



User Guide

Camera matching from a single photo. Trace the edges, get the camera — ready for Blender, Maya, Houdini and every USD application.

Version 0.1 · October 2026

Free and open source · GPL-3.0-or-later

Made by Gelain with AI assistance

github.com/fugaapp/fuga

A camera, from the lines in a photo.

Every photo of a building, a room or a street holds lines that are parallel in the real world. In the picture they converge on **vanishing points** — *pontos de fuga*, which is where the name comes from.

Trace a few of those edges and Fuga works out the camera that took the photo: its **focal length**, **rotation** and **position**, and the scale of the scene if you know one real distance. Export it, and your 3D software gets a camera that lines up with the photo exactly — ready for modelling on top of a reference, set extensions, product shots or architectural previews.

It is a tool for 3D generalists, motion designers, architects and VFX artists, built in the spirit of **fSpy** and rewritten from scratch for precision, with the problems its users reported fixed.

Fuga was designed and directed by Gelain, a motion designer and 3D generalist, and **built with the help of AI**. Every result is checked by automated tests, and the source is public.

10^{-9}

relative error recovering a known camera in the test suite

∞

lines

per vanishing point, combined by least squares

0

bytes

of your data collected. No telemetry, ever.

CONTENTS

Installing	03
A tour of the window	04
Matching a camera, step by step	05
Choosing a method: 1, 2 or 3 points	07
Exporting to your 3D software	08
Features and precision	09
Appearance and privacy	10
Shortcuts and troubleshooting	11
Thank you	12

Download, open, done.

Installers for every platform are on the **Releases** page: github.com/fugaapp/fuga/releases. Fuga is free; installers are not signed with a paid certificate, so your system asks once before the first launch.

Windows

.exe

1. Download Fuga-x.y.z-win-x64.exe and run it.
2. If you see *"Windows protected your PC"*, click **More info** → **Run anyway**.
3. Choose a folder and whether to install for all users.

macOS

.dmg · Apple silicon + Intel

1. Download the arm64 (M-series) or x64 (Intel) DMG and drag Fuga to Applications.
2. On first launch macOS says it can't be opened: go to **System Settings** → **Privacy & Security** and click **Open Anyway**.

Linux

.AppImage

1. Download Fuga-x.y.z-linux-x86_64.AppImage.
2. Make it executable: `chmod +x Fuga-*.AppImage`, then double-click it.
3. If your distribution blocks the sandbox, start it with `--no-sandbox`.

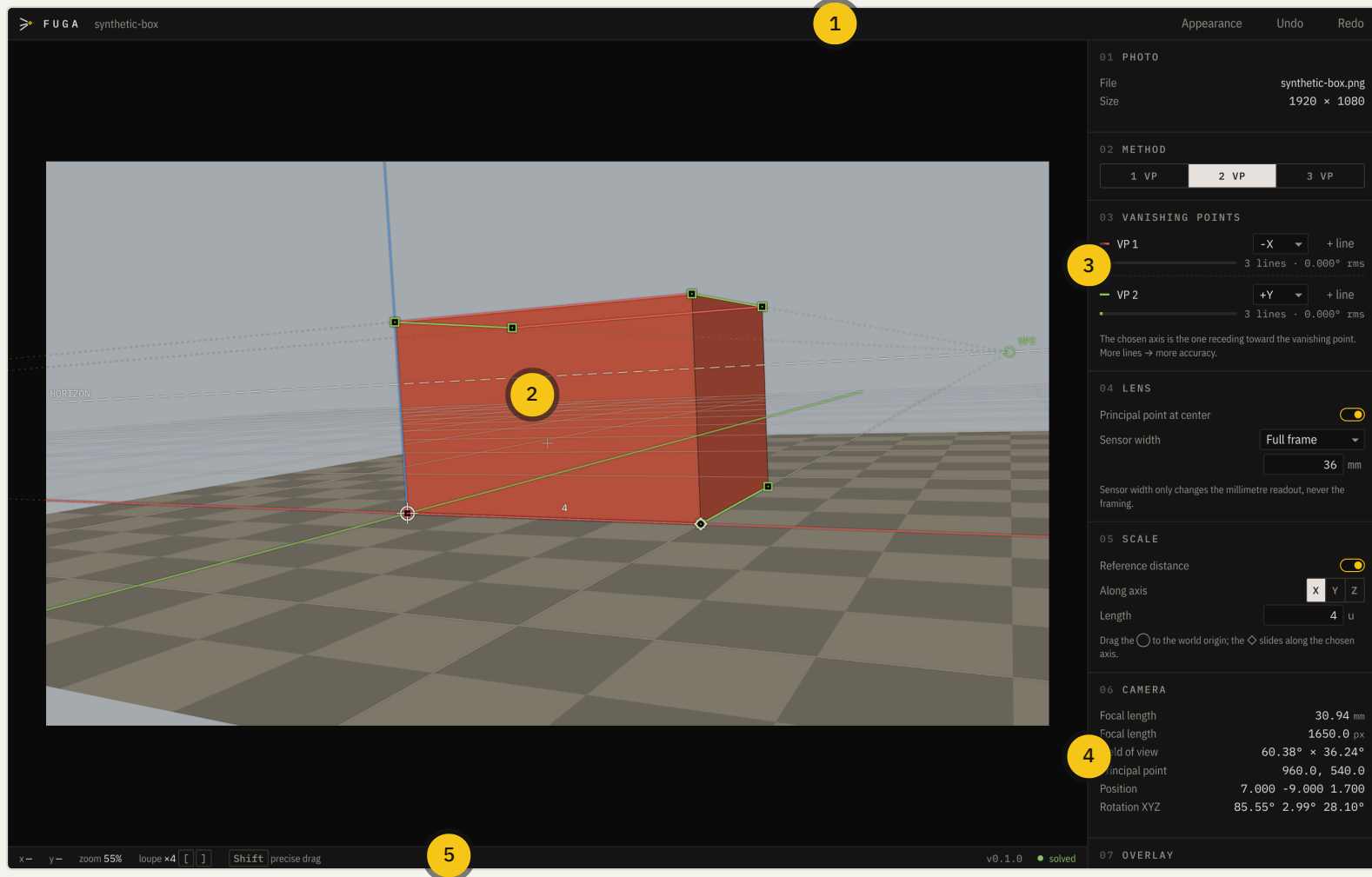
Staying up to date

When a new version is published, Fuga shows a yellow notice in its top bar — **Fuga 0.2.0 available** — that opens the download page. It never installs anything by itself. You can turn the automatic check off in **Help** → **Check for Updates Automatically**.

Running from source

```
# Node.js 22.12 or newer
git clone https://github.com/fugaapp/fuga.git
cd fuga && npm install
npm run dev      # open the app
npm test         # precision tests
```

03 A TOUR OF THE WINDOW



1 Top bar

Project name, update notice, Appearance, Undo / Redo.

2 Viewport

Your photo, the line groups, ground grid and horizon. Scroll to zoom, drag empty space to pan.

3 Inspector

Method, vanishing points with a quality meter, lens, scale, overlay, export.

4 Camera

The live result: focal length in mm and px, field of view, position, rotation.

5 Status bar

Cursor in image pixels, zoom, loupe level, and whether the camera is solved.

01 Open a photo

Drag it onto the window or press `Ctrl/⌘ 0`. PNG, JPG, WEBP, AVIF, BMP and GIF all work. Use the original file: cropping or resizing is fine, but avoid photos that were straightened or perspective-corrected in an editor.

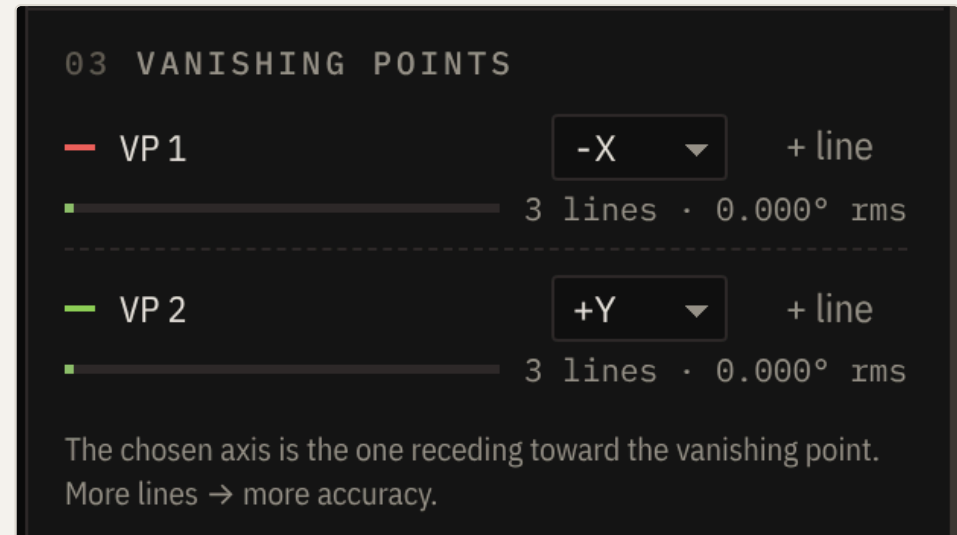
02 Choose a method

2 VP suits most architecture and interiors. Use **3 VP** when vertical edges also converge (shot looking up or down), and **1 VP** when only one direction converges and you know the focal length. Page 07 explains each.

03 Trace parallel edges

Each colour is one group of edges that are parallel in the real world. Drag the square handles onto them; drag a line's body to move it whole. Then pick, for each group, the world axis that **recedes from the camera** toward its vanishing point.

Long edges, far apart, give the best result. Every extra line (+ line) makes the camera more accurate — fSpy only ever used two.



Each group shows its line count and how much its lines disagree, in degrees. A short green bar is good; past 0.35° Fuga warns you.

Reading the overlay

- **Coloured lines** — your traced edges, X red, Y green, Z blue; dashed extensions run to the vanishing point.
- **Grid and horizon** — the solved ground plane. If it sits on the floor of your photo, the camera is right.
- **○ origin, ◇ reference, + principal point** — see the next page.

04 Set the origin and the scale

Drag the  to the point that should become (0, 0, 0) in your 3D scene — a floor corner is ideal. For real-world scale, turn on **Reference distance**, choose the axis, type a known length (a 2.1 m door, a 0.6 m tile) and drag the  to the other end. It slides along the axis by itself.

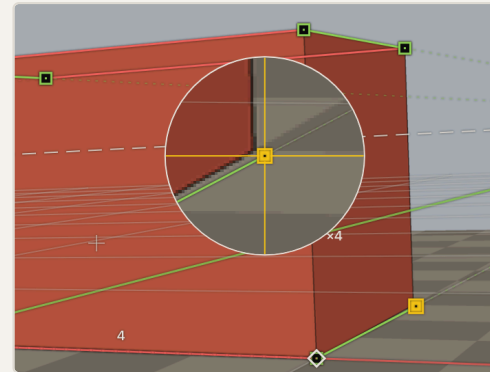
05 Check

The grid should lie flat on the floor and the horizon should pass through the vanishing points. Zoom in and refine the endpoints: watch the camera values settle and the quality meters shrink.

Precision tools: hold **Shift** while dragging for 10× finer movement, nudge the selected point with the **arrows**, and change the loupe magnification with **[]**.

06 Export

One click in **Export**. Then save the project (**Ctrl/⌘ S**) to come back to it later — the .fuga file is small, readable JSON.



While you drag, the loupe magnifies real pixels — and your lines.

06 CAMERA		
Focal length	30.94 mm	
Focal length	1650.0 px	
Field of view	60.38° × 36.24°	
Principal point	960.0, 540.0	
Position	7.000 -9.000 1.700	
Rotation XYZ	85.55° 2.99° 28.10°	

The solved camera updates live.

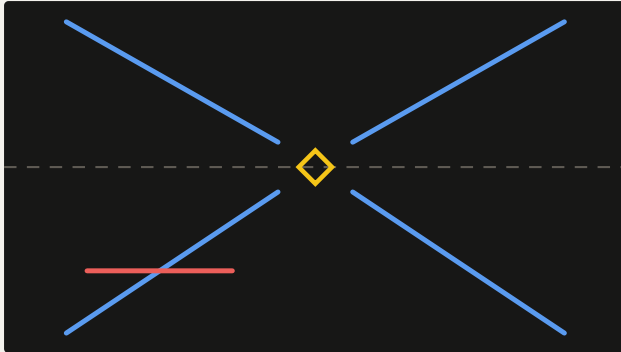
Undo anything

Every change, including each whole drag, is one step of undo. **Ctrl/⌘ Z** and **Ctrl/⌘ Shift Z** — or the buttons in the top bar.

Sensor size

The sensor width only changes how the focal length reads in millimetres; the framing is identical. Pick your camera's sensor (Full frame, APS-C...) so the number matches the lens you used.

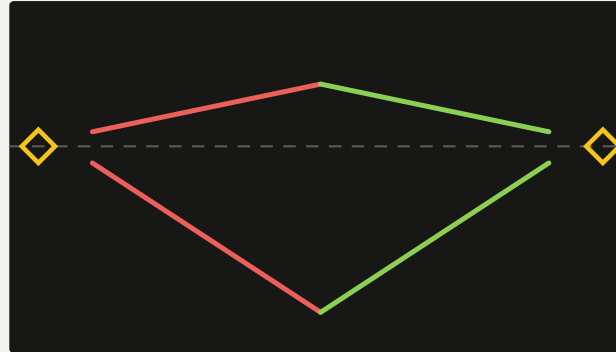
One, two or three vanishing points.



1 VP — one direction converges

Corridors, frontal façades, a street seen straight on. One group converges; one extra line gives the second direction (its arrow points along the positive axis). The focal length cannot be measured from one point, so you type it in.

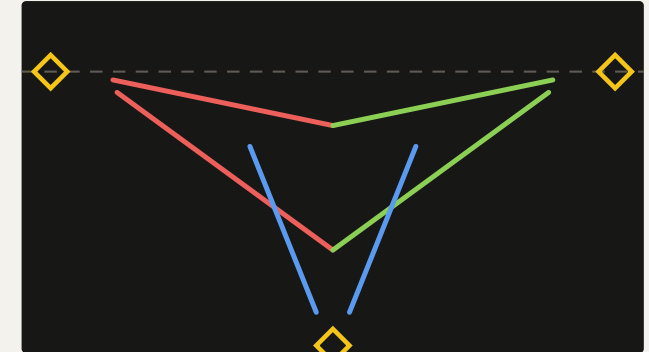
SOLVES rotation · position · scale
NEEDS a known focal length



2 VP — the usual case

A building corner, a room, a product on a table. Two groups of perpendicular edges, each with its own vanishing point. Fuga measures the focal length from the angle between them.

SOLVES focal length · rotation · position · scale
ASSUMES the principal point is the image centre
(or where you put it)



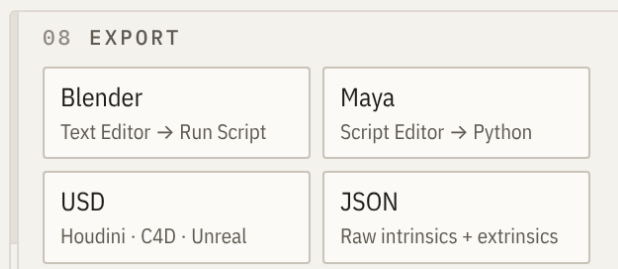
3 VP — verticals converge too

Shots looking up at a tower or down into a courtyard. Three groups, one per axis. Fuga also finds the principal point, so cropped or lens-shifted photos come out right.

SOLVES focal length · principal point (lens shift) · rotation · position · scale

Straight into your scene.

Every export writes the camera's full transform matrix — never Euler angles — so there is no rotation-order ambiguity and no camera upside down. Resolution, focal length, sensor and lens shift come along.



Exports are tested by re-projecting points through each application's own camera model.

Blender

.py

1. Open the **Scripting** workspace.
2. **Text** → **Open** the exported file, then **Run Script** (**Alt P**).
3. A camera appears with the photo as its background, set as the scene camera, with the render resolution matched.

No add-on to install, so nothing breaks between Blender versions. Written for Blender 5.1.

Maya

.py

1. Open the **Script Editor**, Python tab.
2. **File** → **Load Script**, then **Execute**.
3. You get a camera with the correct film back and film offset, the render resolution, and the photo as an image plane.

USD

.usda

1. Import or reference it in **Houdini** (Solaris), **Cinema 4D**, **Unreal**, **Omniverse**, Blender or Maya.
2. The camera is Y-up, in metres, with focal length and apertures in millimetres.

Checked with Pixar's own USD library: points land within a millionth of a pixel.

JSON

.json

1. For Nuke, custom pipelines and photogrammetry tools.
2. Contains the intrinsic matrix K, the OpenCV rotation and centre, and camera-to-world matrices for both Z-up and Y-up worlds, each labelled with its convention.

Built for the last pixel.

Unlimited lines

Any number of edges per vanishing point, combined by least squares and weighted by how much each one can be trusted.

Quality you can see

A meter per vanishing point shows how far its lines disagree, in degrees, and Fuga warns before a bad line spoils the camera.

Loupe that shows lines

×2 to ×32 magnification of real pixels with your overlay drawn inside it.

Fine control

Zoom to 6400 %, **Shift** for 10× finer drags, arrow-key nudges, crisp pixels when zoomed in.

Full undo

Every change is undoable; a whole drag is a single step.

Lens shift

3-VP mode solves the principal point, for cropped and shift-lens photos.

Readable projects

Versioned JSON with the photo path stored relative to the project — move the folder, it still opens.

Light, dark, yours

System, light or dark theme, and your own accent and axis colours.

~10⁻⁹

Focal length, rotation and position recovered from a synthetic camera with exactly known values.

<10⁻⁶ px

Reprojection error of exported files, read back through Blender's, Maya's and USD's camera models.

-30 % error

at least, using 10 lines per vanishing point instead of 2, with realistic click noise ($\sigma = 0.7$ px).

Make it yours.

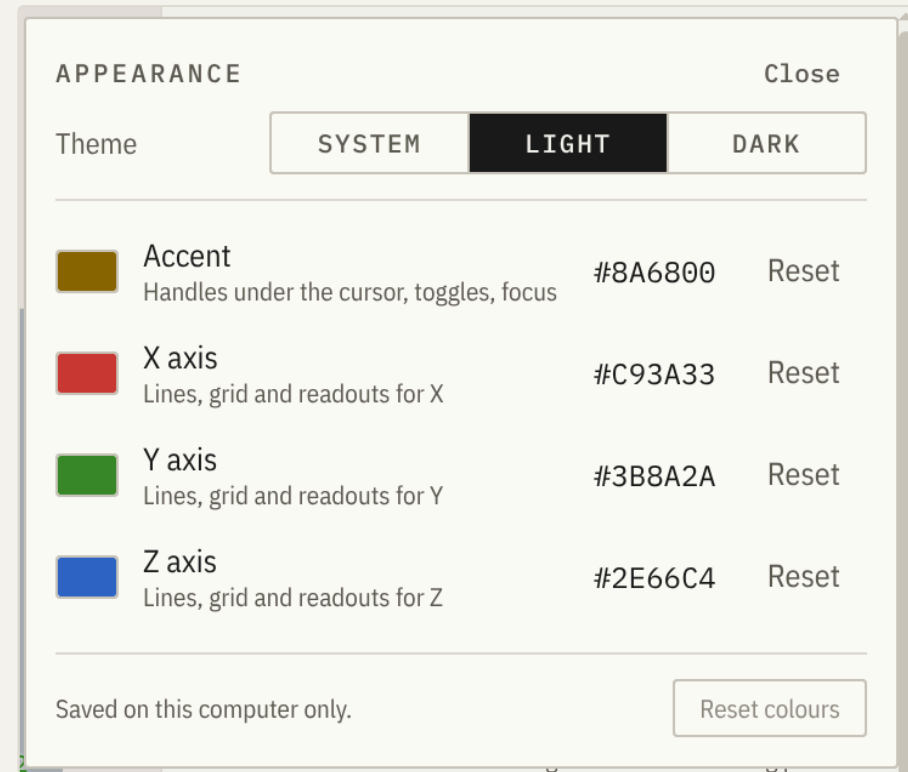
Open **View** → **Appearance...** (`Ctrl/⌘ ,`) or the **Appearance** button in the top bar.

- **Theme** — System follows your OS; Light and Dark override it. Menus and dialogs follow too.
- **Accent** — the colour of whatever is under your hand: dragged handles, toggles, focus.
- **X, Y, Z** — line and grid colours. Pick colours that stand out against your photo.
- **Reset** any colour, or all of them, to the theme default.

Your data stays yours.

Fuga collects nothing — no analytics, no telemetry, no account. The app blocks every network request in code except one: an optional check on GitHub for a newer version, which you can switch off in **Help**. Your photos are never copied or uploaded, and the app asks for no camera, microphone or location access.

Full details: `PRIVACY.md` and `SECURITY.md` in the repository.



The Appearance panel, light theme. Preferences are saved on your computer only.

Shortcuts

Open photo / project	<code>Ctrl/⌘ O</code> · <code>Ctrl/⌘ Shift O</code>
Save / Save as	<code>Ctrl/⌘ S</code> · <code>Ctrl/⌘ Shift S</code>
Undo / Redo	<code>Ctrl/⌘ Z</code> · <code>Ctrl/⌘ Shift Z</code>
Zoom	wheel or pinch, around the cursor
Pan	drag empty space
Fit photo / actual pixels	<code>F</code> or <code>Ctrl/⌘ 0</code> · <code>Ctrl/⌘ 1</code>
Fine drag	hold <code>Shift</code>
Nudge selected point	<code>arrows</code> 1 screen px · with <code>Shift</code> 0.1 px
Loupe magnification	<code>[</code> <code>]</code>
Delete selected line	<code>Delete</code>
Appearance	<code>Ctrl/⌘ ,</code>

When something looks wrong

"Vanishing point N is at infinity"	That group's lines are parallel in the photo. Pick edges that visibly converge, or switch to 1 VP.
"...cannot come from perpendicular directions"	A group mixes edges from different directions, or both groups follow the same one. Re-check which edges belong together.
"Two vanishing points are assigned to the same world axis"	Give each group a different axis in the inspector.
Camera upside down or mirrored	Flip the sign of an axis (+X ↔ -X). The chosen axis must recede from the camera toward its vanishing point.
"Lines ... disagree by 0.4° RMS"	One line is off, or the lens bends straight lines (wide angle). Zoom in, refine, or remove the outlier with <code>Delete</code> .
Scale looks wrong	Check the  origin sits exactly on the corner and the  reference on the far end of the known length.
"Photo not found" when opening a project	The photo moved. Keep it next to the <code>.fuga</code> file — paths are stored relative to it.
Something else	Report it: Help → Report a Problem.

A gift to the community.

Fuga is free and open source, and it will stay that way. If it saves you an afternoon of nudging a camera by hand, it has done its job.

Use it for anything, including commercial work. Share improvements back under the same licence (GPL-3.0-or-later). Ideas, bug reports and pull requests are welcome on GitHub.

Standing on the shoulders of

- **fSpy** by Per Gantelius (stuffmatic) — the free tool that made camera matching accessible to everyone, and the inspiration for Fuga. Its users' reports shaped this design.
- **Electron, Chromium, React, Vite, TypeScript** — the open-source foundations of the app.
- **IBM Plex** by IBM — the typefaces, under the SIL Open Font License.
- **OpenUSD** by Pixar — used to verify the USD export.
- **Everyone who tests, reports and shares** — you make it better.

How it was made

Fuga was designed and directed by Gelain, a motion designer and 3D generalist, to find the exact camera behind a photo (its lens, angle and position) and bring it into 3D software, so the scene lines up with the picture without trial and error. It was built with the help of AI. Every result is checked by automated tests against cameras with known values, and the source is public for anyone to review.

Made by Gelain with AI assistance

github.com/fugaapp/fuga

Issues · github.com/fugaapp/fuga/issues

