f-theta camera model

NVIDIA

I. Introduction

The camera intrinsics transform points from image coordinates (pixels) to camera coordinates (metric). Each pixel in a 2D image represents an optical ray in 3D space. For central cameras, all of the optical rays begin at the same point: the optical center, i.e. the origin of the camera coordinate frame. Thus, each optical ray is determined by a single 3D direction r (a 3-vector with unit-norm). The intrinsic model of a camera is the mapping between a pixel position p and a ray r. This is represented by two functions:

$$p = ray2pixel(r)$$
$$r = pixel2ray(p),$$

where ray2pixel can accept ray-vectors of any magnitude, but pixel2ray always produces ray-vectors with unit-norm (because the depth of each pixel is unknown).

So for a 3D point in camera coordinates $X = [x, y, z]$ the relation is:

$$p = \text{ray2pixel}(X)$$
$$X = d \cdot \text{pixel2ray}(p),$$

where $d = |X|$ is the depth of the pixel.

II. Definitions

Principal point

There is a special point in the image that determines the direction the camera is looking at: the principal point, located at $[u_0, v_0]$. The ray $r_0 = \text{pixel2ray}([u_0, v_0])$ corresponds to the z-axis of the camera coordinate frame. It is often very close to the middle of the image.

III. f-θ model

Our model maps the distance a pixel makes with the principal point to an angle between camera ray and optical axis. The model allows to cover wide range of field of view. The model can be understood as a simplification of the OCAM model[Scaramuzza at al., 2006] but without the final affine mapping. Closest description in literature can be found in the work of Courbon at al.[Courbon et al., 2007] (see TABLE II).

i. Forward transformation (ray2pixel)

Start with ray $R = \begin{bmatrix} r_x \\ r_x \\ r_z \end{bmatrix}$ and compute θ, angle of ray with the optical axis, $\theta = cos^{-1}(r_z/|R|)$

The forward polynomial $f(\theta) = k_0 + k_1\theta + k_2\theta^2 + ... + k_n\theta^n$ maps the angle θ to the distance to principal point in image space. The coefficients of the polynomial are either obtained by fitting it to a manufacturer-provided distortion table or by performing checkerboard based intrinsic calibration procedure, with $k_0 = 0$ (always). Note that k_1 corresponds to the focal length. [1] [2]

[1] the shape of $f(\theta)$ is arbitrary (with the exception of k_0). It does not exactly follow the theoretical odd-only polynomial. The coefficient k_1 is the focal length

[2] DriveWorks rig files only store the coefficients of the backward polynomial $b(r)$, including the constant term 0. The forward polynomial is computed from it.

The pixel coordinates are obtained as follows:

$$x = u_0 + f(\theta)\cos(\phi)$$
$$y = v_0 + f(\theta)\sin(\phi),$$

with u_0, v_0 the principal point and ϕ the angle between the x image axis and the projection of the ray on the image plane denoted by $R_p = \begin{bmatrix} r_x \\ r_y \end{bmatrix}$. When expressing ϕ as $\cos^{-1}(r_x/|R_p|)$ or $sin^{-1}(r_y/|R_p|)$, the pixel coordinates can be expressed as:

$$x = u_0 + f(\theta) \cdot r_x/|R_p|$$
$$y = v_0 + f(\theta) \cdot r_y/|R_p|$$

ii. Backward transformation (pixel2ray)

There is no closed-form inverse model, so it is approximated with a second polynomial, named backward polynomial here.

Start with pixel (distorted) $P_d = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} x - u_0 \\ y - v_0 \end{bmatrix}$, with u_0, v_0 the principal point. The backward polynomial $b(r) = j_0 + j_1 r + j_2 r^2 + ... + j_n r^n$ maps the distance to principal point in image space r to the angle $\theta = b(r_d) = b(|P_d|)$ of the ray with the optical axis. The coefficients of the polynomial are obtained by fitting it to a manufacturer-provided distortion table or by performing checkerboard based intrinsic calibration procedure, with $j_0 = 0$ (always). [3]

The ray coordinates can be expressed as:

$$R = \begin{bmatrix} sin(\theta)cos(\phi) \\ sin(\theta)sin(\phi) \\ cos(\theta) \end{bmatrix}$$

with ϕ the angle between the x image axis and the projection of the ray on the image plane. When expressing ϕ as $cos^{-1}(p_x/|P_d|)$ or $sin^{-1}(p_y/|P_d|)$, the ray can be expressed as:

$$R = \begin{bmatrix} sin(\theta) \cdot p_x/|P_d| \\ sin(\theta) \cdot p_y/|P_d| \\ cos(\theta) \end{bmatrix}$$

References

[Courbon et al., 2007] Courbon, J., Mezouar, Y., Eckt, L., and Martinet, P. (2007, October). A generic fisheye camera model for robotic applications. *In Intelligent Robots and Systems, 2007. IROS 2007*, pp. 1683–1688.

[Scaramuzza at al., 2006] Scaramuzza, D., Martinelli, A. and Siegwart, R., (2006). A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion, *Proceedings of IEEE International Conference of Vision Systems (ICVS'06), New York*, 2006.

[3] The shape of $b(r)$ is arbitrary (with the exception of j_0) and the result of the fit to a distortion LUT. The coefficient j_1 is the inverse of the focal length.