심 파라미터 빠른 기준

파라미터	의미	증가 시	감소 시
RemovePlane <code>threshold_mm</code>	평면 두께 허용	평면 더 많이 제거(굴곡 면도)	얇은 평면만 제거(물체 보존)
SOR <code>std_ratio</code>	이상점 민감도	관대(노이즈 잔존)	엄격(반사 노이즈 제거, 과제거 주의)
DBSCAN <code>eps_mm</code>	군집 이웃 반경	인접 물체가 한 군집으로 병합	한 물체가 여러 군집으로 분할
DBSCAN <code>min_samples</code>	군집 최소 점수	작은/희박 군집 무시(검출↓)	희박 물체도 검출(노이즈도↑)
Voxel <code>voxel_mm</code>	다운샘플 격자	빠르지만 디테일 손실	정밀하지만 느림
ClusterToPose <code>min_points</code>	포즈 산출 최소 점	작은 물체 누락	작은 물체도 포즈화

튜닝 순서 권장: ① CropBox로 영역 한정 → ② RemovePlane 임계 → ③ SOR → ④ DBSCAN `eps/min_samples`(물체 1:1 분리될 때까지) → ⑤ GraspRank 충돌/도달성. 각 단계는 **선택** **까지**로 중간 점군을 확인하며 맞춥니다.

Quality Dashboard (품질 대시보드 · 생산 효율 성과 공유)

검사된 캡처 세션들을 한 화면으로 집계해 품질·생산 KPI를 보여주고, 성과를 내보내/공유합니다. Object Detection ⑤ 검사에서 세션 manifest에 기록된 판정(OK/NG)·결함 검출을 자동으로 모읍니다.



Quality Dashboard — 품질 신뢰도(OK율)·불량률(NG율)·처리량 KPI 카드, 결함 종류별 분포, 일자별 추세, 최근 검사 세션 표, CSV/HTML·키오스크 공유

- **KPI 카드** — 품질 신뢰도(OK율)·불량률(NG율)(목표 대비 색상)·검사 세션·검사 프레임·총 결함·평균 사이클(s)·처리량(frames/min)·이상 검출. 목표값은 `app.yaml` → `quality.targets` (기본 신뢰도 99% / 불량률 ≤ 5%).
- **결함 종류별 분포**(클래스 막대) · **일자별 추세**(검사 프레임 vs NG) · **최근 검사 세션 표**(시각·프레임·결함·판정·불량률).
- **성과 공유/내보내기** — 「 **CSV**」(KPI+클래스+세션) · 「 **HTML 리포트**」(브라우저로 열림) · 「 **세션 폴더**」로 외부/복사 세션 폴더를 집계에 포함.
- 「 **키오스크**」 — 현장 대형 화면/모바일 공유용 **전체화면 KPI**(15초 자동 갱신, Esc로 닫기). 보고서의 "현장 키오스크 전사 공유"에 해당.

대시보드는 페이지를 열 때마다 백그라운드로 다시 집계합니다(세션이 많아도 화면이 멈추지 않음). 품질 KPI는 **검사 판정이 있는 프레임**만 집계하며, 검출 없이 촬영만 한 세션은 세션/프레임 수에는 포함되지만 OK/NG율에는 반영되지 않습니다.

Capture Quality (촬영 품질)

촬영된 세션의 2D 이미지/포인트클라우드를 불러와 비전 검사에 중요한 이미지 품질 지표를 종합적으로 확인합니다. 결함 검사 전에 **촬영이 양호한지** 거르는 1차 게이트입니다(결함 검출이 아님).

Capture Quality

세션 폴더: (기본: data/captures) — 별도 폴더를 선택하면 그 폴더의 세션도 표시

Session: 06-25 09:36 · 검사장치_측면1_r03 · 0컷 · — · 미평가

verdict: (not evaluated) — Evaluate 버튼 클릭

#

종합

선명도

밝기

대비

과노출%

노이즈

SNR

pts

선택 캡처 (2D)

(no signal)

이미지 품질 상세 (선택 캡처)

항목

값

판정

Pointcloud

+X

-X

+Y

-Y

+Z

-Z

ISO

FIT

session=capture_검사장치_측면1_20260625_093628_r03 · records=0 · not evaluated

Capture Quality — 촬영 세션의 이미지 품질 지표(선명도·노출·포인트 밀도 등) 종합 확인

- **세션 목록** — 최신순(생성일)으로 정렬되며, 각 항목에 날짜·시각 · 컷수 · 2D/3D · 품질 배지 (✓ 양호 n/n · ⚠ 주의 · ✗ 불량 · 미평가)가 함께 표시됩니다. 항목에 마우스를 올리면 전체 세션 ID·로봇·카메라가 보입니다.
- **세션 폴더 선택** — 기본은 `data/captures` 의 세션을 읽지만, 상단의 「 폴더 선택... 」으로 **별도 폴더**(외장드라이브·네트워크 공유·다른 PC에서 복사한 세션 폴더)를 지정하면 그 폴더의 세션도 목록에 나타납니다. 「 **기본값** 」으로 되돌립니다. 선택값은 저장되어 재시작 후에도 유지됩니다(`app.yaml` → `inspection.session_dir` ; Object Detection의 폴더 설정과는 별도로 기억).
- **자동 평가** — 세션을 처음 선택하면 자동으로 품질을 평가·저장해 목록 배지에 반영합니다. 「 **Evaluate** 」 버튼으로 언제든지 재평가할 수 있습니다.
- **이미지 품질 지표(2D)** — 한 컷마다 다음을 계산합니다.
 - **종합 판정 반영(게이팅)**: 선명도(Laplacian)·밝기·대비·과노출(≥ 250)·저노출(≤ 5)·엔트로피·노이즈(σ , Immerkär 추정)·SNR·조명 균일도
 - **참고 지표**: 선명도(Tenengrad)·동적범위(p1~p99)·색채도·해상도각 지표는 자체 임계값으로 **양호/주의/불량**을 판정하고, **종합 판정**은 게이팅 지표 중 최악값입니다(3D 세션은 점밀도 pts도 함께 반영).

- **화면 구성** — 좌측 표에 컷별 핵심 지표(**종합·선명도·밝기·대비·과노출·노이즈·SNR·pts**)를 판정 색으로 표시하고 행 전체를 판정 색으로 강조합니다. 컷을 선택하면 우측에 2D 이미지(좌상단에 **종합·선명도·밝기 오버레이**)와 「**이미지 품질 상세**」 패널이 나타나 그 컷의 **모든 지표(값·판정)**를 보여줍니다(항목에 마우스를 올리면 권장 임계값 툴팁).
- **리포트 저장** — 「**Save report**」로 평가 결과를 `data/inspections/<세션>.json`에 저장합니다.

임계값은 휴리스틱 기본값입니다(산업용 금속 비전·난반사 기준). 현장 데이터에 맞게 조정할 수 있으며, 임계값을 바꾼 뒤에는 평가 캐시를 지우거나 「**Evaluate**」로 재평가하면 새 기준이 적용됩니다.

품질검사(Inspection) 방식은 두 가지로 나뉩니다 — **Object Detection**(지도학습: 라벨 →YOLO 검출)과 **Anomaly Detection**(비지도: 정상학습→이상탐지). 아래 두 메뉴가 각각 담당합니다.

Object Detection (용접결함 검출 · 지도학습)

로봇 말단의 2D 카메라+링조명(eye-in-hand)으로 용접 비드 구간을 이동하며 촬영 → 라벨링(박스/폴리곤) → 학습(검출/분할) → 이미지 평가 → 이동 중 실시간 검출 → 이상탐지까지 한 화면(6개 탭)에서 수행합니다. 추론·학습에는 **PyTorch(CUDA) + Ultralytics YOLO**가 패키지에 포함됩니다(수 GB).

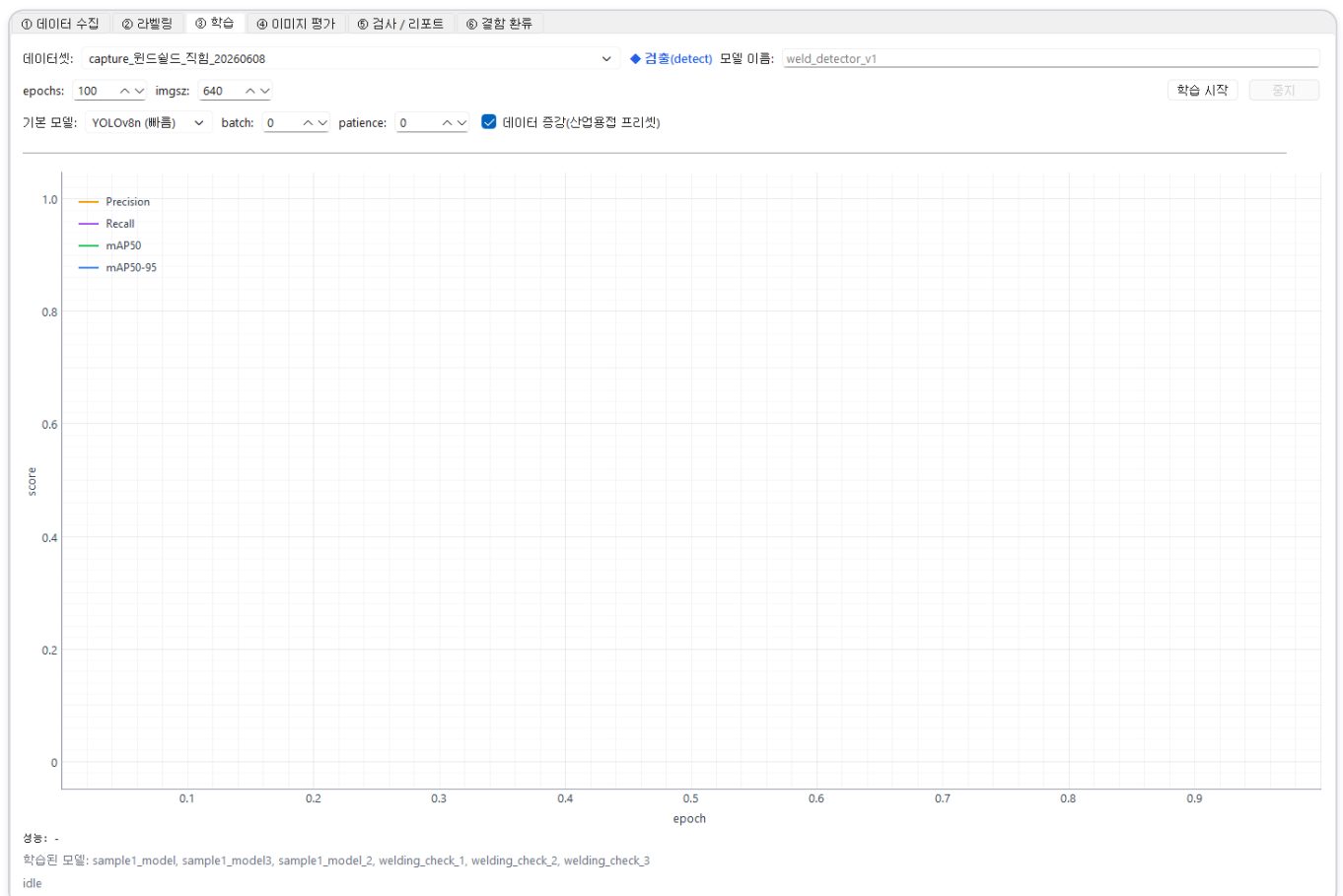
- ① **데이터 수집** — 검사 구간(예: ㄱ자 ~500mm)을 Teaching으로 정의하고 **Trajectory Capture**(검출 OFF)로 촬영. 수집 세션을 라벨링으로 보냅니다.
 - **세션 폴더 선택** — 기본은 `data/captures`의 세션을 읽지만, 「**폴더 선택...**」으로 **별도 폴더**(외장드라이브·네트워크 공유·다른 PC에서 복사한 세션 폴더)를 지정하면 그 폴더의 세션도 목록에 표시됩니다. 「**기본값**」으로 되돌립니다. 선택값은 저장되어 재시작 후에도 유지됩니다(`app.yaml` → `weld.session_dir`). 매니페스트의 이미지 경로가 달라도 자동 보정되어 **복사한 세션 폴더도 그대로 평가**됩니다.
 - **여러 세션을 하나의 데이터셋으로** — 「**여러 세션 합치기** →」 버튼으로 **여러 세션을 체크**해 한 번에 하나의 데이터셋으로 합쳐 **동일하게 라벨링**합니다(데이터셋 이름 지정, 중복 이미지 자동 제외). 한 세션씩 보내는 「**라벨링으로 보내기** →」도 그대로 사용할 수 있고, 라벨링에 데이터셋이 열려 있으면 "추가/새로 시작"을 묻습니다.
- ② **라벨링** — 이미지마다 결함을 표시하고 클래스를 지정. **라벨형식을 박스(detect) 또는 폴리곤(segment)으로** 선택합니다. 박스=드래그로 사각형; 폴리곤=점 클릭 후 더블클릭/Enter로 닫기(Esc 취소, Del 마지막 점 취소). 폴리곤(분할)은 언더컷·균열 같은 **연속 결함의 길이·면적 정량화**에 유리합니다. + **클래스 편집**으로 클래스 추가/이름변경. 기본 유형: 언더컷·오버랩·핀홀·블로홀·기공·균열·스패터·용입불량. 저장 시 YOLO 데이터셋(`data/weld_datasets/`, 라벨형식이 data.yaml의 task로 기록).
- ③ **학습** — 데이터셋·epochs·imgsz와 **기본 모델(YOLOv8n/s)·batch(0=자동)·patience(0=조기종료)·데이터 증강(산업용접 프리셋)**을 선택해 GPU 학습. 데이터셋이 **분할(segment)**이면 자동으로 -

seg 모델로 분할 학습합니다(번들된 `yolov8n-seg.pt` 사용, 데이터셋 옆에 ◆검출/◆분할 표시). 학습 중 **Precision·Recall·mAP50·mAP50-95 곡선** + box/cls/dfl 손실 표시. 결과 `best.pt` + `meta.json` (task 포함) + OD_Workbench 호환 `*_training_info.txt` 를 `data/weld_models/<이름>/` 에 저장.

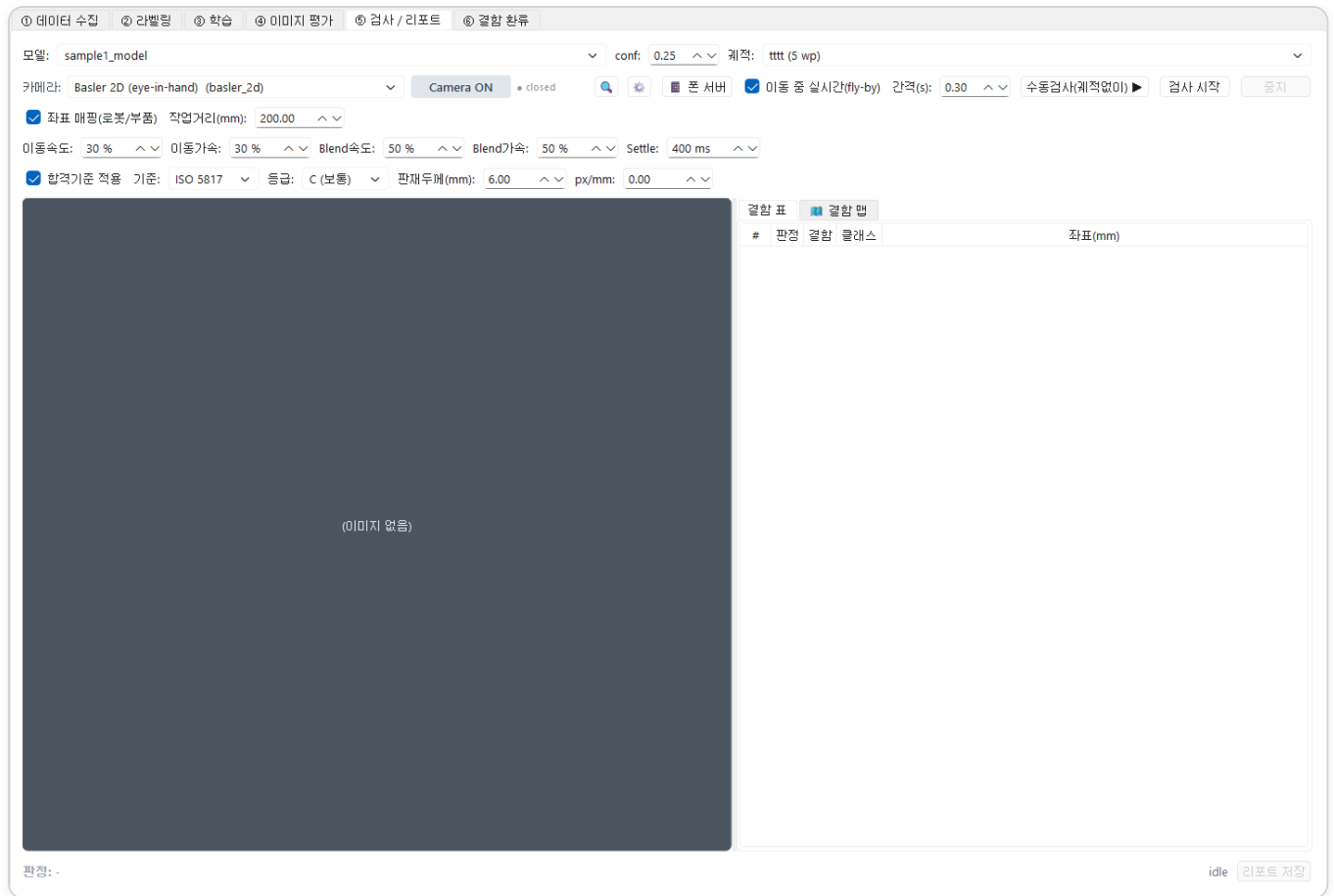
- ④ **이미지 평가** — 로봇 없이 폴더 또는 데이터셋 이미지로 학습 결과를 검증. 선택 시 이미지 목록이 바로 표시되고, 선택 이미지 1장 또는 ▶ **전체 평가** 실행. 행 클릭 시 검출 박스 오버레이, 주석 이미지 저장(`data/weld_reports/eval_...`).
- ⑤ **검사 / 리포트** — 모델·궤적·카메라를 선택하고 **이동 중 실시간(fly-by)** 검사. **카메라 ON 시 검사 시작 전부터 라이브 영상**이 표시되어 용접 비드를 보며 위치를 맞출 수 있습니다(검사 중에는 검출 박스 오버레이, 종료 후 라이브 자동 복귀). **이동 파라미터**(이동속도/이동가속/Blend속도/Blend가속/Settle)를 탭에서 직접 지정합니다(미지정 시 기본 30/30/50/50%-400ms). **좌표 매핑**(intrinsic+hand-eye+작업거리 평면)으로 결함을 로봇 베이스 좌표(mm)로 환산. **합격기준(ISO 5817)**을 적용할 수 있습니다 — 등급(B 엄격/C 보통/D 관대)·판재두께(t)-px/mm 스케일을 지정하면, 검출(+가능 시 치수)을 등급 한계와 비교해 **결함별 OK/NG와 사유**를 판정합니다(예: 균열=무조건 NG, 언더컷 깊이≤한계, 기공경/개수, 오버랩 등급별). 깊이를 측정 못 한 연속결함은 보수적으로 NG 처리합니다. 검사 결과는 **실시간 저장**됩니다 — 검출/판정은 세션 manifest에, 결함 프레임의 주석 이미지는 즉시 `data/weld_reports/<세션>/images/` 에 기록(중단돼도 보존). **검사 완료 시 리포트가 자동 저장**됩니다(HTML 요약·CSV 상세·JSON·주석 이미지·**ISO 5817 판정표**·seam 위치별 결함 분포). 「리포트 저장」 버튼으로 수동 재저장도 가능. 판정은 **OK / NG**로 표시(프레임별·세션 종합). 시작 시 로봇 Lock이 없으면 즉석에서 획득을 묻습니다. **결함 맵(부품 전체)** — 우측 「 결함 맵」 탭에서 검사 프레임을 **로봇 촬영 위치로 합성**하고 그 위에 **결함 위치를 색상 박스로 오버레이**해, 부품의 어느 구역에 결함이 모였는지 한 장으로 봅니다. 검사 종료 시 자동 생성되어 `data/weld_reports/<세션>/defect_map.png` 로 저장되고, 「 결함 맵 생성」으로 다시 만들 수 있습니다(휠 확대/드래그 이동).
- ⑥ **결함 환류 (재학습)** — 검사 결과(AI 검출)를 사람이 검토·교정해 모델을 다시 학습시키는 **지능화 환류 루프**입니다. 세션을 불러오면 각 프레임에 **AI 검출이 박스로 미리 로드**되고, 작업자가 **과검(FP)** 박스 삭제·**미검(FN)** 누락 결함 박스 추가·**오분류** 클래스 변경으로 교정합니다. 교정 결과와 원본 AI 출력을 IoU로 대조해 **TP/과검/미검/오분류 + 혼동행렬(행=정답, 열=AI예측)**·**정밀도/재현율**을 실시간 표시합니다. 「 데이터셋 환류 + 학습」으로 교정 프레임을 학습 데이터셋(`data/weld_datasets/`)으로 만들어 ③ 학습 탭으로 보내, 현재 모델을 base로 **재학습(transfer learning)** 합니다(미검·과검 사례를 모델이 다시 학습). 교정 내용은 `data/weld_feedback/<세션>/corrections.json` 으로 저장/복원됩니다.



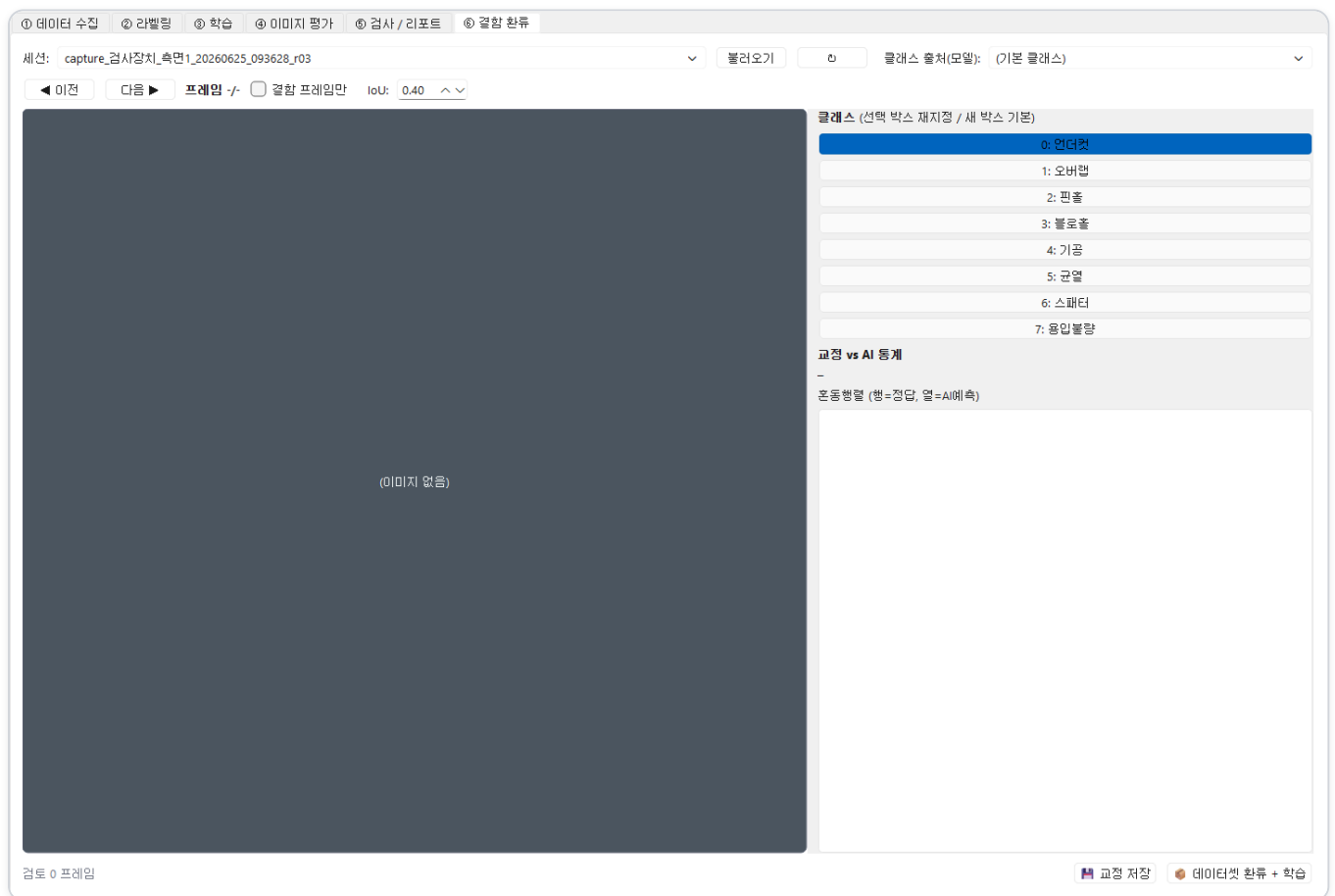
② 라벨링 — 결함을 박스/폴리곤으로 표시하고 클래스를 지정



③ 학습 — YOLO 학습(Precision-Recall-mAP 곡선)



⑤ 검사 / 리포트 — 이동 중 실시간 검출 + 결함 표/맵, ISO 5817 합격판정, 자동 리포트



⑥ 결함 환류(재학습) — AI 검출의 미검/과검 교정 → 혼동행렬 → 데이터셋 환류 + 재학습

이상탐지(정상학습) 기능은 별도 메뉴 **Anomaly Detection**으로 분리되었습니다(아래 참고).

3D 비드 프로파일: 3D 카메라(EpicEye/Photoneo/Zivid)의 organized point cloud로 필릿 비드 단면을 떠 **다리길이·목두께·언더컷 깊이·볼록/오목**을 mm로 측정합니다 (`weld_profile_service`). 측정값은 ISO 5817 룰엔진에 입력되어 치수 기반 합격 판정에 쓰입니다. (필릿 합격 판정의 정석 — 2D 영상만으로는 깊이/치수를 잴 수 없습니다.)

검출 방법 선택 가이드: 분석·판정 방법·검출 매핑은 `docs/weld_inspection_analysis.md` 참고. 요약 — 표면 연속결함=세그멘테이션, 치수결함=3D 프로파일, 미지결함=이상탐지, 최종 OK/NG=ISO 5817 룰엔진. 내부결함은 RT/UT 등 비파괴검사 병행 필요.

스마트폰 카메라로 검사 — 방법 1: 외부 앱(RTSP/웹캠): 아이폰/안드로이드 카메라를 Wi-Fi로 받아 검사할 수 있습니다(드라이버 `generic_opencv`).

1. **네트워크 스트림(PC 드라이버 불필요):** 폰에 RTSP/HTTP-MJPEG 앱 설치 — Android "**IP Webcam**"(`http://<폰 IP>:8080/video`), iOS의 RTSP 카메라 앱(`rtsp://<폰 IP>:8554/live`). 폰과 PC가 **같은 Wi-Fi**여야 합니다.
2. **가상 웹캠:** DroidCam/Iriun/EpocCam을 설치하면 폰이 PC 웹캠으로 잡힙니다 — `device_index` 만 지정.

`config/cameras.yaml` 의 **Phone Camera (Wi-Fi stream)** 항목에서 `url` (또는 `device_index`)을 편집한 뒤, ⑤ 검사/리포트 탭의 **카메라 선택**에서 고르고 Camera ON → 라이브 확인 → 검사하면 됩니다.

스마트폰 카메라로 검사 — 방법 2: 내장 PWA 서버(앱 설치 불필요·AI 판정 내장): 앱 자체 HTTPS 서버를 띄우고 폰 브라우저로 접속해 촬영합니다. 외부 앱이 필요 없고, 폰 화면에서 바로 **OK/NG 판정·결함 박스**를 봅니다.

1. ⑤ 검사/리포트 탭의  **폰 서버** 버튼 → 포트·도메인·(선택)실인증서·**AI 모델**·ISO 5817 등급·두께를 정하고 **서버 시작**.
2. 표시된 **URL(또는 QR)**을 폰 브라우저에서 열기(폰·PC 같은 Wi-Fi). 자체서명 인증서는 1회 '안전하지 않음' 경고를 허용. **실도메인 인증서(cert/key)**를 지정하면 경고 없이 접속됩니다.
3. 폰에서 **카메라 시작** → 후면/플래시/해상도(480~1280)/fps(3~15)를 쓰며 촬영. 「 **자동판정**」 버튼으로 실시간 판정을 ON/OFF(OFF여도 영상은 계속 전송), 「 **1회 판정**」으로 한 장만 판정할 수 있습니다. 판정이 켜지면 서버가 프레임을 AI로 판정해 폰에 OK/NG·박스·사유를 돌려줍니다. **홈 화면에 추가**하면 앱처럼 설치됩니다(PWA).
4. 카메라 선택에서 **Phone Camera (PWA)**(드라이버 `phone_pwa`)를 고르면 폰 영상이 일반 카메라처럼 검사 파이프라인/리포트로도 들어갑니다.
5. **로봇 궤적 없이 검사**: 카메라 바에서 **Phone Camera (PWA)**를 ON 한 뒤 ⑤ 탭의 「수동검사(궤적없이) ▶」 버튼을 누르면, 로봇 이동/궤적 선택 없이 폰(또는 웹캠) 라이브 프레임을 실시간 판정하고 표·오버레이·리포트를 만듭니다. 중지하면 리포트가 자동 저장됩니다.

폰 화면(`cam.fineinspect.xyz`)의 OK/NG 판정 모델은 `config/cameras.yaml` 의

`cam_phone_pwa` → `model` (비우면 가장 최근 학습 모델 자동 선택).

`iso_level` (B/C/D)·`thickness_mm` 로 설정됩니다.  **폰 서버** 다이얼로그에서 모델을 고르거나 「수동검사」를 시작하면 그 모델이 폰 서버에도 적용됩니다.

해상도·fps: 해상도를 낮추고(480/640) fps를 8~15로 올리면 라이브 전송이 더 부드럽습니다. 프레임 전송과 **AI 판정 주기**가 분리돼 있어(`cam_phone_pwa` → `infer_interval_s`, 기본 0.2초) 고평에서도 추론이 병목이 되지 않습니다.

폰 플래시()는 후면 카메라에서 동작합니다. 멀티카메라 폰에서 첫 후면 카메라(초광각/논리 렌즈)에 플래시가 없으면, 앱이 플래시 달린 후면 카메라를 자동으로 찾아 전환합니다.

단, 카카오톡·네이버·라인 등 인앱 브라우저는 플래시 제어를 지원하지 않으니 반드시

Chrome(안드로이드)/Safari(아이폰) 등 일반 브라우저로 여세요. HTTPS는 브라우저 카메라 권한에 필수입니다(자동 자체서명, 인증서는 `data/phone_server/`). 무선 스트림은 지연·압축이 있어 정밀 치수검사보다 **현장 검출·판정 확인**에 적합합니다. 결함의 로봇좌표(mm) 매핑은 폰 카메라 보정이 없으면 제공되지 않습니다.

같은 Wi-Fi가 아니거나 IP가 자주 바뀌면 → **Cloudflare Tunnel**: 다이얼로그의

"**Cloudflare/프록시 뒤**"를 켜고 **공개 URL**(예 `https://cam.fineinspect.xyz`)을 입력하면, 정식 인증서 HTTPS로 어디서나 경고 없이 접속됩니다(로컬은 HTTP). 설정 단계는

`docs/cloudflare_tunnel.md`, 터널 예시는 `config/cloudflared.config.example.yml` 참고.

Camera (2D/3D) 페이지에서도  **폰 서버**로 동일하게 시작할 수 있습니다.

좌표 매핑은 카메라 intrinsic + Hand-Eye 캘리브레이션(Camera 페이지)이 선행되어야 합니다. 2D 카메라는 깊이가 없으므로 용접면을 평면(작업거리)으로 가정합니다.

주석 이미지의 라벨(클래스명+score)은 한글 클래스명도 깨지지 않도록 시스템 CJK 폰트 (Malgun Gothic 등)로 렌더링됩니다.

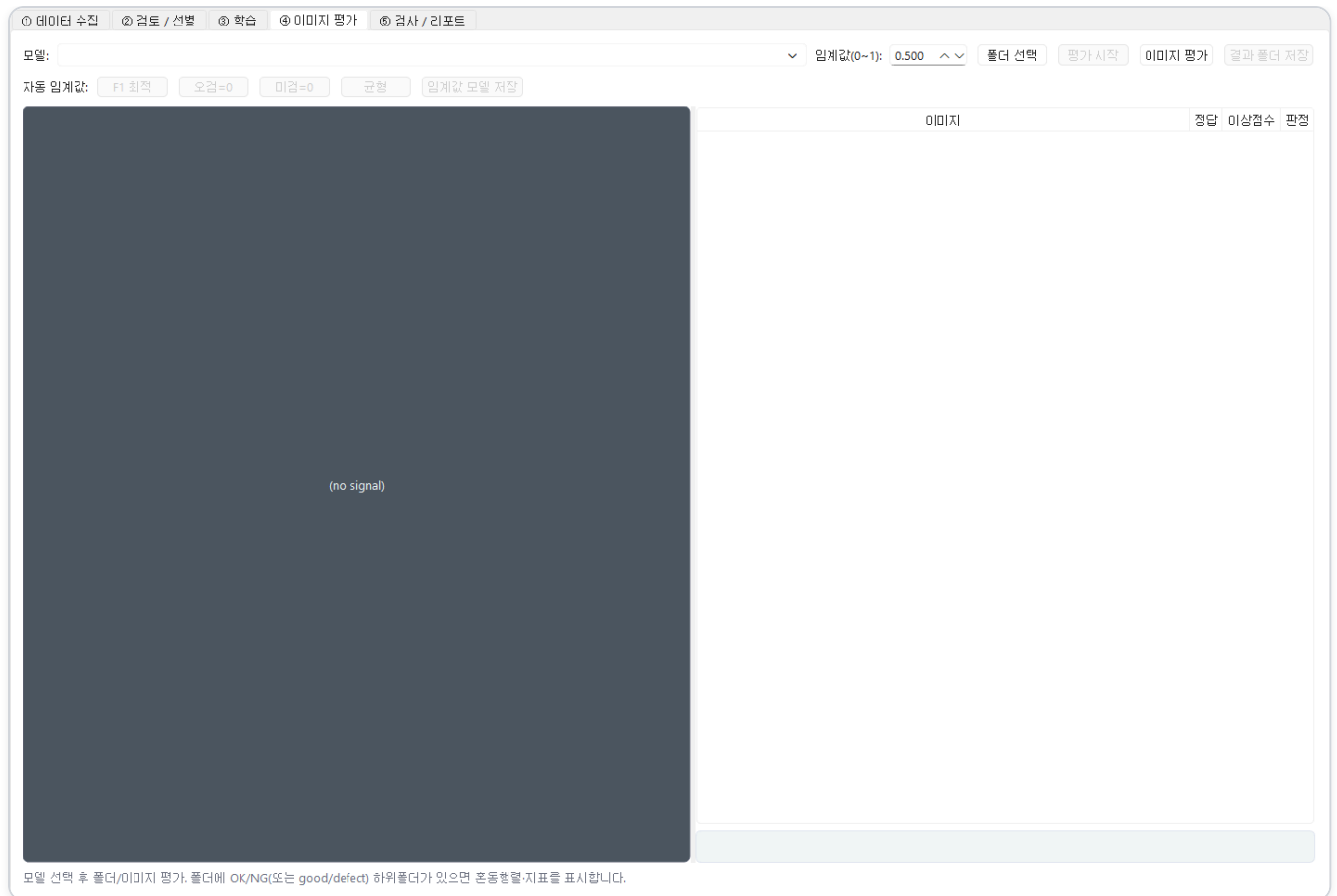
검출이 모두 OK(pass)로 나올 때: 보통 ① 데이터가 너무 적거나(또는 거의 같은 이미지 반복) 또는 ② conf 임계값이 높아 약한 검출이 걸려진 경우입니다. 평가/검사 탭의 **conf**를 낮춰 보세요(약한 검출이 보이면 임계값 문제). 그래도 0이면 **다양한 이미지를 더 많이 라벨링** 해 재학습하세요. (추론 시 로그에 `PROBE(conf>=0.01): ... top confidences=...` 가 남아 신뢰도를 확인할 수 있습니다.)

3D 로봇 뷰에는 용접헤드(링조명+카메라)가 TCP에 부착되어 표시됩니다. 표시할 말단공구는 `config/app.yaml` → `ui.active_tool` (`weld_head` / `gripper_2fg7` / `none`)로 선택합니다.

3D 로봇 뷰에는 용접헤드(링조명+카메라)가 TCP에 부착되어 표시됩니다. 표시할 말단공구는 `config/app.yaml` → `ui.active_tool` (`weld_head` / `gripper_2fg7` / `none`)로 선택합니다.

Anomaly Detection (이상탐지 · 비지도)

결함 라벨 없이 **정상(무결함) 이미지**만으로 **정상 패턴**을 학습하고, 그 분포에서 벗어나는 것을 이상으로 잡습니다. 라벨이 부족하거나 **미지의 결함**을 잡아야 할 때 Object Detection을 보완합니다. 엔진은 두 가지를 ③ 학습에서 선택합니다 — **PatchCore**(v2: 백본 특징 → 코어셋 메모리뱅크 → k-NN 거리)와 **PaDiM**(v1: 위치별 다변량 가우시안 → 마할라노비스 거리). torch(GPU 자동 사용)를 씁니다. (참조: 표준 PaDiM/PatchCore Trainer/Evaluator.)



Anomaly Detection — 정상학습 모델로 이상점수(0~1)·OK/NG·히트맵 평가

- ① 데이터 수집 — 정상 이미지를 데이터셋으로 모읍니다(폴더/파일 추가, 라벨 불필요). `data/ad_datasets/<이름>/good/` 에 저장.
- ② 검토 / 선별 — 잘못 찍힌(정상이 아닌) 이미지를 제외합니다. Object Detection의 "라벨링" 자리지만, 비지도라 박스가 아니라 포함/제외만 정합니다.
- ③ 학습 — 방식(PatchCore/PaDiM)·백본(wide_resnet50_2 등)·이미지크기와, PatchCore는 코어셋비율·proj_dim·k, PaDiM은 채널수(0=자동)를 정하고 정상 데이터셋으로 학습. 진행률/로그 기록, 모델은 `data/ad_models/<이름>/` 에 저장. 임계값은 학습 단계에선 **placeholder(0.5)**입니다 — 비지도(정상만) 학습이라 최적 임계값을 정할 수 없고, ④ 평가에서 산출됩니다. 학습이 진행되는 동안 하단 패널에 이상탐지 개념·파라미터·데이터 팁이 자동 스크롤로 표시됩니다.
- ④ 이미지 평가 — 「폴더 선택」 → 대표 이미지 표시 → 「평가 시작」 순서로 평가합니다(한 장씩 즉시 표시). 이상점수(0~1 정규화)·OK/NG(점수≥임계값=NG)를 표로, 행 선택 시 히트맵을 오버레이. 임계값: 라벨(OK/NG)이 있으면 평가 완료 시 **F1 최적 임계값**을 자동 산출·적용하며, 「임계값 모델 저장」으로 모델에 저장하면 ⑤ 검사가 그 값을 씁니다. 화면에서 수동 조정도 가능. 하위폴더로 정답 자동 인식 — `OK` / `good` 폴더는 정상(OK), 그 외 모든 폴더명은 실제 결함유형(scratch-dent-기공 등 = NG)으로 간주합니다. 정답이 있으면 표의 "정답" 열에 결함유형명을 표시하고 혼동행렬과 지표(정확도·정밀도·재현율·특이도·F1·FPR·FNR·AUROC)를 보여주며, 자동 임계값(F1 최적/오검=0/미검=0/균형) 버튼을 제공합니다. (예: `eval/OK/*.jpg`, `eval/scratch/*.jpg`) 히트맵은 고정 0~1 스케일(jet-α0.4·vmin0 vmax1)로 그려 정상은 차분하게, 결함만 붉게 표시됩니다. 「결과 폴더 저장」은 참조 프로그램과 동일 형식으로 출력합니다 — 결과이미지(원본 | 히트맵, `<Class>_<파일>.png`) · `evaluation_results.csv` · `configuration_info.csv` · 혼동행렬/ROC/PR/F1/히스토그램 PNG(`data/ad_reports/`).

- ⑤ **검사 / 리포트** — 카메라(웹캠/폰 PWA 포함)를 켜고 **실시간 검사** → OK/NG·히트맵 표시, NG 프레임 이미지와 요약 리포트를 `data/ad_reports/<세션>/`에 저장.

Object Detection vs Anomaly Detection: 결함 종류를 알고 라벨이 충분하면 **Object Detection**(정밀·분류), 라벨이 적거나 미지 결함 위주면 **Anomaly Detection**(정상학습)을 쓰세요. 두 방식을 함께 쓰면 상호 보완됩니다.

최초 학습 시 백본의 ImageNet 사전학습 가중치가 **인터넷에서 1회 다운로드**됩니다(torch 캐시). 오프라인 PC라면 가중치 캐시를 미리 준비하세요. 정밀 학습/추론은 **GPU(CUDA)** 권장입니다.

Hole Inspection (천공 검사 · Rule Based)

규칙 기반 천공(구멍) 검사입니다. Trajectory Capture로 **이동 촬영해 저장한 세션**을 분석해 구멍의 **개수·직경·위치·누락**을 판정합니다. 메뉴는 **QUALITY** 하단 「**RULE BASED → Hole Inspection**」. 학습(모델)이 필요 없는 **고전 비전 + 자기참조(격자 자기적합)** 방식이라, 기준 도면 없이도 검출된 구멍 집단 스스로를 기준으로 이상치를 찾습니다.

자기참조(Self-reference) 판정 — 골든/CAD 기준이 없으므로, 검출된 구멍들에 **규칙 격자(lattice)**를 피팅해 그 이상적 격자를 기준으로 삼습니다: **개수**=채워진 격자 노드, **누락**=비어있는 노드, **직경**=집단 median에서 벗어난 정도, **위치**=격자 예상위치 대비 잔차. (곡면 패넬은 프레임을 **국소 타일**로 나눠 타일마다 격자를 맞춥니다.)

사용 순서

- ① **세션 선택** — 저장된 캡처 세션을 드롭다운에서 고릅니다(🔄 새로고침). 폴더 내 **전 프레임**을 검사합니다.
- ② **검출 옵션** — **px/mm**(0=세션 stitch 정보에서 자동 인식), **기대 Ø(mm)**(검출 스케일), **타일**(곡면 대응 국소 격자, 기본 3×2), **검출민감도**(Hough 임계값 — 낮을수록 더 많이 검출).
- ③ **판정 기준(OK/NG) 설정** — **직경공차(mm)·위치공차(mm)**로 "무엇을 결함으로 볼지", **최대허용 결함**으로 "몇 개까지 OK로 볼지", **기대개수**(전역 구멍 최소수), **누락도 불량처리** 여부를 정합니다. 판정프레임의 후보결함 합이 **최대허용결함**을 넘으면 **NG**.
- ④ 🎯 **분석 실행** — 백그라운드로 분석(수 초). 완료 시 **판정 बै지**(PASS/FAIL/보류), 요약(px/mm·판정프레임·직경·전역 구멍 수·후보결함·소요시간)이 표시됩니다.

결과 보기

- **전체(stitched) 뷰** — 프레임 콤보 맨 위 「 전체 (stitched)」에서 **패널 전체 이미지에 검사결과 마킹**을 얹어 한눈에 봅니다(개별 프레임도 이전/다음으로 열람).
- **마킹 색상** — 초록=정상 구멍, **자홍 X=누락 의심(검토 후보)**, 주황=직경이상, 빨강=위치이상.
- **결과 마킹 표시 On/Off** — 끄면 **원본 이미지**만 표시해 구멍을 마킹 없이 상세 확인, 켜면 판정 마킹을 겹쳐 표시. **배율**(맞춤/100%/150%/200%)+스크롤로 확대 관찰.
- **결과 폴더 열기** — 세션 폴더의 `hole_inspection/` 에 프레임별 `overlay_*.png` · `overlay_stitched.png` · `holes.csv` (구멍별 좌표·직경·잔차·상태) · `hole_report.json` 이 생성됩니다.

현재 한계(정직) — 저대비·직물 텍스처 패널에서 고전 검출기의 **검출율이 100%가 아니므로**, "누락"에는 **검출 공백**이 섞일 수 있습니다. 그래서 기본값은 **누락=자동 NG 아님(검토 후보로만 표시)**이며, verdict는 직경/위치 이상치 위주로 냅니다. 검출율이 낮은 프레임은 「**판정 보류**」로 정직하게 표시됩니다. 검출민감도를 낮추면 더 많이 잡지만 과검출도 늘 수 있습니다.

CAD 기준 검사(v1.2 신규 · E1) — spec 행의 「 CAD 파일...」에서 **2D DXF(원 엔티티)** 또는 **좌표표 CSV(x,y[,d])**를 지정하면, 검출된 구멍을 도면과 **정합해 절대 개수·누락·위치편차**를 판정합니다. CAD 기준이 있으면 **누락이 신뢰 가능**해져 「누락도 불량처리」를 켤 수 있습니다(정합 RMS를 함께 표시). 결과 뷰어의 「 CAD 정합 맵」에서 매칭●/누락×/잉여+/위치이탈○을 도식으로 확인합니다. 완전 대칭 격자는 대칭군 동치까지만 정합되므로 비대칭 특징이 있는 실부품에 적합합니다.

Recipe & Job Runner (레시피 & 잡 러너)

v1.2

레시피(Recipe)는 제품 1종의 전체 검사 셋업 스냅샷입니다 — 로봇·궤적·카메라와 검사 엔진 조합(룰/YOLO/AD/골든), 판정 정책을 하나로 묶습니다. 제품 교체는 **레시피 선택 한 번**으로 끝납니다. **Job Runner**(RUN 그룹 첫 화면)가 레시피를 **반복 사이클**로 실행합니다.

사용 순서

- ① **레시피 만들기** — 「+ 새 레시피」 → 로봇·궤적·카메라 선택 + **검사 엔진 스택**(체크한 엔진만 사용) + 판정 정책 →  저장. `data/recipes/<이름>.yaml` 에 저장됩니다.
- ② **실행** — 레시피와 **사이클 수**, (선택) **시리얼**(바코드 스캔/입력)을 정하고 ▶ **시작**. 궤적을 따라 촬영하며 프레임마다 엔진이 판정하고, 사이클 단위로 **OK/NG**가 집계됩니다.
- ③ **모니터링** — **대형 판정등**(원거리 시인성)·OK/NG·불량률·텍트, 라이브 프레임(엔진별 중간결과), **최근 NG 갤러리**(더블클릭=상세), 사이클 로그.

3-엔진 통합 검사(E0) — 플랫폼 기본 문법 — 모든 제품 검사를 **Rule(정량 스펙) + Object Detection(알려진 결함) + Anomaly(미지 결함)** 세 엔진의 조합으로 구성합니다. 신제품 초기엔 **AD 단독**(정상 몇 장 또는 Zero-Shot 0장)으로 즉시 가동하고, 데이터가 쌓이면 OD·Rule을 점진 추가합니다. 각 엔진은 공통 결과 스키마로 동작하므로 Job Runner·대시보드·결함맵·환류학습이 엔진 구분 없이 작동합니다.

결과 기록(트레이서빌리티 · C2) — 사이클마다 시리얼·레시피·시각·판정·택트·엔진 지표 ·NG 이미지 경로가 `data/results.db` (SQLite)에 1행씩 기록됩니다. 「Factory I/O」 화면에서 시리얼/레시피로 조회하고, 「Reports / SPC」에서 통계 분석합니다.

09-21 Job Runner 재생 개선 — ① 궤적 waypoint 의 **그리퍼 동작**(포즈/파지/해제/⑤ 파지 프리셋)이 Job 실행에서도 도달 후 실행됩니다(Teaching 재생기와 같은 함수). 예전에는 Job 경로(촬영 파이프라인)에 이 단계가 없어 로봇만 움직이고 그리퍼는 가만히 있었습니다. ② 같은 자세에 그리퍼 동작만 다른 waypoint 는 이동·완료 대기를 건너뛰고, **촬영도 하지 않습니다**(카메라는 로봇이 실제로 이동한 waypoint 에서만 동작, 첫 waypoint 는 항상 촬영). ③ 레시피의 「조명 강도 적용」을 끄면 그 실행 동안 촬영마다 조명 컨트롤러에 접속하지 않습니다(컨트롤러가 꺼져 있을 때 "조명 연결 실패 ... timed out" 경고와 촬영 지연이 사라짐). 켜면 예전처럼 레시피 강도로 촬영합니다.

Operator · Engineer 이중 모드 v1.2

화면 최상단 전역 상태바에 **E-STOP**(연결된 모든 로봇 일괄 정지)·로봇/카메라/조명/PLC 상태·현재 레시피·**모드 전환** 버튼이 상시 표시됩니다.

- **Operator 모드** — **RUN** 그룹만 보이고 레시피 편집이 잠깁니다(현장 오조작 방지). 큰 버튼·판정 중심.
- **Engineer 모드** — 전체 메뉴 접근. Operator→Engineer 전환에는 **PIN**이 필요합니다 (`config/app.yaml` 의 `ui.engineer_pin` , 기본 `0000` — 배포 시 변경 권장).

Few-Shot / Zero-Shot 이상탐지(AD) v1.2

Anomaly Detection 화면에서 **신제품을 즉시 도입**하는 두 가지 방식이 추가되었습니다.

- **Few-Shot(E6-1/2)** — 학습 화면에서 「**Few-Shot**」 체크. 정상 **1~8장**만으로 회전/플립 증강 + 코어셋 1.0 + **라벨 없는 자동 임계값**(통계 분위수)으로 모델을 만듭니다. 백본을 **dinov2_vits14**(AnomalyDINO 방식)로 선택하면 1-shot에서도 높은 정확도를 냅니다.

- **Zero-Shot(E6-3)** — 정상 이미지 **0**장으로 즉시 검사. 레시피의 AD 모델 칸에 `zeroshot:<영문 객체명>` (예: `zeroshot:metal panel`)을 입력하면, CLIP이 "정상 vs 결함" 텍스트 프롬프트와의 유사도로 이상확률을 계산합니다.

오프라인 현장 유의 — dinov2-CLIP 백본 가중치는 최초 1회 인터넷 다운로드가 필요합니다. 인터넷이 없는 현장 PC는 가중치 캐시(`~/.cache/huggingface`)를 미리 배치하세요.

골든샘플 비교 검사 v1.2

양품 이미지 1~N장만으로 즉시 적용하는 **차영상** 검사입니다(레시피의  **Golden Sample** 엔진). 등록된 양품들을 정렬해 픽셀별 정상 통계(평균±σ)를 만들고, 검사 이미지를 정렬 후 **z-맵**으로 비교해 결함 영역을 찾습니다. 라벨·학습이 필요 없어 **반복 촬영 자세가 일정한 Job Runner**와 궁합이 좋습니다. 레시피 편집기의 골든 행에서 「등록...」으로 양품 파일을 추가하고 **z임계**(낮을수록 민감)를 정합니다.

판정 정책 — 3-엔진 결합 v1.2

레시피 편집기 하단에서 엔진별 결과를 하나의 OK/NG로 결합하는 방식을 정합니다.

- **하나라도 NG(기본)** / **다수결**(판정 엔진 과반 NG).
- **가중 투표(E3)** — 엔진별 가중치 합이 임계 이상이면 NG. 예: rule 1.0(단독 NG 가능), yolo·ad 0.6(둘 다일 때만).
- **사용자식** — 불리언 식으로 자유 결합. 변수: `rule.defects`, `yolo.count`, `ad.score`, `golden.regions`, `merged.defects` / `merged.multi` (엔진 간 **같은 위치 검출을 병합한** 결함 수 / 2개 이상 엔진이 동시에 잡은 고신뢰 결함 수). 예: `rule.defects>=1 or ad.score>0.8 or merged.multi>=1`.
- **중복병합 반경(mm)** — 여러 엔진이 같은 결함을 잡아도 1건으로 셉니다.

캘리브레이션 자동화 위저드 v1.2

Camera 화면의 Hand-Eye 위저드에 「 **원클릭 캘리브레이션**」이 추가되었습니다: 로봇이 보드 주위 **9**자세를 자동 이동·촬영 → 계산 → **품질 리포트**(샘플별 재구성 오차 mm/°, 이상치 강조, 등급) → 저장까지 한번에 진행합니다. 「 **보정 확인**」은 고정 보드를 3자세에서 관측해 캘리브레이션의 **종단 오차**를 mm로 검증합니다. 픽 정확도는 hand-eye 오차가 좌우하므로, 픽 보정(Pick Guidance)보다 먼저 이 위저드로 품질을 확보하세요.

Pick Guidance — 2D 위치보정 픽 v1.2

제품이 놓인 위치·각도 편차를 카메라로 측정해 로봇 픽 자세를 보정합니다(비전 가이드 로봇, VGR). GUIDE 그룹.

1. ① **기준 촬영** — 골든(기준) 자세에서 촬영. px/mm은 캘리브레이션 working scale에서 자동 입력됩니다. ROI로 부품 영역을 지정합니다.
2. ② **픽 자세 지정** — 로봇을 실제 픽 자세로 옮기고 기록 → 이름 입력 후 저장.
3. **측정 촬영** — 새 제품을 촬영하면 $\Delta X \cdot \Delta Y \cdot \Delta R_z$ 와 신뢰도가 표시됩니다(기준 **+**/검출 **○**/화살표 오버레이).
4. **테스트 픽** — 보정 자세로 이동. 한계($\pm \text{mm}/\pm^\circ$) 초과 보정은 실행이 거부되어(오적재·오매칭 방어) 로봇이 잘못된 위치로 가지 않습니다.

정밀 서보잉(Visual Servoing) v1.2

Pick Guidance의 「 정밀 서보잉」은 목표 특징이 화면 중앙에 오도록 **페루프로 미세보정**합니다(커넥터 삽입·정밀 안착용). 이미지 오차를 협동로봇 **저속 구간**에서 작은 상대이동으로 줄이며, **수렴/정체/추적실패 시 안전하게 중단**합니다. 매 스텝 이동량이 클램프되어 잘못된 추적이 큰 동작으로 이어지지 않습니다.

Bin Picking(빈피킹) v1.2

3D 카메라로 **빈에 쌓인 부품**을 촬영해 파지 후보를 계산합니다. GUIDE 그룹.

1. **촬영/PLY 로드** → 「**파지 후보 계산**」 — 바닥 평면을 제거하고 부품을 클러스터링해 **클러스터별 최상단 수직 파지점**을 만듭니다.
2. **순위표** — 점수(높이·지지·여유), 높이, **도달 가능 여부**, 비교(하강 충돌 위험 등). 3D 뷰에 파지점이 축으로 표시됩니다.
3. **테스트 파지** — 최상위 후보로 수직 접근(로봇 연결·Lock 필요).

09-21 개선 — ① 촬영/PLY 로드 후 3D 뷰가 **CAM(카메라 눈)** 시점으로 시작합니다(📷

·Camera 화면과 동일). ② **인식 ROI**: Camera ▶ 3D ROI / 📷 P&P 와 같은 프리셋을 골라 상자 안 점만으로 후보를 계산합니다(기하 = 점 필터, 학습 모델 = 이미지·점군 크롭). 후보가 수십 개 나올 때 ROI 로 줄이세요. ③ 순위표에 **X/Y/Z(mm)·접근 기울기(°)** 열이 추가됐고, 표 제목에 좌표계(로봇 베이스 / 카메라)가 표시됩니다. 도달 여부는 베이스 변환이 있을 때만 판정(없으면 '?'). ④ 베이스 변환은 📷 엔진과 같은 단일 출처(핸드아이 모드 자동: eye-to-hand 고정 카메라 = 저장된 T_base_cam, eye-in-hand = 현재 TCP × T_tcp_cam)를 씁니다. 예전에는 eye-in-hand 값만 찾아 고정 카메라에서 "base 좌표 변환이 없습니다" 가 떴습니다. 테스트 파지가 막히면 안내문에 **원인**(활성 로봇 없음 / 카메라 미선택 / 해당 로봇×카메라 캘리브레이션 없음 / TCP 자세 없음)이 표시됩니다. ⑤ **테스트 파지**는 표에서 선택한 후보(없으면 최상위)의 40 mm 위로 **현재 TCP 방향을 유지한 채** 직선 이동합니다. 예전에는 자세를 (180,0,0)°로 고정해 손목을 베이스 축에 맞춰 크게 돌리는 바람에 "이상한 곳으로 가는" 동작이 됐습니다. 확인 대화에 현재 TCP·목표·이동 거리가 표시되고, 이동 중에는 **|| 일시정지**(다시 누르면 재개) · **■ 정지**(task stop) 버튼이 살아 있습니다(이동은 백그라운드 대기라 화면이 멈추지 않음). 수직 진입 방향·손끝 정렬까지 포함한 실제 파지는 📷 3D Pick&Place 에서 합니다.

단계적 구현 — 1단계는 **구조화 빈**(평면 분리·최상단·수직 파지)이며, 도달성/하강 충돌은 **간이검사**(베이스 워크스페이스 shell + footprint 간섭)입니다. 2단계 난적재는 **6-DoF 모델 정합**(아래 3D 포즈 추정)과 결합합니다.

3D 포즈 추정(6-DoF)

v1.2

포인트클라우드를 기준 모델(골든 캡처/CAD 샘플점)과 **정합(FPFH RANSAC + ICP)**해 부품의 **6자유도** 자세를 추정하고, hand-eye 체인으로 **베이스 좌표 파지 자세**를 만듭니다. 신뢰도(fitness)가 낮으면 로봇을 구동하지 않습니다. Zivid/Photoneo SDK의 매칭 결과를 직접 받는 병행 경로도 지원합니다.

Safety Vision (안전 비전 · 보조)

v1.2

⚠ **보조 감시(정보성)일 뿐 인증 안전장치가 아닙니다.** 실제 안전은 **안전PLC·안전스캐너**로 별도 구성해야 하며, E-STOP·모션은 항상 로봇 안전계가 우선입니다.

광각 카메라로 작업영역의 **작업자를 감지**(YOLO 사람 클래스)하고, 최근접 거리를 추정해 **ISO/TS 15066(SSM)** 기준의 권고 속도를 표시합니다(zone: CLEAR/SLOW/STOP). 「**속도 자동 적용**」을 켜면 활성

로봇에 감속/일시정지를 적용합니다. 거리 추정 = 화면의 기준 신장 픽셀 / 그때 거리 두 값으로 보정합니다.

Factory I/O (공장연동 · PLC / DB) v1.2

SYSTEM 그룹. **Modbus TCP** 서버를 앱이 호스팅하고(앱=슬레이브, PLC=마스터), 레지스터 맵으로 신호를 주고받습니다.

- **PLC 사용법** — `trigger` (1번지)에 1을 쓰면 선택된 레시피로 **Job 자동 시작**, `heartbeat` (0번지)는 1초마다 증가(워치독), `result_verdict` (3번지) 1=OK/2=NG, OK/NG 누계도 발행됩니다.
- **레지스터 맵은 설정** — `config/factory_io.yaml` 에서 포트(기본 5020, 표준 502는 관리자 권한)·주소를 바꿉니다(하드코딩 아님).
- **결과 DB 조회** — 같은 화면에서 레시피/시리얼로 트레이서빌리티 결과를 검색합니다.

Reports / SPC (리포트 · 통계공정관리) v1.2

Job Runner가 쌓은 측정치(구멍 직경 평균, AD 점수 등)를 **통계공정관리**로 분석합니다. 레시피·측정치를 고르고 **부분군 크기·규격(LSL/USL)**을 정한 뒤 「 분석」: **X-R 관리도**(CL/UCL/LCL·규격선), **Cp/Cpk**(1.33/1.0 기준 색상), **경향 경보**(3σ 이탈, 연속 9점 한쪽, 연속 7점 상승/하강)를 제공합니다.

Digital Twin (디지털 트윈) v1.2

-  **가상 카메라(F2)** — 현재 TCP+hand-eye 카메라 자세 기준으로 **시야 절두체·바닥 커버리지**를 3D로 미리 보여 **촬영 사각을 예방**합니다(FOV·작동거리 입력).
-  **리플레이(F3)** — 저장 세션의 **이미지+포즈 프레임**을 시간순으로 재생하는 "블랙박스"입니다. 슬라이더로 스크립하며 문제를 분석합니다.

Model History (모델 이력/재학습 · 액티브 러닝) v1.2

INSPECT 그룹. **미검/과검 환류**를 자동 재학습으로 잇습니다.

- **재학습 큐** — Object Detection 환류(교정)가 쌓인 세션 목록과 **미검(fn)/과검(fp)** 통계를 보여줍니다.
- **데이터셋 생성** — 큐의 교정을 하나의 전이학습 데이터셋으로 취합 → Object Detection 학습 탭에서 재학습.
- **야간 자동 재학습** — 지정 시각에 취합→학습→**A/B 검증**(F1 개선 & recall 미퇴행 시)→**승격**을 자동 수행합니다.
- **모델 버전·성능 이력** — 패밀리별 버전·지표·승격 상태 표. 원하는 버전으로 **승격/롤백**합니다.

헤드리스 러너 · 웹 모니터 v1.2

GUI 없이 레시피를 실행하는 **콘솔/서비스** 모드입니다(라인 PC·크론).

```
fineai-robodk-run --recipe 콧핏상단_v3 --cycles 100 --robot rb1 [--sim]
                    [--web-port 8770] [--no-web]
```

읽기 전용 웹 모니터 — GUI Job Runner의 「 웹 모니터」 또는 헤드리스 실행 시 자동으로 켜지며, 브라우저에서 `http://<PC IP>:8770/` 로 **OK/NG·사이클·택트·판정**등을 원격 조회합니다(제어 불가라 LAN 노출 안전).

멀티 로봇 벤더 v1.2

로봇 드라이버가 **벤더 추상화(RobotDriver)**로 정리되어, Robots 등록 품의 **Vendor** 선택으로 드라이버를 지정합니다((자동)은 Type에서 추론). 현재 실장비 드라이버는 **rainbow(RB 시리즈)**이며, doosan/universal_robots/techman 키가 예약되어 향후 드라이버 추가 시 호출부 변경 없이 확장됩니다.

폰 원격접속 — cam.fineinspect.xyz (Cloudflare Tunnel)

 **폰 서버**를 같은 Wi-Fi가 아닌 곳에서도, 공인 IP/포트포워딩 없이, 정식 인증서 HTTPS로 열기 위해 **Cloudflare Tunnel**을 씁니다. 폰 브라우저는 항상 `https://cam.fineinspect.xyz` 로 접속하고, Cloudflare 엣지가 TLS를 종단해 PC의 로컬 서버(평문 HTTP)로 전달합니다.

```
폰 브라우저 —HTTPS—> Cloudflare 엣지(cam.fineinspect.xyz)
                        | (cloudflared 아웃바운드 터널, 이름: fineinspect)
                        ▼
PC: cloudflared(서비스) —HTTP—> 127.0.0.1:8443 (fineAi 폰 서버)
```

구성요소 (이 PC에 이미 설정됨)

항목	값 / 위치
공개 주소	<code>https://cam.fineinspect.xyz</code> (DNS는 Cloudflare가 관리)
터널 이름	<code>fineinspect</code> (이 이름의 터널이 <code>cam.fineinspect.xyz</code> 로 라우팅됨)
cloudflared	Windows 서비스로 등록 — 부팅 시 자동 실행(StartType=Automatic)
터널 설정	<code>C:\Users\W<사용자>\W.cloudflared\config.yml</code> — ingress: <code>cam.fineinspect.xyz</code> → <code>http://127.0.0.1:8443</code>
자격증명	<code>C:\Users\W<사용자>\W.cloudflared\W<TUNNEL-UUID>.json</code> + <code>cert.pem</code>
앱 폰 서버	로컬 <code>127.0.0.1:8443</code> (HTTP) — 앱의  폰 서버에서 시작

접속이 되게 하려면 (체크리스트)

1. **cloudflared 서비스 실행 중** — 부팅 시 자동 시작됩니다. 점검: `Get-Service cloudflared` → **Running**. 터널이 `fineinspect` 를 실행하는지 점검: `cloudflared tunnel info fineinspect` → **CONNECTOR가 1개 이상이면 정상**.
2. **앱의 폰 서버 시작** — 앱에서 **Camera (2D/3D)** 또는 **Weld ⑤ 검사/리포트** →  폰 서버 → "Cloudflare/프록시 뒤" 체크, 포트 **8443**, 공개 URL `https://cam.fineinspect.xyz` → 서버 시작. (`config/cameras.yaml` 의 `cam_phone_pwa` 에 `public_url` 이 들어 있어, 카메라를 열면 같은 설정으로 자동 시작됩니다.)
3. 폰 브라우저에서 `https://cam.fineinspect.xyz` (또는 다이얼로그의 QR) 열기 → **카메라 시작**.

빠른 자가진단(PC PowerShell): `(Invoke-WebRequest https://cam.fineinspect.xyz/ -UseBasicParsing).StatusCode` → **200**=정상 · **502**=터널은 정상이나 앱 폰 서버 미실행(2번 수행) · **530**=cloudflared가 `fineinspect` 터널을 안 돌리는 중(아래 "재부팅 후" 참고).

재부팅 후 (자동 복구)

cloudflared가 서비스로 등록되어 있으므로 PC를 껐다 켜도 터널은 자동으로 다시 연결됩니다. 사용자는 **앱의 폰 서버만 다시 시작**(체크리스트 2번)하면 됩니다.

재부팅 후 530 오류가 나면 — cloudflared 서비스가 엉뚱한 터널(토큰/다른 이름)을 실행 중일 수 있습니다. 서비스가 `fineinspect` 터널을 실행하도록 ImagePath를 고친 뒤 재시작하세요(관리자 PowerShell):

```
$reg = "HKLM:SYSTEMWCurrentControlSetWServicesWcloudflared"
$bin = '"C:\Program Files (x86)\cloudflared\cloudflared.exe" tunnel --config "C:\Users\<사용자>\AppData\Local\cloudflared\config.yml"'
Stop-Service cloudflared -Force
Set-ItemProperty -Path $reg -Name ImagePath -Value $bin
Start-Service cloudflared
cloudflared tunnel info fineinspect # CONNECTOR가 보이면 정상
```

(config.yml의 ingress가 `cam.fineinspect.xyz` → `http://127.0.0.1:8443` 인지도 확인.)

다른 노트북에서 같은 주소(cam.fineinspect.xyz)로 쓰기

현장을 옮겨 다른 노트북에서 fineAi를 돌려도 폰이 동일하게 `https://cam.fineinspect.xyz` 로 접속하게 하려면, 그 노트북에서도 같은 `fineinspect` 터널을 실행하면 됩니다(터널 자격증명만 옮기면 됨).

1. **cloudflared 설치** (새 노트북, 1회): `winget install --id Cloudflare.cloudflared` 또는 Cloudflare의 Windows `.msi`. 확인: `cloudflared --version`.
2. **터널 자격증명 복사** — 기존 PC의 `C:\Users\<사용자>\W.cloudflared\W` 폴더에서 아래 3개 파일을 새 노트북의 같은 경로(`%USERPROFILE%\W.cloudflared\W`)로 복사:
 - `config.yml` (ingress: `cam.fineinspect.xyz` → `http://127.0.0.1:8443`)
 - `<TUNNEL-UUID>.json` (예: `c93ce059-...\.json` — 터널 자격증명)
 - `cert.pem`

※ `config.yml`의 `credentials-file` 경로(사용자명)를 새 노트북에 맞게 수정하세요.

3. **서비스로 등록**(부팅 자동) — 새 노트북 관리자 PowerShell:

```
cloudflared --config "C:\Users\<사용자>\W.cloudflared\config.yml" service install
Get-Service cloudflared # Running 확인
```

또는 테스트로 한 번만: `cloudflared --config "...Wconfig.yml" tunnel run fineinspect`.

4. 새 노트북에서 fineAi 실행 →  **폰 서버**(프록시 뒤.포트 8443·공개 URL `https://cam.fineinspect.xyz`) 시작 → 폰으로 접속.

한 번에 한 노트북만: `cam.fineinspect.xyz` 하나의 주소는 동시에 여러 PC가 같은 터널을 돌리면 요청이 여러 PC로 분산되어 엉뚱한 노트북으로 갈 수 있습니다. 옮겨 쓸 때는 **이전 노트북의 cloudflared 서비스를 정지**(`Stop-Service cloudflared`)하고 새 노트북에서만 켜세요. 두 현장을 동시에 운용하려면 노트북마다 별도 hostname(예 `cam2.fineinspect.xyz`)을 쓰는 것을 권장합니다 — 절차는 `docs/cloudflare_tunnel.md` 참고.

처음부터 새로 터널을 만들거나, DNS 라우트·Zero Trust 인증(이메일/PIN) 등 상세 설정은 `docs/cloudflare_tunnel.md` 와 `config/cloudflared.config.example.yml` 를 참고하세요.

설치 / 실행

패키지(onedir) 폴더 구조:

항목	설명
<code>fineAi_RoboDK.exe</code>	실행 파일 (더블클릭).
<code>_internal/</code>	번들(라이브러리, assets, logo, docs).
<code>config/ · data/ · logs/</code>	실행 파일 옆에 생성되는 쓰기 영역(설정·캡처·로그).

배포는 `distWfineAi_RoboDKW` 폴더를 통째로 복사하면 됩니다.

로그

- 앱 로그 — `logs/fineai_robodk_YYYY-MM-DD.log` (일자별).
- 동작 로그(operations) — `logs/operations_YYYY-MM-DD.log` — 카메라/로봇/자동화 동작을 `시각`
`| Level | Source | 메시지` 형식으로 기록(EpicEye/Camera/Robot/Capture/EpicPro/PA-Engine).
일자별 자동 rotation + 14일 보관.

Basler 카메라는 대상 PC에 **pylon 런타임**이 필요할 수 있습니다. EpicEye는 카메라와 같은 네트워크(같은 서브넷)에 PC가 있어야 검색/연결됩니다. **한 카메라는 한 번에 한 PC에서만** 라이브/캡처하세요(동시 접근 시 캡처 실패).

문제 해결

- (PA Engine) “새 그래프”가 반응 없어 보임 — 누르면 `CameraSource` 시작 노드가 자동 생성됩니다. 노드가 보이지 않으면 **Fit**으로 화면을 맞추세요.
- (PA Engine) 노드가 추가되지 않음 — 팔레트의 그룹 헤더가 아니라 하위 연산자를 선택해야 합니다. 더블클릭 대신 선택 후 **+** 추가도 사용하세요.
- (PA Engine) 연결이 안 됨 — 출력(우측)→입력(좌측) 방향이어야 하고 포트 타입이 일치해야 합니다. 드래그 중 선이 빨강이면 타입 불일치입니다(예: Image2D를 PointCloud 입력에 연결 불가).
- (PA Engine) 검증 실패 — 필수 입력 미연결/순환/타입 불일치 메시지를 확인하세요. 사이클(순환)은 허용되지 않습니다.

- **(PA Engine) 연결선만 삭제** — 연결선을 클릭(흰색·굵게 강조) 후 **Del.** 전체 보기는 **FIT** 버튼.
- **(PA Engine) CameraSource “프레임 없음”** — 상단 **카메라** 콤보에서 카메라를 선택하세요(자동 Open·재시도). 그래도 실패하면 다른 PC가 카메라를 점유 중인지 확인.
- **(캘리브레이션) 보드 검출 0** — 보드 종류(체커/원형)와 cols×rows·간격이 실제 보드와 정확히 일치해야 합니다. 🔍 **검출 확인**으로 맞을 때까지 조정(가로/세로 전치도 시도). 조명·반사·보드 전체가 프레임에 들어오는지 확인.
- **(Epic Pro) “hand-eye 캘리브레이션 없음” 오류** — Camera 메뉴에서 해당 (로봇,카메라) 쌍의 Hand-Eye 캘리브레이션을 먼저 수행하세요(`config/hand_eye.yaml`).
- **(Epic Pro) 동작 없이 “포즈만 산출”** — 로봇 미연결/Lock 미보유 또는 캘리브레이션 없음(안전 동작). Robots에서 연결·Lock 후 재실행하세요.
- **(Epic Pro) no_detection으로 바로 종료** — CropBox 작업영역, 군집 `eps_mm` / `min_samples` , 평면 제거 임계가 장면과 맞는지 PA Engine에서 ▶ **선택까지**로 단계별 확인하세요.
- **EpicEye가 안 보임** — IP/전원/네트워크 확인. 검색은 멀티캐스트(UDP 5665–5668)를 사용하므로 동일 서브넷이어야 합니다.
- **로봇 모션이 외부 트래픽으로 중단** — 로봇 제어 포트(5000)로 가는 HTTP/포트스캔을 방화벽으로 차단 권장. (앱은 외부 노이즈를 무시하도록 처리되어 있습니다.)
- **2D보다 3D 이미지가 더 넓어 보임** — 해상도는 동일(1920×1200, 픽셀 정렬). 2D 노출(ExpTime2D)이 부족하면 가장자리가 어둡게 빠져 그렇게 보입니다. 노출을 올리세요.
- **로그** — `logs/` 폴더의 일별 로그 파일 확인.