BTP

Camera calibration

optical & tangential distortions

Rotated
Translated
Tilted.

} computes camera matrix, distortion parameter, rotation and translational vectors.

(removes distortion)

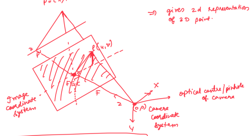
⇒ **Camera Pinhole model :-**

↳ Focal length is considered as scaling facto
↳ c_x, c_y → location of optical centre

3D world coordinate system

$$\underset{\text{scaling factor}}{w}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

⇒ gives 2d representation of 3D point.



Why Camera calibration ?

$\boxed{c_x, c_y \equiv (0,0)}$ (set by us)

1. Barrel distortion
2. Pincushion distortion

→ (caused by lenses of camera).

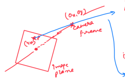
→ To remove distortion on our camera lens so that we get better frames & camera parameters.

Zhang's Method :-

↳ Linear solutions for pinhole camera model.

→ Take a bunch of photos of a chessboard grids at different angles and ~~planes~~. planes.

can we find 3D point corresponding to its projection in the image?



If I give you a single pt. P in the image, can we tell where the corresponding 3D point lies?
⟶ NO

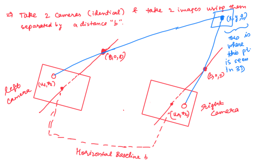
But vice versa is true.

i.e., we do know that the corresponding 3D point must lie on an outgoing ray.
⟶ after camera calibration, we know eqn of this ray.

$$\boxed{\begin{array}{ll} \text{3D} \to \text{2D} \ (\text{point}) \Rightarrow & u = f_x \dfrac{x_c}{z_c} + O_x \\ & v = f_y \dfrac{y_c}{z_c} + O_y \end{array}}$$

$$\boxed{\begin{array}{ll} \text{2D} \to \text{3D} \ (\text{eq}^n \text{ of outgoing ray}) \Rightarrow & x = z/f_x \, (u - O_x) \\ & y = z/f_y \, (v - O_y) \\ & z > 0 \end{array}}$$

⇒ Take 2 cameras (identical) & take 2 images using them separated by a distance "b".



1. $u_l = f_x\left(\frac{x}{z}\right) + O_x$
   $v_l = f_y\left(\frac{y}{z}\right) + O_y$

2. $u_r = f_x\left(\frac{x-b}{z}\right) + O_x$
   $v_r = f_y\left(\frac{y}{z}\right) + O_y$

f_x, f_y, b, O_x, O_y are known by camera setup.

from eq. ① & ②, we can get x & y & z.

$$(u_l, v_l) = \left(f_x \frac{x}{z} + O_x,\ f_y \frac{y}{z} + O_y\right)$$
$$(u_r, v_r) = \left(f_x\left(\frac{x-b}{z}\right) + O_x,\ f_y \frac{y}{z} + O_y\right)$$

⇒ solving for (x, y, z), we get

$$x = \frac{b(u_l - O_x)}{(u_l - u_r)}, \quad y = \frac{b f_x (v_l - O_y)}{f_y (u_l - u_r)}$$

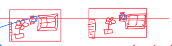
$$z = \frac{b f_x}{(u_l - u_r)}$$

Now, $(u_l - u_r)$ is called disparity)

$$z \propto \frac{1}{(u_l - u_r)} \quad \& \quad (u_l - u_r) \propto b$$

i.e., if we take images of an object located very very far away from both the cameras (infinity), then doesn't matter the baseline "b", we will get 2 identical images of the object.

Window Based Methods — Template Matching :-



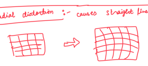
we take this window & move it along the horizontal line so that we find the exact same window in the right image.

we knows the the corresponding lies along the same horizontal line.

so, we can easily get disparity & using this we can get (x, y, z).

Consequences :-
1. we need textured surface.

Radial distortion :- causes straight lines to appear curved.



farther pts.
⇓
Radial distortion increases

$$x_{distorted} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{distorted} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

Tangential distortion :- occurs bcoz the image-taking lens is not ~~perfectly~~ aligned perfectly parallel to the imaging plane. (some areas look nearer than expected).

$$x_{distorted} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_{distorted} = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$

clearly, we need $(p_1, p_2, k_1, k_2, k_3)$.

Intrinsic parameters
⇓
specific to camera
Ex. focal length (f_x, f_y)
optical centre (c_x, c_y)

Extrinsic parameters
⇓
corresponds to rotation and translation vectors

↳ camera matrix $= \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$

Idea :- Take several images of a chessboard from a camera at different angles & positions.
Now, we will compare the corners of the chesscubes in all images.

Assumption :- chess board was kept stationary at XY plane, so $z = 0$ always & camera was moved accordingly..

Now, pass (0,0), (1,0), (2,0) for (X, Y).
lets say squares were 1mm.