

Scene Pipeline V2 — Technical White Paper

Title: Photo-Matched Hybrid Scene Reconstruction for Blender via ComfyUI

System: Portal Worlds 0.3.0 + Portal9 Hy3D

Date: August 2026

Audience: Graphics engineers, pipeline TDs, researchers evaluating hybrid reconstruction

Abstract

Scene Pipeline V2 reconstructs editable 3D scenes from a single RGB photograph by **splitting the problem by geometric class**. Discrete objects are segmented (SAM3), optionally completed (Flux Kontext), and meshed with Hunyuan3D. Planar regions — floors, roads, sidewalks, parking, walls — are reconstructed by **depth unprojection and RANSAC plane fitting**, then extruded in Blender with procedural markings. Lighting and weather are estimated from sky and shadow cues and applied as a Nishita world plus atmosphere. A parallel **Asset Bundle** path generates kits from contact sheets without spatial layout. The system is implemented as generated ComfyUI graphs (up to 24 gated item branches) plus a Blender importer that consumes a versioned `layout.json` contract.

1. Motivation

1.1 Failure modes of naïve photo→3D scene pipelines

End-to-end image-to-3D on a full photo produces a single fused mesh that is hard to edit, poorly separated by object, and especially bad on **large planar surfaces** (roads warp into bowls; walls become lumpy sheets). Cropping every SAM mask and sending each to Hunyuan improves object quality but:

1. Occluded crops lack geometry Kontext could restore.
2. Ground and walls still waste GPU budget as “objects.”
3. Placement without a shared depth camera drifts.
4. Graphs fixed at N slots fail when the photo has fewer than N detections.
5. Blender still receives a blank world — reconstructed geometry without matching light.

1.2 Design thesis

Objects are generative; planes are measured; light is estimated.

Treat each class with the cheapest correct method, and define a narrow JSON contract so Comfy and Blender stay independently testable.

2. System architecture

2.1 Dual-repository layout

Layer	Repository	Responsibility
Graph + inference	ComfyUI_Porta19_Hy3D	Segmentation, Kontext, Hunyuan, depth, plane fit, env analyze, layout write
Authoring / DCC	porta19_world (Blender)	Queue, watch, import, urban mesh, Nishita/sun/weather, props, UI

Workflows are **generated** from a hand-validated 12-item Comfy export (`photo_to_scene_full_12`), expanded to 24 branches by `scripts/build_scene_variants.py` . Regenerating from the older `build_photo_to_scene_workflows.py` is deprecated — it omits gates, depth, and global settings.

2.2 Pipeline graph

```
flowchart TD
    Photo[LoadImage] --> Segment[SceneSegment SAM3]
    Photo --> Depth[DepthAnythingV2]
    Segment --> Filter[SceneFilter]
    Filter --> Isolate[SceneIsolate]
    Isolate --> Kontext[Flux Kontext per item]
    Kontext --> ImgGate[SceneImageGate 1..N]
    ImgGate --> Shape[Hy3D 2mini or full]
    Shape --> MeshGate[SceneGate 1..N]
    Filter --> Surfaces[SceneSurfaces]
    Depth --> Surfaces
    Photo --> Env[SceneEnvAnalyze]
    Depth --> Env
    Surfaces --> Layout[SceneLayout]
    Env --> Layout
    Depth --> Layout
    Filter --> Layout
    MeshGate --> Layout
    MeshGate --> Assemble[SceneAssemble optional]
    Layout --> JSON[layout.json + depth.png]
    JSON --> Blender[Blender import]
```

2.3 Branch scaling and bypass

- `MAX_GATE_ITEMS = 24` on gate sockets and assemble wait inputs.
- Default `SceneItemPick.on_missing = skip` → `ExecutionBlocker` prunes the branch including `OUTPUT_NODE` sinks.
- Addon `_trim_photo_scene_branches(workflow, max_items)` deletes nodes whose titles encode indices $\geq$ `max_items` and clears numbered gate sockets **before** `/prompt` , so intent (user asked for 6) matches Comfy's schedule.

These three layers separate **intent** (trim), **reality** (skip), and **graph capacity** (24 sockets).

3. Semantic segmentation and class kinds

3.1 Kind-split vocabulary

Presets are not flat label lists. Each preset maps:

Kind	Pipeline fate
object	Isolate → Kontext → Hunyuan
surface	Plane fit → environment.planes
wall	Plane fit → environment.walls

Prompt blocks may mark kinds explicitly: road [surface]: road asphalt . Unedited default blocks defer to the preset combo (is_default_prompts).

3.2 Detection hygiene

- NMS runs separately for objects vs regions so large floors do not suppress chairs.
- Surfaces use a lower effective area floor so large regions survive.
- SKIP_ALWAYS removes people-related labels.
- Asset preset has empty surface/wall vocabularies and looser thresholds for thumbnail sheets.

4. Generative object path

4.1 Isolation and Kontext

SceneIsolate writes black-background crops and full-frame masks. Each crop branch runs Flux Kontext to restore occluded regions before shape. **Mesh-only variants strip TexGen only** — never Kontext. This was the root cause of the prior 2mini “incomplete crop” regression.

4.2 Shape backends

Backend	Model	Typical steps	TexGen
2mini Turbo	hunyuan3d-2mini-turbo-fp16 + mini VAE	5	No (build-time mesh graph)
Hy3D v2	hunyuan3d-dit-v2-0-fp16	higher	Optional separate variants

Windows surface extraction prefers marching cubes (mc) when DiffDMC/ diso is unavailable.

4.3 Shared settings fan-out

SceneGlobalSettings drives steps / octree / face budgets into per-item nodes via links. Addon detail patching is **link-aware** (_set_unlinked) so widgets do not sever fan-out.

5. Geometric surface reconstruction

5.1 Camera and unprojection

Pinhole camera from FOV (+ optional ground pitch/height from depth). Metric depth from relative DepthAnything maps via `(z_near, z_far, is_inverse)`.

`unproject_points` keeps reconstructed height (unlike `unproject_to_ground`, which drops to `y=0`). Plane fitting requires that freedom for walls and tilted roads.

5.2 RANSAC and classification

Per surface/wall mask (`p9_scene/surfaces.py`):

1. Subsample mask pixels (deterministic stride).
2. Unproject to world.
3. Seeded RANSAC plane + least-squares refine on inliers.
4. Classify by normal vs up-axis: - $|n \cdot up| \geq 0.85 \rightarrow \text{ground / surface}$ - $|n \cdot up| \leq 0.45 \rightarrow \text{wall}$
Gap rejects ambiguous fits.
5. Convex hull in plane basis $\rightarrow$ Douglas–Peucker simplify $\rightarrow$ polygon (≤ 16 verts).
6. Geometry overrides mislabelled kind (horizontal “wall” $\rightarrow$ surface).

Sky masks contribute coverage / horizon metadata, not planes.

5.3 Layout environment schema (v3)

```
environment.planes[] = { label, kind, polygon[[x,z]], y, normal, area, ... }
environment.walls[]  = { label, polygon[[x,y,z]], normal, thickness, height, ... }
environment.ground   = AABB fallback from planes or item span
environment.sky       = optional coverage stats
environment.lighting  = phase-5 block (may arrive from EnvAnalyze)
depth.png            = 16-bit relative depth beside layout.json
```

Readers of v2 still find `ground` and box walls. Assemble extrudes polygons when present, else boxes.

5.4 Blender urban mesh

`portal9_world/core/poly2d.py` (bpy-free): area, inset, ear-clip triangulation, PCA axis, dash/stall/zebra generators.

`portal9_world/mesh/urban.py` : extruded slabs, curb inset to reduce z-fight, marking materials, wall solids in plane basis, Y-up $\rightarrow$ Z-up remapping.

6. Environment automation

6.1 Estimation (`SceneEnvAnalyze`)

Pure-numpy estimates into `environment.lighting` :

Field	Signal
Indoor vs outdoor	Sky mask + class mix + preset
Sun elevation	Sky luminance gradient / horizon
Sun azimuth	Contact-shadow direction at item UVs
Color temperature	Highlight / sky white balance → Kelvin
Weather	Saturation, contrast, depth falloff → clear / overcast / rain / fog / snow
Time of day	Elevation + warmth → night / dawn / day / dusk / golden

Widget overrides allow correcting a bad estimate without re-running shape.

6.2 Realization (`env_build.py`)

- **Nishita** `ShaderNodeTexSky` (or flat / indoor dim).
- **SUN** lamp with matching elevation/azimuth (coordinate remap) and softer angle for overcast.
- Weather forwarded to existing `build_atmosphere()` ; **snow** added beside rain in Geometry Nodes (`gn_env.apply_snow`).
- Photo camera extended: render resolution from aspect, sensor fit, DOF focus near median item depth.
- Camera rig operator: photo match, orbit turntable, three-quarter hero, top-down ortho.

7. Asset Bundle mode

Layout-free graph: Segment → Filter → Isolate → Kontext → Shape → SaveMesh.

No depth, surfaces, layout, assemble, or mesh gate.

Blender `import_asset_bundle` places meshes on a growing square grid, pivots to footprint, sits on z=0, scales to target size. Job id defaults to `assets_bundle` .

8. Procedural urban props and fallback

New `PropKind` builders: trashcan, dumpster, streetlight, traffic sign/light, hydrant, bollard, parking meter, bus stop, mailbox, modern planter, curb.

On scene import, if a layout item lacks a usable mesh (or the imported mesh is below a face threshold) and the label maps to a known prop, a procedural stand-in is placed with the item transform — preferring a readable prop over a hole.

9. Reliability engineering

Mechanism	Purpose
Template expansion builder	Preserve staged architecture (gates, Kontext, VRAM cleanups)
Structural checker	Dangling refs, contiguous branches, Kontext integrity, 2mini VAE routing, surfaces wiring, unknown input names
Build-time mesh-only variants	Avoid runtime TexGen strip dangling waits
Link-aware detail patch	Preserve GlobalSettings fan-out
Defensive import	Environment / fallback failures print and continue
Headless tests	Presets, surfaces, env math, poly2d, workflow prep, Blender urban/env smokes

10. Coordinate systems

Portal9 layout space is **Y-up, look +Z**, ground near `y=0` .
Blender is **Z-up**. Conversion used throughout import:

$$[(x, y, z)_{\mathrm{P9}} \mapsto (x, -z, y)_{\mathrm{Blender}}]$$

Euler remapping follows the same convention for camera and item transforms. Sun direction from EnvAnalyze is remapped before aiming `P9_Sun` .

11. Performance and operational limits

- **VRAM / time:** Cost scales roughly with $\text{Max Items} \times (\text{Kontext} + \text{shape})$. Prefer trimming over relying on blockers alone for honesty in the progress UI.
- **SAM recall:** Surface/wall concepts are imperfect; plane fitting degrades gracefully when masks are missing.
- **Sequential batching:** Not required yet; if 24 parallel branches exceed practical budgets, a future option is two passes of 12 rather than changing the contract.
- **Quality vs speed:** 2mini for layout iteration; full Hy3D (+ TexGen) for hero props.

12. Evaluation strategy

Level	Method
Unit	Synthetic depth planes for RANSAC; polygon inset/triangulation; lighting estimators
Structural	<code>check_scene_workflows.py</code> on all generated API/UI graphs
Integration (headless Blender)	Urban mesh markings; Nishita/sun; urban prop builds
Production	Artist review of <code>layout.json</code> planes/lighting + viewport against source photo

Automated suites do not replace a live Comfy photo run for SAM/Kontext/Hunyuan quality.

13. Related work (positioning)

This system is a **hybrid DCC pipeline**, not a single neural scene model. It combines open segmentation (SAM-family), diffusion completion (Flux Kontext), feed-forward mesh generation (Hunyuan3D), classical multi-view geometry (plane RANSAC), and procedural DCC authoring (Blender). Closest industrial analogues are “select → generate asset → place” tools; V2’s contribution is the **kind-routed graph**, the **layout.json v3 contract**, and **automatic light/weather transfer** into an editable scene graph.

14. Conclusions

Scene Pipeline V2 demonstrates that photo-to-scene for production authoring improves when:

1. Objects and planes use different reconstructors.
2. Occlusion completion is kept independent of texture painting.
3. Branch count is both capacity (24) and intent (trim) plus reality (skip).
4. A versioned layout file — not a fused GLB alone — is the integration surface.
5. Environment cues travel with geometry so Blender does not start from a blank lookdev.

Version **0.3.0** packages these pieces for Portal Worlds Studio and the Portal9 Hy3D Comfy pack.

Appendix A — Key modules

Module	Role
p9_scene/presets.py	Kind-split vocabularies
p9_scene/segment.py	SAM3 + region/object split
p9_scene/surfaces.py	RANSAC planes / walls
p9_scene/environment.py	Lighting estimation
p9_scene/layout.py	Photo-match layout + depth persist
p9_scene/assemble.py	Polygon-aware scene.glb
scene_nodes.py	Comfy node bindings
scripts/build_scene_variants.py	Graph generation
portal9_world/comfy/workflows.py	Patch, trim, queue prep
portal9_world/scene_import.py	Import, camera, fallback
portal9_world/mesh/urban.py	Roads / walls
portal9_world/mesh/env_build.py	Nishita / sun / weather
portal9_world/core/poly2d.py	Planar geometry helpers

Appendix B — Workflow stems

Stem	Role
photo_to_scene_<preset>_24	Full Hy3D + TexGen
photo_to_scene_<preset>_24_mesh	Hy3D mesh-only
photo_to_scene_<preset>_2mini	2mini Turbo mesh-only + Kontext
photo_to_assets_24	Asset sheet, 2mini
photo_to_assets_24_hy3d	Asset sheet, Hy3D

Presets for scene stems: room , indoor , outdoor , city , town .

Appendix C — Document set

- [SCENE_PIPELINE_V2_SUMMARY.md](#) — product overview
- [SCENE_PIPELINE_V2_HOWTO.md](#) — operator guide
- This white paper — architecture and rationale