Industrial Design Extraction Swarm

A multi-lens, multi-agent computer vision architecture to extract manufacturable physical properties from 2D industrial design documents.

Version 1.0 | 18 February 2026

Authoring context: synthesized from a design conversation about building a 'Vision Swarm' that behaves like a cranky senior ID/mech review team (evidence-first, physics-validated, reproducible).

Contents

1. Executive summary

This document specifies an Industrial Design (ID) Extraction Swarm: a vision-layer pipeline that transforms 2D industrial design artifacts (concept sketches, patent drawings, exploded views, orthographic projections, photos/renders, and mixed PDFs) into a structured, evidence-backed model of a manufacturable object.

The core idea is a Mixture-of-Experts approach applied to computer vision. The system generates a pack of transformed images ("lenses") that expose different physical signals: geometry, seams/splits, surface finish, annotations, depth cues, and scale anchors. Specialized agents interrogate those lenses with narrow checklists. A synthesizer fuses results into kinematic and parametric graphs, while a red-team and physics/DFM validators enforce realism.

Unlike UI/UX extraction (pixels to DOM), physical ID extraction is constrained by physics, tooling, manufacturing processes, and ergonomics. Therefore the pipeline is intentionally cranky: it refuses to guess without evidence, labels assumptions, provides disproof tests, and loops back with targeted retries.

Key outputs at a glance

Output	What it contains	Why it matters
Kinematic Assembly Graph (BOM + joints)	Parts, seams/splits, fastening methods, degrees of freedom, assembly order (when inferable).	Explains how the product is built and moves; supports serviceability and reverse engineering.
Parametric Feature Graph	Primitives (lines/arcs/circles), constraints (symmetry, parallelism), feature taxonomy (holes, fillets/chamfers), dimensions/tolerances when present.	Enables CAD reconstruction and measurement with error bars.
CMF Map (Color/Material/Finish)	Material regions, gloss proxy, anisotropy direction, texture class, overmold candidates.	Captures surface intent, premium cues, and finish constraints.
Evidence Pack (Edited images)	Lens images + overlays + vector annotations, bundled with a manifest for traceability and UX.	Builds user trust, powers interactive UI, supports review and export.

2. Problem framing: from UI pixels to physical atoms

Moving from UI/UX extraction to physical Industrial Design (ID) extraction is a paradigm shift. UI/UX pipelines map pixels to a logical interface tree (DOM/components). Physical ID pipelines must map pixels to a 3D, manufacturable object bound by physics, tooling constraints, assembly realities, and human ergonomics.

A 2D ID artifact (sketch, patent drawing, orthographic projection, exploded view, photo, or photoreal render) is a compressed representation of 3D intent. To recover that intent, we must extract not only shapes, but also manufacturing cues: seams and parting lines, draft angles, fasteners, overmolds, finishes, tolerances, and kinematic degrees of freedom.

Typical source documents

- Concept sketches (marker, pencil, vellum scans): expressive lineweight, partial dimensions, ambiguous shading.
- Patent drawings: standardized line styles (solid/dashed/chain), reference numerals, leader lines, multiple views, minimal shading.
- Engineering drawings: dimensions, tolerances, GD&T, section views and hatching, material callouts.
- Exploded views: assembly order cues, callouts, fastener visibility, subassemblies.
- Photos and renders: strong CMF signal, shading/specularity, perspective distortion and lens artifacts.
- Mixed PDFs: pages with both drawings and embedded photos, plus text blocks and tables.

Physical properties we care about

- Geometry: primitives (lines/arcs/circles), fillets vs chamfers, hole taxonomy (through/counterbore/countersink), symmetry and proportional relationships.
- Surfacing: convex/concave regions, continuity cues (smooth vs sharp transitions), ergonomic affordances (thumb recesses, grips).
- CMF (Color/Material/Finish): material regions, gloss vs matte, brushed direction (anisotropy), microtexture class, overmold candidates.
- Assembly: seams/splits, parting lines, weld lines, fastening methods (screws/snaps/adhesive/rivets), hinges and degrees of freedom.
- Metrology: absolute scale hypotheses via anchors (ports, standard fasteners, paper formats) and derived dimensions with error bars.
- Manufacturability: process hypothesis (injection molding, CNC, sheet metal, casting), draft/undercut risks, likely parting strategy.

Key difficulty: most of these properties are not directly visible in a single raw image. They emerge after targeted transformations that isolate weak signals (seams, lineweight language, specular highlights, curvature) and after cross-checking against physical constraints.

3. System goals, assumptions, and non-goals

Goals

- Evidence-first extraction: every claim is traceable to observable cues in one or more transformed images.
- Physics and manufacturing realism: validator gates prevent physically impossible interpretations from shipping silently.
- Reproducibility: every run produces a manifest capturing tool parameters, model versions, and coordinate transforms.
- Human-in-the-loop ready: the system exposes edited images (overlays) as first-class assets for review, UX, and export.
- Modularity: lenses and agents are composable; new lenses/agents can be added without rewriting the whole pipeline.
- Cost-aware iteration: retries are targeted (crop + specific lens), not full reruns.

Assumptions

- Input is primarily 2D (no multi-view photogrammetry requirement), but may contain multiple orthographic views across pages.
- Absolute dimensions may be absent; scale is inferred from anchors when possible and otherwise left as relative with uncertainty.
- We can rasterize PDFs at sufficient DPI for downstream CV (typically 200-400 DPI; higher for fine tolerances).
- Deep models (segmentation, depth) are allowed where they materially improve signal extraction.

Non-goals (initially)

- Fully automatic production CAD output with guaranteed manufacturable correctness in every case (this is a downstream capability).
- Perfect material identification from a single photo/render (we produce candidates + evidence + confidence).
- Replacing engineering judgement: the system produces review-grade artifacts and highlights unknowns.

4. High-level pipeline (phases and data flow)

The pipeline is structured as phases. Each phase produces both machine-readable data and human-reviewable edited images.

Phase overview

- Phase 0 - Ingest and classify: determine document type per page (drawing/photo/mixed) and route to lens packs.
- Phase 1 - Preprocess: dewarp, undistort, normalize illumination, generate multi-scale pyramid and tiles.
- Phase 2 - Lens Pack generation: create transformed images that isolate geometry, seams, CMF, annotations, depth cues, and anchors.
- Phase 3 - Specialist agents: each agent interrogates specific lenses and outputs structured claims with evidence.
- Phase 4 - Synthesis and validation: fuse claims into graphs; run red-team conflict checks and physics/DFM validators; trigger targeted retries.
- Phase 5 - Packaging: produce final graphs, manifests, and an evidence pack (edited images + overlays + vectors) for UX and export.

Minimal data flow sketch

```
INPUT (PDF/pages/images)
 -> Rasterize / Extract pages (DPI-controlled)
 -> Preprocess (dewarp, undistort, normalize, rectification candidates)
 -> Lens Pack (edges, lines, top-hat seams, specular masks, depth/normals, OCR masks, etc.)
 -> Agents (geometry, CMF, seams/assembly, annotations, metrology, DFM...)
 -> Synthesizer (merge + provenance + confidence)
   -> Red-Team + Physics/DFM Validators
     -> Targeted Retry Requests (crop + specific lens + parameter changes)
 -> Final Outputs (graphs + param sketch constraints + CMF map)
 -> UX Assets (base + overlays + vectors + manifest + contact sheet)
```

Crucial design choice: lenses are deterministic and inspectable. Agents should request lenses, not run ad-hoc image edits. This is how the system becomes trustworthy and debuggable.

5. Lens library: transformative CV maps for industrial intent

A 'lens' is a deterministic transformation that converts the raw image into a representation that exposes a specific physical signal. Lenses are generated systematically so that both humans and agents can verify evidence.

Lens design principles

- Each lens has a single purpose (geometry, seams, CMF, annotations, depth cues, or metrology).
- Every lens produces: (a) a transform image, (b) an overlay image aligned to the base/rectified image, and (c) optional masks/vectors/statistics.
- Lenses are parameterized and versioned (kernel sizes, thresholds, model versions). Parameters are captured in the run manifest.
- Lens outputs live in a common coordinate space (typically rectified space) with explicit transform matrices to map back to original pixels.

Lens catalog (summary)

Lens family	What it isolates	Typical ops	Common outputs
Spatial grid & tiling	Relative position and local detail	Grid overlay; adaptive tiling; ROI crops	Anchors, bboxes, tiles, interactive UX targets
Geometry & linework	Edges, primitives, vectorizable strokes	Canny/Scharr; LoG/DoG; skeletonize; LSD/Hough	Lines, circles, contours, primitive fits
Perspective & rectification	True proportions on planar faces	Vanishing points; homography; undistort	Rectified image, homography matrix
Volume & surfacing	3D cues: convexity, curvature, continuity	Depth estimation; normals; curvature maps	Depth/normals, curvature overlays
CMF & specularity	Finish/material regions, gloss cues	Lab/HSV splits; specular masks; intrinsic decomposition	Gloss proxy maps, material regions
Texture & anisotropy	Microtexture direction and periodicity	Gabor; LBP; FFT/Fourier; wavelets	Brush direction, vent pitch, texture classes
Seams & assembly	Panel gaps, parting lines, weld seams	Top-hat/black-hat; morph gradients; low-contrast enhancement	Seam masks + ranked candidates
Annotations & drafting language	Text, dimensions, leader lines, hatching	OCR; arrow/leader detection; dash/lineweight separation	Text masks, dimension bindings, section markers
Metrology anchors	Absolute scale hints	Anchor detectors; template matching; learned detectors	Scale hypotheses + derived mm estimates

5.1 Spatial grid and tiling lenses

Problem: vision models and even classical CV struggle with stable absolute coordinates and with tiny details in large scans. Solution: give them anchors (grid) and isolate micro-features (tiling).

- Grid overlay (coarse): e.g., 10x10 labeled A-J / 1-10 for quick human referencing.
- Grid overlay (fine): e.g., 100x100 for precise localization when needed.
- Adaptive tiling: generate crops around high-frequency regions (corners, holes, ports, seams), plus a uniform tiling baseline.
- Tile priority: corners, edge midpoints, detected holes, high curvature zones, high seam-score zones, OCR-dense zones.

Outputs

- Overlay images with labeled grid coordinates.
- Tile images with consistent naming and coordinate metadata (tile bounding boxes in rectified space).
- Optional: heatmap showing tile selection rationale (entropy/edge density/seam score).

5.2 Geometry and linework lenses (CAD-ish)

Goal: isolate geometry primitives from drawings, sketches, or product silhouettes. This family is the backbone of parametric reconstruction.

- Edge maps: Canny + Scharr (Scharr is often cleaner for fine edges).
- Second-derivative edges: Laplacian / LoG / DoG to highlight shallow relief (emboss/deboss/grooves).
- Binarization + skeletonization: converts strokes to single-pixel centerlines for vectorization.
- Line detection: LSD (line segment detector) or Hough lines; reproject detected lines as overlays.
- Circle/arc detection: Hough circles; arc fitting via RANSAC where Hough fails.
- Contour tracing + polygon simplification: Douglas-Peucker for simplified outlines.

Outputs

- Vector primitives: lines, circles, arcs, polylines with coordinates in rectified space.
- Primitive fit residuals and confidence (supports downstream constraint solving).
- Overlay images showing detected primitives on top of the original/rectified image.

5.3 Perspective, orthographic detection, and rectification

Problem: photos/renders introduce perspective distortion, which corrupts measurements and parallelism. Solution: estimate projection type and rectify planar faces when possible.

- Projection classifier: orthographic vs perspective vs isometric (heuristic + learned).
- Vanishing point estimation: detect dominant line directions and intersect to find vanishing points.
- Homography estimation: rectify a planar face (e.g., front panel) into an orthographic-like view.

- Lens undistortion: correct barrel/pincushion effects when camera metadata or calibration cues exist.

Outputs

- Rectified base image(s) per candidate plane.
- Homography matrix H and inverse H^-1 for mapping coordinates between spaces.
- Rectification confidence score and failure reasons.

5.4 Volume, topography, and surface normal lenses (form)

This family upgrades the pipeline from 'seeing outlines' to 'feeling shape'. Depth and normal maps disambiguate convex domes vs concave pockets, counterbores, and soft grip geometry.

- Monocular depth estimation: produces a relative depth map (topography).
- Surface normals: compute from depth gradients (or predict directly) to represent surface orientation.
- Curvature maps: derived from normals; highlight where surfaces bend (fillets, ribs, embossing).
- Ridge/valley extraction: isolate lines of maximum curvature (often corresponds to feature edges).

Outputs

- Depth map image + statistics (min/max, percentile range).
- Normal map image (RGB orientation) + optional quantized orientation bins.
- Curvature overlay highlighting candidate fillets/chamfers and embossed features.

5.5 Photometric specularity and intrinsic decomposition (CMF)

Industrial designers infer finish from how light bends. We approximate that reasoning by isolating specular highlights and separating shading from albedo.

- Lab/HSV channel splits: L channel approximates luminance; S channel approximates chroma intensity.
- Specular highlight mask: threshold high luminance (often low saturation) to isolate reflections.
- Highlight geometry metrics: continuity (broken vs smooth), width (sharp vs diffused), orientation consistency.
- Intrinsic decomposition: separate albedo (base color) from shading (illumination), reducing false seam detections caused by shadows.
- Color clustering / superpixels: discretize regions to identify material/paint/inserts boundaries.

Outputs

- Specular mask overlay and computed gloss proxy per region.
- Albedo vs shading preview images (where available).
- Material-region segmentation masks and region adjacency graph (useful for CMF mapping).

5.6 Texture, anisotropy, and periodicity lenses

Texture analysis distinguishes rubberized grips from painted shells, finds brushed direction on metal, and measures vent/knurl pitch.

- Gabor filter bank: detect oriented textures; output dominant direction per pixel/region (anisotropy).
- LBP (Local Binary Patterns): describe microtexture class; robust to lighting changes in many cases.
- Fourier magnitude spectrum: detect periodic patterns; estimate pitch and orientation (vents, grills).
- Wavelet decomposition: isolate frequency bands to separate form vs texture.

Outputs

- Orientation map (dominant brush direction) + confidence.
- Texture class map per region.
- Periodicity estimates (pitch in pixels -> convertible to mm once scale is known).

5.7 Kinematic top-hat and seam isolation lenses (assembly)

Panel gaps, parting lines, and weld seams are often subtle. Top-hat/black-hat morphology suppresses large shapes and makes thin ridges/valleys pop.

- Top-hat (bright ridges) and black-hat (dark ravines): isolate microfeatures like seams and gaps.
- Morphological gradient: emphasizes low-contrast boundaries even when edges are weak.
- CLAHE + gradient: amplifies subtle seams in renders/photos without over-amplifying noise.
- Seam candidate scoring: length, straightness, continuity across lighting normalization, and alignment with symmetry axes.

Outputs

- Seam mask + ranked seam segments (vector polylines).
- Overlay heatmaps and candidate seam polylines on the base image.
- Seam persistence checks: does the seam remain after intrinsic decomposition (shadow removal)?

5.8 Drafting language lenses (lineweight, dashes, annotations)

In patent and engineering drawings, stroke style is semantics. Separating thick/thin/dashed lines prevents AI from confusing arrows and hidden lines with geometry.

- Semantic lineweight separator: split strokes by width (thick outlines vs thin detail vs fillet hints).
- Dashed-line detector: classify hidden edges and centerlines; avoid treating them as visible seams.
- OCR and layout segmentation: identify text blocks, dimension strings, and callouts.
- Leader-line and arrowhead detection: bind dimension text to features.

- Hatching detector: identify section views and interpret cross-section meaning.

Outputs

- Separate stroke-layer images (thick/thin/dashed/centerline).
- Text masks and bounding boxes; extracted strings; leader-line bindings.
- Section/hatching overlays with detected hatch angle/frequency.

5.9 Metrology anchors and scale hypothesis lenses

Absolute scale is often missing in concept sketches and patents. When possible, infer scale from known anchors: ports, fasteners, batteries, paper formats, or hands.

- Anchor detection: template matching or learned detectors for standard items (e.g., USB-C, common screw heads, A4 borders).
- Multiple-hypothesis scale: propose several candidate scales and pick the one most consistent across independent anchors.
- Error bars: compute uncertainty from anchor detection confidence and perspective distortion residuals.

Outputs

- List of scale hypotheses with rationale and confidence.
- Derived dimensional estimates (overall size, feature diameters) with uncertainty.
- Overlay showing anchor detections and reference measurement lines.

5.10 Recommended default lens pack

For most pages, generate a default pack and let agents choose which layers to trust. The default pack also powers the UX Evidence Gallery.

```
DEFAULT LENS PACK (per page)
  Base: original + normalized + (optional) rectified candidates
  Geometry: Canny, Scharr, LoG, Skeleton, LSD/Hough Lines, Hough Circles
  Assembly: Top-Hat / Black-Hat seams, Morphological Gradient
  CMF: Lab split, Specular mask, Color clusters
  Texture: Gabor orientation map, Fourier periodicity map
  Annotations: OCR blocks, Text mask, Arrow/leader overlay, Lineweight layers (if drawing-like)
  Metrology: Grid overlay (coarse), Anchor detections (if enabled)
  Tiles: adaptive crops around seams/holes/ports/high curvature
```

For document-type-specific optimization, enable/disable families. For example: technical drawings prioritize lineweight/dash/hatching; renders prioritize specularity/depth.

6. The Senior Engineering Swarm: specialist agents and question sets

Instead of one monolithic 'Vision Agent', we create a swarm of narrowly-scoped specialist agents. Each agent sees a tailored subset of lenses and answers a small set of questions. This reduces cognitive load, reduces hallucinations, and improves extraction accuracy.

Agent design rules

- Each agent produces structured claims, not prose.
- Each claim must include evidence pointers (which lens, where, what cue).
- If evidence is insufficient, the agent must output UNKNOWN rather than guessing.
- Agents should state a disproof test: what would change their mind.
- Agents operate in the same coordinate space (rectified by default).

Agent roster (summary)

Agent	Primary responsibility	Inputs (lens families)	Key outputs
Volumetric Sculptor	Form & surfacing: continuity, fillets/chamfers, draft hints, affordances	Depth, normals, curvature, ridge/valley, silhouette	Surface features, continuity class, draft cues, ergonomic affordances
Geometer	2D structure and primitive geometry	Edges, skeleton, LSD/Hough, contours, rectified views	Primitives, containment graph, symmetry constraints, feature taxonomy
Surveyor	Layout, bounding boxes, relative placement and spacing	Grid overlays, tiles, rectified base	BBoxes, alignment, spacing/pitch estimates
CMF Physicist	Material and finish reasoning from light & texture	Specular masks, Lab/HSV splits, intrinsic decomposition, texture maps	Material regions, gloss proxy map, anisotropy direction, overmold candidates
Texture/Pattern Analyst	Periodic patterns and microtexture	Gabor, LBP, Fourier, wavelets	Pattern type, pitch/orientation, texture class
Seam & Parting-Line Detective	Assembly boundaries and splits	Top-hat/black-hat, morph gradient, low-contrast boundary lens	Seam masks, seam polylines, housing split hypothesis
Annotation & GD&T Reader	Dimensions, tolerances, section interpretation	OCR + layout, leader lines/arrows, hatching, dash styles	Bound dimensions, datums, tolerances, section meaning
Mechanist	Kinematics & joints: DOF, fasteners,	Seams, lineweight layers, tiles,	Joint types, DOF graph, fastening

	hinges	primitives	method candidates
Metrologist	Absolute scale and derived dimensions	Anchors, grid, rectification	Scale hypotheses, mm estimates + error bars
DFM Validator	Manufacturing process plausibility	Draft cues, seams, primitives, texture/finish hints	Process hypothesis + constraints, red flags
Draft/Undercut Checker	Molding feasibility and undercut risks	Normals/curvature + seam candidates	Pull direction candidates, undercut risk map
Red-Team Reviewer	Attack unsupported claims and contradictions	All claims + evidence	Conflict list, evidence gaps, targeted retry requests

6.1 Claim schema (what every agent outputs)

Agents output a list of Claim objects. The claim schema encodes the cranky senior mindset: evidence, assumptions, and falsifiability.

```
// Claim object (conceptual schema)
{
 "claim_id": "seam_07",
 "type": "ASSEMBLY_SEAM | HOLE | FILLET | MATERIAL_REGION | DIMENSION | JOINT | ...",
 "statement": "Primary housing appears to be a 2-part clamshell split along the midline.",
 "location": {
  "space": "RECTIFIED",
  "bbox": [x, y, w, h],
  "polyline": [[x1,y1],[x2,y2], ...],
  "grid_anchor": "E5-F6"
 },
 "evidence": [
  {"lens": "black_hat_seams_k11", "cue": "continuous dark ravine persists after illumination normalization"},
  {"lens": "morph_gradient", "cue": "boundary visible as consistent gradient edge"}
 ],
 "confidence": 0.82,
 "assumptions": [
  "Image is a render/photo (not a stylized sketch).",
  "Seam is not a shadow; verified via intrinsic decomposition."
 ],
 "disproof_test": "If the boundary disappears in albedo-only (intrinsic) view, it was shading not a seam.",
 "severity_if_wrong": "BLOCKER | CONCERN | NIT",
```

```
  "next_action_if_uncertain": "Tile around midline and rerun black-hat with kernel=15."
}
```

6.2 Specialist question sets (examples)

Volumetric Sculptor (Form & surfacing)

- Are transitions sharp (chamfers) or smooth (fillets)? Are there cues for higher continuity (smooth highlight flow)?
- Do vertical walls show draft angles (taper) consistent with molding? If not visible, mark UNKNOWN.
- Where are ergonomic affordances (thumb recess, trigger contour, grip texture zones)?
- Are there concave pockets indicating counterbores, recesses, or buttons?

CMF Physicist (Materials & finish)

- Based on specularity spread, classify finish candidates (high-gloss, satin, matte, bead-blasted, anodized).
- Do highlights form sharp continuous lines (Class-A) or broad diffused blobs (matte/rough)?
- Where do texture frequency changes suggest overmolded grip areas?
- Is the body clear, translucent, or opaque (based on internal light scattering cues)?

Mechanist (Kinematics & assembly)

- Map seams/parting lines to determine housing piece count (clamshell vs unibody).
- Identify fastening methods: screw bosses, snap-fits, rivets, hinges, adhesives, ultrasonic weld seams.
- Identify degrees of freedom for moving parts (single-axis hinge, linear slide, latch).
- Flag assembly claims that violate basic feasibility (e.g., no split strategy for enclosed volumes).

Annotation & GD&T Reader (Drawings)

- Extract dimension strings and units; bind each to a feature via leader lines.
- Detect GD&T symbols and feature control frames; map to datums and features.
- Interpret section hatching: which parts are cut, which are hidden, what material conventions appear.
- Separate hidden/dashed lines from visible geometry to avoid false seams.

Metrologist (Scale)

- Locate universal anchors (ports, standard fasteners, paper frames, known components).
- Produce multiple candidate scales; pick the most consistent across independent anchors.
- Return overall dimensions and key feature sizes with error bars and rationale.

7. 'Cranky senior team' behavior: evidence, red-teaming, and validation gates

The goal is not to make agents sound cranky. The goal is to make the system behave like a tough design review: claims are accepted only with evidence, contradictions are surfaced, physics is enforced, and unknowns are documented explicitly.

7.1 The Senior Review Contract

- No evidence, no claim: if the system cannot point to an observable cue in the provided lenses, it must output UNKNOWN.
- Separate facts from assumptions: every claim lists assumptions that could be wrong (lighting, projection type, hidden fasteners).
- Provide a disproof test: each claim includes a concrete way it could be falsified with available lenses.
- Severity labels: BLOCKER (physics/DFM contradiction), CONCERN (weak evidence), NIT (minor).
- Traceability: every accepted claim references the lens outputs (IDs) and coordinates.

7.2 Evidence thresholds (multi-lens requirement)

Senior reviewers distrust single-signal conclusions. Encode that distrust as policy:

- High confidence requires support from at least two independent lens families (e.g., seam visible in black-hat and persists after intrinsic decomposition).
- If a claim is supported by only one lens, cap confidence (example policy: confidence <= 0.6).
- If lenses disagree, downgrade confidence and create a retry request with specific instructions.

7.3 The Red-Team Reviewer (allowed to be annoying)

A dedicated red-team agent audits claims for unsupported leaps, conflicts, and missing disproof tests. It produces a punch list of targeted reruns, similar to review comments on a mechanical drawing.

- Conflict detection: two claims describe mutually exclusive topology ("inside" vs "adjacent"), or different material classes for the same region.
- Evidence gap detection: claim has no lens-based cue or uses only subjective language ("looks like").
- Physics plausibility check: claim violates manufacturing constraints (e.g., injection-molded enclosed hollow without split strategy).
- Retry synthesis: produce a minimal set of reruns (crop + lens + parameter change) to resolve the conflict.

7.4 Physics and DFM validators (hard gates)

Validators are deterministic rules (or lightweight models) that enforce the laws of manufacturing and mechanics. They turn 'cool story' into 'show me the seam / draft / joint'.

Example validator rules

- Injection molding plausibility: if process=InjectionMolded, the model must include (a) parting strategy or weld strategy, (b) plausible draft directions, (c) undercut flags if present.
- Enclosure reality check: a sealed hollow volume requires an assembly seam, welding, adhesive joint, or a different process (e.g., blow molding).
- Fastener realism: if screws are claimed, expect boss/counterbore cues or at least consistent mounting zones; otherwise downgrade confidence.
- Kinematic DOF sanity: any moving part claim must correspond to a visible joint/clearance path; otherwise mark UNKNOWN.

Validation loop example (from the conversation)

Sculptor claims: 'Completely enclosed hollow sphere.' CMF claims: 'Rigid injection-molded ABS.' Synthesizer detects impossibility: injection molding cannot produce a fully enclosed hollow sphere without a split/weld strategy. The system triggers a targeted retry: 'Zoom around the equator using black-hat/top-hat; find a parting line or ultrasonic weld seam.'

7.5 Triage board output (what users see)

Instead of a single confident narrative, produce a review-style summary that mirrors real ID/mech culture:

- Accepted facts: high-confidence, multi-lens-backed claims.
- Open questions: explicitly unknown items with the exact data needed to resolve them.
- Blockers: physics/DFM contradictions.
- Concerns: weak evidence or conflicts needing review.
- Nits: minor cleanup (naming, annotation alignment, etc.).

7.6 Targeted retries (cost-aware iteration)

Retries are always targeted: rerun a specific lens with specific parameters on a specific ROI. Avoid global reruns unless the preprocessing or rectification step changed.

```
RETRY REQUEST (example)
 ROI: bbox=[x,y,w,h] around suspected seam
 Lenses to rerun:
  - black_hat_seams(kernel=15)
  - intrinsic_decompose()  // verify seam is not shading
  - lsd_lines()          // check alignment with geometry axes
 Expected decision:
  - seam persists across lenses -> accept clamshell split
  - seam disappears in albedo-only -> downgrade seam claim; treat as shading
```

8. Synthesis outputs: kinematic assembly graph, feature graph, CMF map

The synthesizer merges agent claims into a coherent object model. Unlike UI extraction which outputs a DOM tree, industrial design extraction outputs an assembly and feature representation grounded in manufacturing.

8.1 Precedence and conflict resolution

When claims conflict, the synthesizer uses precedence rules based on which lens family is more authoritative for the question.

- Topology (inside/contains) -> prioritize geometry/linework evidence (Geometer + vector primitives).
- Coordinates and spacing -> prioritize grid/tiling evidence (Surveyor).
- Material regions/finish -> prioritize specularity + intrinsic + texture evidence (CMF Physicist).
- Dimensions/tolerances -> prioritize annotation bindings and metrology anchors (Annotation Reader + Metrologist).
- Assembly feasibility -> prioritize seam evidence and DFM/draft validators (Mechanist + Seam Detective + Validators).

8.2 Kinematic Assembly Graph (BOM + joints)

A kinematic assembly graph models parts and how they connect. Nodes are parts/subassemblies; edges are joints (static fixations or moving joints).

Core fields

- Parts: housing halves, grips, buttons, triggers, covers, inserts.
- Manufacturing hypothesis per part: injection molded PC/ABS, CNC aluminum, sheet metal, overmolded TPE, etc.
- Surface finish cues: matte texture families, gloss accents, brushed direction.
- Tooling cues: draft present, parting line candidate, undercut risks.
- Kinematic connections: screws, snaps, welds, pins, hinges; plus DOF for moving joints.

Example: kinematic assembly graph JSON (illustrative)


```
{
 "product_concept": "Handheld_Power_Tool",
 "ergonomics": {
   "absolute_scale_estimate": {
     "value": "Height ~185 mm",
     "basis": "Anchor: battery footprint / handle proportion (example)",
     "confidence": 0.55,
     "error_bar_mm": 15
   }
 },
 "assembly_graph": {
```

```
  "housing_left": {
   "manufacturing": {
    "process": "InjectionMolded",
    "material_candidates": ["PC-ABS", "ABS"],
    "evidence": ["draft_cues", "seam_midline_candidate"]
   },
   "surface_finish": {
    "finish_candidates": ["Matte textured (VDI-like)", "Satin"],
    "accents": ["High-gloss band"],
    "evidence": ["specular_mask_stats", "texture_map"]
   },
   "tooling_cues": {
    "draft_present": "UNKNOWN",
    "parting_line": {"path_id": "seam_07", "confidence": 0.82}
   },
   "connections": [
    {"to": "housing_right", "fastening": "Screws_or_snaps", "joint": "StaticFixation",
"confidence": 0.62},
    {"to": "trigger_mechanism", "fastening": "Pivot_pin", "joint": "1AxisRotation", "confidence":
0.70}
   ]
  },
  "overmold_grip": {
   "manufacturing": {"process": "Overmold", "material_candidates": ["TPE"], "confidence":
0.68},
   "surface_finish": {"texture": "Ribbed micro-texture", "evidence": ["lbp_texture_class",
"gabor_orientation"]}
  }
 }
}
```


Note: this graph is intentionally allowed to contain UNKNOWNs. For physical design, honest uncertainty is better than confident nonsense.

8.3 Parametric Feature Graph (geometry you can rebuild)

The parametric feature graph represents geometry as primitives plus constraints. It is the bridge from CV extraction to CAD reconstruction.

- Primitives: line segments, circles, arcs, splines (if fitted), polylines.
- Constraints: parallel, perpendicular, concentric, equal radius, symmetry axis, coincident, tangent.

- Features: holes (through/counterbore/countersink), fillets/chamfers, pockets, ribs (when inferable).
- Measurements: derived from anchors or extracted dimensions; stored with uncertainty.

8.4 CMF map (Color/Material/Finish)

CMF outputs are region-based. The system should avoid claiming a single material for the whole product when evidence is local.

- Region segmentation: paint/inserts/overmold regions.
- Finish proxies: gloss map from specularity; matte classification; highlight continuity metrics.
- Anisotropy: brushed direction map and confidence.
- Texture: microtexture class per region (grip vs shell).

8.5 Evidence Pack (edited images) as a first-class output

Every synthesis must include an evidence pack: the lens images and overlays that justify claims. This pack is used both for internal debugging and for product UX (interactive inspector, exportable reports).

9. Constraint solving: from detection to parametric reconstruction

Detection is not reconstruction. To move from 'I see a circle-ish thing' to 'this hole is 6.2 mm ± 0.4 mm and concentric with feature X', we treat extracted primitives and dimensions as constraints in an optimization problem.

9.1 Why constraint solving matters

- Industrial design intent is often expressed via symmetry and repeated radii: constraints recover that intent even when pixels are noisy.
- Constraint solving reduces drift and accumulates evidence across lenses and pages.
- The solver can output error bars, enabling honest uncertainty reporting.
- It enables downstream CAD sketch generation (2D constraints are the language of parametric CAD).

9.2 Inputs to the solver

- Primitive fits: lines, circles, arcs with residuals and confidence (from geometry lenses).
- Symmetry hypotheses: axes/planes detected by the Symmetry & Constraints agent.
- Dimensions: from annotation bindings (hard constraints) and from metrology anchors (soft constraints with uncertainty).
- Topological relations: containment/adjacency relationships from the Geometer.
- Rectification transforms: homography matrices relating views/spaces.

9.3 Solver approach (practical)

A pragmatic solver stack for 2D reconstruction:

- Step 1 - Robust primitive fitting: use RANSAC to fit lines/circles/arcs and reject outliers; keep residuals.
- Step 2 - Build a constraint graph: nodes are primitives/features; edges are constraints (parallel, equal radius, concentric, symmetric).
- Step 3 - Solve as weighted least squares: hard constraints from dimensions; soft constraints from anchors; weights from confidence/residuals.
- Step 4 - Validate: ensure solved geometry is consistent with observed pixels (reprojection error) and with manufacturing validators.
- Step 5 - Export: output a parametric sketch representation (constraints + dimensions) for CAD reconstruction.

9.4 Multi-view / multi-page consistency

When a document provides multiple views (front/side/top/isometric), solve them jointly where feasible:

- Shared dimensions: the same feature diameter (e.g., hole) should match across views within tolerance.
- Shared symmetries: global symmetry axes should be consistent across orthographic views.

- Cross-view anchoring: map features via reference numerals (patents) or via feature descriptors (holes, corners).
- Joint inference: if a hinge is visible in one view, confirm its axis in another view.

9.5 Output of constraint solving

- Updated primitives with cleaned parameters (true circles, straighter lines) and attached uncertainty.
- Explicit constraints list (CAD-ready).
- Reprojection error metrics and a 'fit quality' score.
- Flags for underconstrained cases: where multiple solutions fit the same evidence.

10. Tooling: CV engines and how to expose them as agent tools

Agents should not directly manipulate images via ad-hoc code. Instead, provide a curated toolbelt: callable functions that run deterministic CV operations or model inference and return both data and edited images. This keeps the system reproducible and productizable.

10.1 Core CV engines (what you wrap)

Classical CV and image processing

- OpenCV (cv2): warps, filtering, morphology, edges, contours, Hough, geometry, calibration.
- scikit-image: skeletonize, watershed, regionprops, morphology utilities, measurements.
- SciPy/NumPy: convolution, FFT, distance transforms, optimization primitives.
- Pillow / imageio: reliable I/O and basic operations.
- libvips/pyvips (optional): huge-image performance for large scans.

Document rendering and ingestion

- PyMuPDF (fitz) or Poppler/pdf2image: rasterize PDF pages at controlled DPI.
- Optional vector extraction (when PDFs contain vector drawings): extract paths in addition to raster images.

Deep vision models (where they materially help)

- Segmentation/detection: Torchvision, Detectron2, MMDetection/MMSegmentation; optionally SAM-style segmentation for region proposals.
- Depth estimation: monocular depth models for topography; normals derived from depth.
- Intrinsic decomposition: albedo vs shading separation (reduces shadow-induced errors).
- OCR and layout: Tesseract / PaddleOCR / docTR; layout segmentation to separate text/figures.

Geometry and graphs

- Shapely: 2D geometric operations once you have contours/polylines.
- Open3D/trimesh (optional): depth-to-pointcloud utilities and normals processing.
- NetworkX: kinematic assembly graphs and feature graphs.

10.2 Expose capabilities as tools (the right abstraction)

Do not expose 500 OpenCV functions. Expose 30-80 stable 'lens tools' that return: (a) a transformed image ID, (b) an overlay image ID, (c) structured outputs (masks/vectors/stats).

Tool API shape (recommended)

```
ToolInput:
 image_id: string
 roi: optional bbox or polygon in RECTIFIED space
 params: operation parameters (kernel sizes, thresholds, model name/version)

ToolOutput:
```

```
transform_image_id: string
overlay_image_id: string
masks: {name: mask_id, ...}
vectors: {name: vector_json_id, ...}
stats: {...}
coordinate_space: "RECTIFIED"
transform_refs: {base_to_transform: matrix_id, ...}
```

Recommended tool families (examples)

- Preprocess: dewarp_page, undistort_lens, normalize_illumination, rotate_sweep, build_pyramid
- Rectification: detect_vanishing_points, rectify_plane_homography
- Geometry: canny_edges, scharr_edges, log_edges, skeletonize, lsd_lines, hough_circles, contours_simplify
- Assembly: top_hat_seams, black_hat_seams, morph_gradient_boundaries, seam_segment_rank
- CMF: lab_split, hsv_split, specular_mask, intrinsic_decompose, color_cluster_regions
- Texture: gabor_orientation_map, lbp_texture_map, fourier_periodicity
- Annotations: ocr_blocks, mask_text_regions, detect_arrowheads, detect_leader_lines, detect_hatching, separate_lineweights, detect_dashed_lines
- Metrology: detect_anchors, estimate_scale_hypotheses

10.3 Caching and reproducibility

- Content-address lens outputs: cache by (image_hash, tool_name, params, roi).
- Store model versions and parameters in the manifest; treat outputs as immutable artifacts.
- Use a consistent coordinate convention (top-left origin, x right, y down) and record every transform matrix applied.
- Stamp overlays with small legends (tool + key params) so exported images remain interpretable.

11. Edited images as UX assets: overlays, manifests, and interactive inspector UI

Edited images are not just debug artifacts. In an ID/mech workflow, they are the product: the evidence trail that users trust, share, and export. Therefore the pipeline must generate UX-ready assets: base images, transparent overlays, vectors, and manifests.

11.1 UX primitives powered by edited images

Evidence Gallery (entry point)

For each page, show a grid of thumbnails that summarize what the system extracted: original/rectified, edges, lines, seams, specular/gloss, depth/normals (if available), OCR/annotations.

Layered Inspector (main workspace)

- One canvas with zoom/pan, plus layer toggles and opacity sliders (like Figma/Photoshop).
- Lens tabs: Geometry / Seams / CMF / Annotations / 3D / Metrology.
- Hover/click targets driven by vector overlays (lines, circles, polylines, bboxes).
- Before/after slider: original vs selected overlay composite.

Claim ↔ evidence binding (trust engine)

- Every claim is a clickable card; clicking highlights the corresponding region on the image.
- Every highlighted region links back to the lens layers that support it (multi-lens evidence).
- Users can Accept / Reject / Needs better view; rejections trigger targeted retries on that ROI.

Compare modes

- Lens compare: side-by-side edges vs seams vs specular vs albedo.
- Iteration compare: run #1 vs run #2 after user feedback.
- Cross-page compare: align multiple views (front/side/top) when present.

11.2 Asset packaging: ship layers, not just flattened PNGs

A flattened image is limiting. For UX, ship base + overlays + vectors + masks + metadata so the frontend can render interactively.

Recommended output bundle per page

```
/page_003/
  base_rectified.webp
  base_original.webp
  overlays/
    overlay_edges.png        (alpha)
    overlay_lines.png        (alpha)
    overlay_seams.png        (alpha)
    overlay_specular.png     (alpha)
```

```
  overlay_ocr.png          (alpha)
  overlay_depth.png        (alpha or standalone)
vectors/
  geometry_vectors.json    (lines/circles/polylines)
  seam_vectors.json
  annotation_vectors.json  (text boxes, arrows, leader lines)
masks/
  material_regions.rle.json  (or PNG masks)
  seam_mask.png
manifest.json
thumbnails/
  gallery_contact_sheet.webp
```

Manifest requirements (the glue)

- List every lens produced, with tool name, parameters, and artifact IDs.
- Record coordinate space, image dimensions, and any rectification transforms (homographies).
- Map claim IDs to vector/mask IDs and evidence lens IDs.
- Include versioning: pipeline version, model versions, tool versions.

11.3 Performance for big scans (map-tile approach)

Industrial PDFs can be huge. For smooth UX, generate pyramidal tiles for base and key overlays. Lazy-load tiles as the user zooms (like map apps).

- Create multi-resolution tiles for base_rectified and high-value overlays (seams, edges).
- Use WebP/AVIF for base images; keep overlays as PNG alpha where needed.
- Keep vectors lightweight; use RLE for large masks.
- Ensure the manifest references tile pyramids and coordinate transforms.

11.4 Visual language (make overlays coherent)

- Consistent category colors (Geometry vs Seams vs CMF vs Text), with legends.
- Opacity defaults that preserve the base image; hover/selected states increase contrast.
- Avoid red/green-only cues; support color-blind safe palettes.
- Stamp overlays with tool + key params for exportability and auditability.

12. Data model and file formats (IDs, masks, vectors, coordinate spaces)

A robust data model prevents hidden coupling between CV, agents, and frontend. The key is to treat every artifact (images, overlays, masks, vectors, matrices) as an immutable object referenced by ID.

12.1 Coordinate spaces

- ORIGINAL: pixels as provided by the source image or PDF rendering.
- NORMALIZED: after illumination normalization and denoise (same geometry, different intensities).
- RECTIFIED: after perspective correction/homography for a chosen plane.
- TILE: local coordinate system for a crop; must include mapping back to RECTIFIED and ORIGINAL.
- MODEL: if a model expects resized input, record the resize transform explicitly.

Transform matrices (must be explicit)

Store forward and inverse transforms whenever you change coordinate space. At minimum: ORIGINAL <-> RECTIFIED homography, and RECTIFIED <-> TILE affine transforms.

12.2 Artifact identifiers

- image_id: points to a raster image (base or lens output).
- overlay_id: points to an alpha image aligned to a base space.
- mask_id: points to a binary/label mask (PNG or RLE).
- vector_id: points to vector geometry (JSON/SVG).
- matrix_id: points to a transform matrix (JSON).
- claim_id: points to a claim object, which references artifacts.

12.3 Vector formats

Vectors drive interactivity: hover, click targets, measurement readouts. A simple JSON schema is often sufficient.

```
// Example vector JSON
{
 "space": "RECTIFIED",
 "primitives": [
  {"id": "line_12", "type": "LINE", "p1": [x1,y1], "p2": [x2,y2], "confidence": 0.9},
  {"id": "circle_03", "type": "CIRCLE", "center": [cx,cy], "r": 42.1, "confidence": 0.7},
  {"id": "poly_07", "type": "POLYLINE", "points": [[x,y],...], "confidence": 0.8}
 ]
}
```

12.4 Mask formats

- Binary masks: PNG with alpha or 0/255 values (simple, big).

- Label masks: indexed PNG for region IDs (material regions, components).
- RLE encoding: compact for large masks; good for web transfer and storage.

12.5 Run manifest schema (conceptual)

```
{
 "run_id": "2026-02-18T12:34:56Z_abcd",
 "pipeline_version": "id-swarm-1.0",
 "inputs": {
  "source_type": "PDF",
  "pages": [{"page": 3, "original_image_id": "...", "dpi": 300}]
 },
 "spaces": {
  "rectification": [
   {"plane_id": "front_panel", "H_matrix_id": "...", "confidence": 0.76}
  ]
 },
 "lenses": [
  {"name": "black_hat_seams", "params": {"kernel": 11}, "transform_image_id": "...",
"overlay_id": "..."},
  {"name": "specular_mask", "params": {"colorspace": "Lab"}, "mask_id": "...", "overlay_id":
"..."}
 ],
 "claims": [
  {"claim_id": "seam_07", "agent": "SeamDetective", "evidence_lenses":
["black_hat_seams_k11", "intrinsic_albedo"], "confidence": 0.82}
 ],
 "outputs": {
  "assembly_graph_id": "...",
  "feature_graph_id": "...",
  "cmf_map_id": "...",
  "evidence_pack_path": "..."
 }
}
```

In production, treat the manifest as the single source of truth for auditability, caching, and UX reconstruction.

13. Implementation architecture (services, storage, compute, caching)

A production implementation typically separates orchestration (decisions) from computation (CV operations). The swarm agents can run in an LLM orchestrator layer, while CV runs in worker services.

13.1 Reference architecture

```
[Client UI]
 -> Upload / Select Document
 -> View Evidence Gallery + Inspector

[API Gateway]
 -> Auth, rate limits, upload URLs

[Orchestrator Service]
 -> Ingest + classify pages
 -> Build lens plan
 -> Dispatch CV jobs to workers
 -> Run agents on produced artifacts
 -> Synthesize + validate + retry
 -> Produce final graphs + manifests

[CV Worker Pool]
 -> OpenCV + skimage operations
 -> OCR workers
 -> Deep model inference workers (segmentation, depth, intrinsic)
 -> Tile generation + pyramid tiling

[Storage]
 -> Object store for images/overlays/masks/vectors/manifests
 -> Metadata DB for runs, artifacts, claims, indexing

[Queue]
 -> Job queue for CV tasks and model inference (retries + prioritization)
```

13.2 Key implementation details

- Artifact immutability: treat every generated file as immutable; new parameters create new artifact IDs.
- Caching by content: avoid recomputing lenses when inputs and params match.
- ROI-first compute: most expensive lenses should support ROI (crop) execution for targeted retries.
- Deterministic builds: pin model versions and tool versions; record in manifest.

- Coordinate rigor: every output must declare coordinate space and mapping matrices.

13.3 Scheduling and prioritization

- Generate lightweight lenses first (edges, top-hat, OCR) to provide early UX feedback.
- Defer expensive deep lenses (depth, intrinsic decomposition, SAM) until needed by validators or user requests.
- Use red-team outputs to prioritize reruns that resolve blockers first.
- Support progressive refinement: user can start reviewing while deeper lenses compute.

13.4 Security and privacy (baseline)

- Treat customer documents as sensitive IP: encrypt storage, restrict access by tenant.
- Keep manifests free of raw text content unless user explicitly opts in (especially for OCR).
- Provide configurable retention policies and export controls.
- Log only high-level pipeline metadata; avoid storing unnecessary intermediate artifacts.

14. Evaluation and QA (metrics, gold sets, regression tests)

Because the system produces structured claims and edited images, evaluation should combine quantitative metrics with review workflows. The goal is not only accuracy, but also calibrated uncertainty and debuggability.

14.1 Build a gold document set

- Collect representative documents across types: patents, engineering drawings, sketches, renders, exploded views.
- Annotate ground truth where feasible: seams, splits, primitives, dimensions, material regions, joint types.
- Include hard cases: low-contrast seams, heavy shading, messy scans, rotated pages, dense annotations.

14.2 Metrics by subsystem

Geometry primitives

- Line/arc/circle detection: precision/recall vs annotated primitives.
- Fit quality: average residual and reprojection error.
- Constraint satisfaction rate after solving.

Seams and assembly

- Seam detection: precision/recall for parting lines/panel gaps.
- Split hypothesis accuracy: clamshell vs unibody classification accuracy where labeled.
- Joint type accuracy (screw/snap/weld) where labeled.

CMF and finish

- Material-region segmentation: IoU / Dice for region masks (when ground truth exists).
- Gloss proxy calibration: correlation with annotated finish classes (matte/satin/gloss).
- Anisotropy direction error: angular error vs annotated brush direction.

Annotations and metrology

- OCR: character/word accuracy on dimension strings and callouts.
- Dimension binding accuracy: correct mapping from text to feature via leader lines.
- Scale inference error: mm error vs known dimensions (when known).

System-level

- Claim calibration: do confidence scores match empirical accuracy (reliability curves)?
- Unknown rate: how often the system chooses UNKNOWN appropriately (better than hallucinating).
- Blocker catch rate: how often validators catch physically impossible outputs before shipping.

14.3 Regression tests

- Lock a snapshot of the gold set and run it on every model/tool update.
- Compare manifests to detect silent parameter changes.

- Pixel-diff or structural diff key overlays (edges, seams) to catch unexpected behavior changes.
- Treat validator changes like code changes: unit test each rule with curated examples.

15. Roadmap: MVP to production-grade reverse engineering

A practical roadmap builds trust early (edited images + simple claims) and adds sophistication (depth, constraint solving) once the evidence loop is solid.

15.1 MVP (trust-first, low compute)

- PDF rasterization + page classification (drawing vs photo).
- Preprocess: dewarp (basic), illumination normalization, rotation fix.
- Lens pack v0: grid overlay, edges (Canny/Scharr), line detection (LSD/Hough), top-hat/black-hat seams, OCR blocks + text masks, color clustering.
- Agents v0: Geometer, Seam Detective, Annotation Reader (basic), Metrologist (anchors optional), Red-Team.
- Outputs: evidence gallery, claims list with evidence pointers, initial assembly split hypothesis.
- Export: downloadable evidence pack (base + overlays + manifest).

15.2 v1 (industrial depth: CMF + manufacturing realism)

- Specularity lens + gloss proxy metrics; anisotropy maps for brushed finishes.
- Intrinsic decomposition (albedo/shading) to reduce false seams.
- DFM validators: injection molding plausibility, enclosure split checks, basic fastener plausibility.
- Mechanist agent for DOF/joints; Draft/Undercut checker (heuristic first).
- Interactive inspector UI: layer toggles, claim highlighting, accept/reject workflow, targeted retries on ROI.

15.3 v2 (parametric reconstruction and multi-view consistency)

- Monocular depth + normals + curvature maps for form/fillet extraction (Volumetric Sculptor).
- Constraint solving pipeline (robust fitting + weighted least squares) and CAD-ready constraint export.
- Cross-page multi-view linking: bind features across orthographic views and patents via reference numerals.
- Exploded view BOM extraction (callout linking + assembly order) where applicable.

15.4 Production hardening

- Artifact caching and content-addressed storage to control cost.
- Pyramidal tiling for large images and overlays (smooth zoom).
- Model and tool version pinning, regression suite, and monitoring.
- Tenant isolation, encryption, retention policies, and export controls.

Closing note

The philosophical core of this system is the cranky senior mindset: extract intent only when evidence supports it, expose the evidence visually, enforce physics, and document uncertainty. That is what turns 'image recognition' into 'automated reverse engineering'.