**3D distance from a 2D bounding box**.

**1. Camera projection model**

A camera is modeled (pinhole camera model) as:

s[uv1]=K [XYZ]s​uv1​​=K​XYZ​​

where:

* (u,v)(u,v) = pixel coordinates on the image
* (X,Y,Z)(X,Y,Z) = real-world coordinates (meters)
* KK = intrinsic matrix (depends on focal length, sensor size, resolution)
* ss = scale factor

The key part: the further away an object is (ZZ large), the **smaller it appears in the image**.

**2. Focal length in pixels**

We usually convert focal length to "pixels" (so that we can directly relate pixels to meters):

fpx=fmm⋅NpxWsensor,mmfpx​=fmm​⋅Wsensor,mm​Npx​​

where:

* fmmfmm​ = focal length (mm)
* NpxNpx​ = image width in pixels
* Wsensor,mmWsensor,mm​ = physical sensor width (mm)

This gives us fpxfpx​, the effective "magnification" of the camera in pixels.

**3. Relationship between object size and distance**

Suppose the object has a **known real-world size** SrealSreal​ (e.g. wingspan = 30 m).

If its projection in the image has a **measured pixel size** SpxSpx​:

Z=fpx⋅SrealSpxZ=Spx​fpx​⋅Sreal​​

This formula comes directly from similar triangles:

* At distance ZZ, the object subtends an angle θ=SrealZθ=ZSreal​​.
* On the image plane, this angle projects to Spx=fpx⋅SrealZSpx​=fpx​⋅ZSreal​​.
* Rearranging gives the distance:

Z=fpx⋅SrealSpxZ=Spx​fpx​⋅Sreal​​

That’s the **depth (distance from camera along Z-axis)**.

**4. Recovering X, Y (lateral position)**

The center of the box (u,v)(u,v) can be converted to real-world meters:

X=(u−cx)⋅Zfpx,Y=(v−cy)⋅ZfpxX=fpx​(u−cx​)⋅Z​,Y=fpx​(v−cy​)⋅Z​

where:

* cx,cycx​,cy​ = image center (principal point, usually image\_width/2, image\_height/2).

This shifts the measurement so that the camera’s optical axis is the origin.

**5. Why use box length?**

The bounding box gives us two useful measurements:

* **long side** (e.g., plane length when seen side-on, wingspan when seen from above)
* **short side** (the other dimension).

If the plane is tilted, one side may shrink more due to perspective. By using **both** (long and short), and comparing them to the plane’s real dimensions, we can average or cross-check.

That’s why in my function I used both longer and shorter sides:

Zlong=fpx⋅LrealLpx,Zshort=fpx⋅WrealWpxZlong​=Lpx​fpx​⋅Lreal​​,Zshort​=Wpx​fpx​⋅Wreal​​

Then combined them (average) to get a more robust estimate.

✅ **Summary of the logic**

* Real-world size ↔ pixel size scales inversely with distance.
* Focal length converts this proportionality into actual distance.
* Bounding box center gives lateral offsets in X and Y.
* That’s how a **2D box → 3D position** mapping works under the pinhole camera model.