

POV-Ray Photons: Actual Behavior (Source-Verified)

Generated for: William Walker — Date: 2025-12-11

This document summarizes the actual behavior of photons in POV-Ray based on the current source code (notably *source/core/lighting/photons.cpp* and related tasks) and official documentation. It reconciles practical observations from the POV-Ray forums with the implemented logic in the renderer.

1) Photon Mapping in POV-Ray (two-pass)

POV-Ray implements photon mapping as a two-pass system: (a) a photon shooting pass emits photons from light sources, traces them through the scene, and stores them on surfaces and/or in media; (b) the render pass gathers nearby photons to estimate indirect/caustic illumination. [Docs: 3.6/3.7 Quick Reference] References: POV-Ray 3.7/3.6 docs on Photons; GitHub sources for photon modules.

2) Targets vs. Pass-Through Objects (Pre-Target Logic)

When a photon ray is aimed at a **target** object, POV-Ray enforces special pre-target rules in *photons.cpp*: • If the ray hits an object marked as **pass_through** before the target, the photon ray continues. With the macro **PT_FILTER_BEFORE_TARGET** enabled, such objects can tint the photon color. • If the ray hits a **non-pass-through** object that is not the target, the photon ray is terminated and does not continue toward the target. • Upon hitting the target, subsequent behavior (reflection/refraction, storage) depends on merged flags from the light and object.

```
// photons.cpp (excerpted behavior, paraphrased)
#define PT_FILTER_BEFORE_TARGET // pass_through before target tints photons
// #define PT_AMPLIFY_BUG // old v3.6 amplifier (disabled in modern builds)

// In PhotonTrace::TraceRay():
// - Detect PH_TARGET_FLAG and PH_PASSTHRU_FLAG
// - If pre-target hit is pass_through: continue, possibly tint color
// - If pre-target hit is non-pass-through and not the target: stop tracing
```

3) Reflection/Refraction: Merged Flags Decide Behavior

Whether photons are reflected and/or refracted at a target is decided by **merged flags** from the light and object, computed via *LightTargetCombo::computeMergedFlags()* in the estimation/shooting code. POV-Ray requires **ON** without an overriding **OFF**: • Reflection: *PH_RFL_ON_FLAG* set and *PH_RFL_OFF_FLAG* not set. • Refraction: *PH_RFR_ON_FLAG* set and *PH_RFR_OFF_FLAG* not set. If neither condition is met, photons are not shot for that light+object combo (early bail-out).

4) Defaults & Categories (Show vs. Cause Caustics)

Per the docs: • **Show caustics** (storage): All surfaces store photons by default; opt out with *photons{ collect off }*. • **Cause caustics**: Objects do not cause caustics by default; make them *target* and enable *reflection on/refraction on* in the object's photons block, and configure the light likewise.

5) Media Photons (Volumetric Beams)

Media photon interaction is off by default. Enable it in *global_settings { photons { media steps[, factor] } }* and set the media container to *photons { pass_through }* so photons traverse into the volume. A dedicated media photon map and gatherer are used in rendering.

6) Estimation, Spacing & Density

During estimation, POV-Ray approximates photon counts and spread using target *Ph_Density*, global *surfaceSeparation*, and shooting geometry (radius, distance). An internal value similar to $x = (rad / (density * separation))^2 * \pi$ contributes to the count estimate, which then rescales the separation.

7) Storage Structure & Acceleration

The photon map stores photons in fixed-size blocks ($2^{14} = 16384$ photons per block), managed via *PhotonMap::PhotonBlock*. Backend tasks handle sorting and gathering; core acceleration utilities (e.g., octree) are used where appropriate.

8) Current Caveats (Betas, Light Models)

POV-Ray 3.8 betas: some users report broken photon behavior with the new true inverse-squared lighting, while legacy fading behaves as expected. Also, modern builds intentionally disable the historical v3.6 'photon amplifier' behavior.

9) Practical Setup Patterns

- Make an object **cause** caustics: object photons { target reflection on refraction on collect off } and light photons { reflection on refraction on }.
- Media beams: global_settings { photons { media steps } } and media container photons { pass_through }.
- Pre-target filters: keep default PT_FILTER_BEFORE_TARGET behavior so pass-through panes tint but do not block the beam.
- Tune spacing/count and radius/gather based on scene scale and desired detail vs. noise.

References (URLs)

- POV-Ray source: photons.cpp — <https://github.com/POV-Ray/povray/blob/master/source/core/lighting/photons.cpp>
- POV-Ray source: photon estimation task — <https://github.com/POV-Ray/povray/blob/master/source/backend/lighting/photonestimationtask.cpp>
- POV-Ray Documentation 3.6: Media & photons — <https://www.povray.org/documentation/view/3.60/101/>
- POV-Ray Documentation 3.6: Using Photon Mapping — <https://www.povray.org/documentation/view/8.2.0/425/>
- POV-Ray Wiki: Reference: Photons — <https://wiki.povray.org/content/Reference:Photons>
- Beta test thread (inverse squared lighting & photons) — <https://news.povray.org/povray.beta-test/thread/%3C66732e0a%40news.povray.org%3E/>