# On camera model conversions

Eva Goichon, Guillaume Caron, Pascal Vasseur, Fumio Kanehiro

# On camera model conversions

Eva Goichon, Guillaume Caron, Pascal Vasseur, Fumio Kanehiro

***Abstract*— On the one hand, cameras of conventional field-of-view usually considered in computer vision and robotics are very often modeled as a pinhole plus possibly a distortion model. On the other hand, there is a large variety of models for panoramic cameras. Many camera models have been proposed for fisheye cameras, catadioptric cameras, and super fisheye cameras. But in both cases, few models offer the possibility of converting them into another model.**

**This paper contributes to filling this gap in, to allow an algorithm designed with a projection model to accept data of a camera calibrated with another model. So, a pre-existing data set can be used without having to recalibrate the camera. We provide the methodology and mathematical developments for three conversions considering three different types of cameras that are evaluated with respect to calibration and within a visual Simultaneous Localization And Mapping benchmark. The source code of the camera model conversions studied in this paper is shared within the libPeR library for Perception in Robotics: https://github.com/PerceptionRobotique/libPeR_base.**

## I. INTRODUCTION

As soon as geometry computation from images is concerned, projection models describing the geometrical image formation of the camera that captures the image are to be considered [1]. They can be implicitly hidden in the layers of neural networks in computer vision [2] or explicitly appearing in the equations of model-based 3D reconstruction methods in photogrammetry [3] and Simultaneous Localization And Mapping (SLAM) in robotics [4].

Most of the time, the cameras considered are of conventional field-of-view (FoV), designed by the manufacturer to fit the pinhole camera model within the depth-of-field, the part of captured volumes by the camera appearing sharp in the image. To account for misalignments of lens and sensor or poor lens quality, a distortion model is generally considered in addition to the pinhole. Distortion models are many in the literature but they are almost always a variation around a mix of tangential distortions and a rational polynomial model of radial distortions [5], often simplified as either a polynomial [6] or division [7] model. Indeed, the former involves more parameters than both latter that are easier to calibrate.

Less classical is the use of panoramic cameras such as those using a fisheye lens [8] or a catadioptric lens involving a curved mirror [9]. Many camera models have been proposed for describing the image formation with the latter lenses [10] because they capture a horizontal FoV of 180 degrees or more that leads the pinhole model to a singularity. The most often encountered projection models for fisheye cameras are those to which the manufacturers are trying to fit such as equidistant and equisolid models [8] but also those more general such as the angular polynomial model [11]. Catadioptric camera models are relying on the shape of the mirror used, such as the paraboloidal ad hoc model [12], or, for the set of camera-mirror pairs implementing a single viewpoint, the Unified Central projection Model (UCM) [13], [14]. When the mirror shape or the position of the camera with respect to the mirror is not accurate, the UCM is extended with the same polynomial distortion models than for the pinhole camera [15]. But the latter extension makes a two steps model so, in the same idea as the angular polynomial model designed for fisheye cameras, the Cartesian polynomial model has been expressed to account simultaneously for the geometry of the mirror and the distortions [16]. It can actually also be used to model some fisheye cameras, even though super-fisheye cameras (FoV beyond 200 degrees) are more accurately modeled with the double sphere projection model [17] or even generic models [18], though harder to calibrate than parametric models [19].

To summarize, the landscape of camera models is very large and rich but rarely the newly introduced models came with the way to formulate them from previous models and vice versa, except for the UCM that showed its *equivalence* to ad hoc models of catadioptric cameras [13], [14]. The only work we found tackling the camera model conversion issue [20] lies in the remote sensing field and expresses the photogrammetric model [21] used for computing digital terrain models from the CAHVOR camera model [22] used for machine vision and vice versa. However, the robotics field considers other camera models for which the lack of conversion tools prevents to easily use a vision-based method, e.g. a visual SLAM for robot localization, that implements a camera model to use images captured with a camera calibrated with a different model, either at the factory or provided together with a dataset. The practical solution is to recalibrate the camera, but if the data used does not contain the calibration images, this is not possible. Moreover, it is sometimes difficult to obtain a calibration as accurate as the one done at the factory. In any case, recalibrating the camera takes time and requires expertise in setting the calibration targets in the camera FoV [23].

We contribute to avoid camera recalibration as follows:

- the formulation of several state-of-the-art projection models in a unified notation

E. Goichon, G. Caron and P. Vasseur are with Université de Picardie Jules Verne, MIS laboratory, 80000 Amiens, France {eva.goichon, guillaume.caron, pascal.vasseur}@u-picardie.fr

F. Kanehiro, E. Goichon and G. Caron are with CNRS-AIST JRL (Joint Robotics Laboratory), IRL, and the National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, 305-8560, Japan f-kanehiro@aist.go.jp

- the methodology to convert from a camera model to another, developed for three conversions
- the programs within an open-source library to perform the conversions easily

The three model conversions are applied for three cameras: a 185 degrees fisheye camera, a 180 degrees catadioptric camera and a conventional camera with distortions. The conversion accuracy is evaluated with respect to calibrations with the destination models. The third camera is used within visual SLAM to compare the accuracy of estimated trajectories with converted and calibrated models.

The rest of the paper is organized as follows. Section II related the camera models considered in the paper with a unified notation. Section III introduces the camera models conversion method developed to three conversions. After that, Section IV reports the conversion results obtained, their evaluation and the use and evaluation of one of them in visual SLAM, before conclusion (Sec. V).

## II. RELATED WORKS

This section recalls quickly the camera projection models considered in this paper with a unified writing: three wide FoV camera projection models (Sec. II-A, Sec. II-B, and II-D) and distortion models (Sec. II-C).

### A. Unified central camera model

Central omnidirectional cameras leverage the unified central camera projection model (UCM) [14] instead of the pinhole one classically used. Considering intrinsic parameters $\alpha_u \in \mathbb{R}^*$, $\alpha_v \in \mathbb{R}^*$ as the generalized focal length, $u_0 \in \mathbb{R}$, $v_0 \in \mathbb{R}$ as the principal point coordinates and $\xi \in \mathbb{R}$ a parameter associated to the lens shape (or mirror shape in case of catadioptric lens), the UCM projects 3D points $\mathbf{X} = [X, Y, Z]^\top \in \mathbb{R}^3$ to digital image points $\mathbf{u} = [u, v]^\top \in \mathbb{R}^2$ with three steps. First, $\mathbf{X}$ is projected as $\mathbf{X}_S = [X_S, Y_S, Z_S]^\top \in \mathbb{R}^3$ on a unit sphere centered at the camera origin such that:

$$X_S = X/\rho \quad \text{and} \quad Y_S = Y/\rho \quad \text{and} \quad Z_S = Z/\rho, \tag{1}$$

with $\rho = \sqrt{X^2 + Y^2 + Z^2}$. Second, $\mathbf{X}_S$ is projected as $\mathbf{x} = [x, y]^\top \in \mathbb{R}^2$ on the normalized image plane thanks to a second projection center distant of $\xi$ from the sphere center:

$$x = X_S/(Z_S + \xi) \quad \text{and} \quad y = Y_S/(Z_S + \xi). \tag{2}$$

Third, $\mathbf{x}$ is transformed to the digital image plane as $\mathbf{u}$:

$$u = \alpha_u x + u_0 \quad \text{and} \quad v = \alpha_v y + v_0. \tag{3}$$

As a recall, setting $\xi = 0$ in (2) allows one to retrieve the classical pinhole projection model.

### B. Equidistant fisheye model

This model assumes symmetric radial distortions in the image and implements a regular radial resolution of angle $\phi \in \mathbb{R}$, the elevation with respect to the camera optical axis $\mathbf{Z}_c \in \mathbb{R}^3$. Writing $\theta \in \mathbb{R}$ the azimuth angle, these angles can be expressed from unit spherical Cartesian coordinates (1):

$$\phi = \arccos(Z_S) \quad \text{and} \quad \theta = \arctan(Y_S/X_S). \tag{4}$$

The equidistant fisheye model [8] maps the azimuth and elevation angles to normalized image plane coordinates by:

$$x = \phi\cos(\theta) \quad \text{and} \quad y = \phi\sin(\theta). \tag{5}$$

With $f \in \mathbb{R}_+$ the focal length of the fisheye lens and $k \in \mathbb{R}_+$ the pixel pitch (square pixels assumed), the point $\mathbf{x}$ coordinates in the digital image are:

$$u = \frac{f}{k}x + u_0 \quad \text{and} \quad v = \frac{f}{k}y + v_0. \tag{6}$$

### C. Distortions model

Camera projection models are often extended with a model of distortions to account for residual distortions of lenses or misalignment of lenses and the image sensor. The so called Brown-Conrady's distortion model [6] considers a polynomial model of radial and tangential *distortions applied to a 2D point* $\mathbf{x}$, which coordinates are expressed in the normalized image plane, to obtain their distorted counterparts $\mathbf{x}_d = (x_d, y_d)^\top \in \mathbb{R}^2$. The radial distortions coefficient $d_B \in \mathbb{R}$ is a polynomial of $\rho_\mathbf{x} = \sqrt{x^2 + y^2}$ such as:

$$d_B = 1 + k_{B_1}\rho_\mathbf{x}^2 + k_{B_2}\rho_\mathbf{x}^4 + k_{B_3}\rho_\mathbf{x}^6, \tag{7}$$

leading to the expression of $\mathbf{x}_d$:

$$\begin{cases} x_d = d_B x + 2p_1 xy + p_2(\rho_\mathbf{x}^2 + 2x^2) \\ y_d = d_B y + p_1(\rho_\mathbf{x}^2 + 2y^2) + 2p_2 xy \end{cases}. \tag{8}$$

More recent distortion models extend (7) to a rational polynomial model [5] for the radial distortions:

$$d_R = \frac{1 + k_{R_1}\rho_\mathbf{x}^2 + k_{R_2}\rho_\mathbf{x}^4 + k_{R_3}\rho_\mathbf{x}^6}{1 + k_{R_4}\rho_\mathbf{x}^2 + k_{R_5}\rho_\mathbf{x}^4 + k_{R_6}\rho_\mathbf{x}^6}, \tag{9}$$

to use instead of $d_B$ in (8).

Finally, the distorted digital image coordinates $\mathbf{u}$ are computed by substituting $x$ and $y$ in (3) with $x_d$ and $y_d$ of (8).

### D. Cartesian Polynomial model

This model was designed for catadioptric and fisheye cameras of FoV of 180 degrees and more [16]. This model maps the scene to the digital image by considering a radial model of *distortions depending on the Cartesian expression of 3D lines of sight*. By considering a single $\alpha = \alpha_u = \alpha_v$, it expresses the line of sight from $\mathbf{u}' = \mathbf{u} - [u_0, v_0]^\top$ to the 3D point $\mathbf{X}$ of norm $\rho$ by:

$$\frac{\rho}{\alpha}[u', v', r(\rho_{\mathbf{u}'})] = \mathbf{X}, \tag{10}$$

where $\rho_{\mathbf{u}'} = ||\mathbf{u}'||$ and the radial distortions function:

$$r(\rho_{\mathbf{u}'}) = a_0 + a_1\rho_{\mathbf{u}'} + a_2\rho_{\mathbf{u}'}^2 + \ldots + a_n\rho_{\mathbf{u}'}^n. \tag{11}$$

In practice, the degree $n$ is at most $n = 4$ and $a_1$ is set to 0, shrinking the radial distortions function to 4 parameters at most.

## III. CONVERTING CAMERA MODELS

This section develops the method to convert a camera projection model to another for two conversions of wide-angle camera projection models (Sec. III-A and III-B) and one conversion between distortion models (Sec. III-C).

### A. Equidistant to Unified Central Model

Focusing on coordinates expressed on the horizontal axis ($\theta = 0$), we can state that (3) and (6) should be equal, i.e. after substitutions with (2), resp. 5:

$$\alpha_u \frac{X_S}{Z_S + \xi} + u_0 = \frac{f}{k}\phi\cos(\theta) + u_0. \tag{12}$$

Above, $u_0$ is directly identified and the rest simplifies to:

$$\alpha_u \frac{X_S}{Z_S + \xi} = \frac{f}{k}\phi, \tag{13}$$

that is rewritten to show an equation linear in the unknowns $\alpha_u$ and $\xi$:

$$\alpha_u X_S = (Z_S + \xi)\frac{f}{k}\phi \tag{14}$$

$$\frac{X_S}{f/k\ \phi}\alpha_u = Z_S + \xi \tag{15}$$

$$\frac{X_S}{f/k\ \phi}\alpha_u - \xi = Z_S. \tag{16}$$

As $\theta = 0$, (4) leads to $\sin(\phi)$, resp. $\cos(\phi)$, substituting $X_S$, resp. $Z_S$:

$$\frac{\sin(\phi)}{f/k\ \phi}\alpha_u - \xi = \cos(\phi). \tag{17}$$

Then, a minimum of two instances $i \in \mathbb{N} \setminus \{1\}$ of (17) with $\phi_i \neq 0$ is necessary to solve for the two unknowns. In matrix form, we rewrite the above equation as:

$$\begin{bmatrix} \vdots & \vdots \\ \frac{\text{sinc}(\phi_i)}{f/k} & -1 \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} \alpha_u \\ \xi \end{bmatrix} = \begin{bmatrix} \vdots \\ \cos(\phi_i) \\ \vdots \end{bmatrix}, \tag{18}$$

solved with the matricial inverse, or pseudo-inverse if $i > 2$.

### B. Unified Central Model to Cartesian Polynomial

Following a similar methodology than in Sec. III-A, we focus on coordinates expressed on the horizontal axis ($Y = 0$). We can thus simplify the Cartesian Polynomial model (10) as:

$$\begin{cases} \frac{\rho}{\alpha}(u - u_0) = X \\ \frac{\rho}{\alpha} r(\rho_{\mathbf{u}'}) = Z \end{cases}, \tag{19}$$

which the ratio of the second over the first equation of the above system removes factor $\frac{\rho}{\alpha}$:

$$\frac{r(\rho_{\mathbf{u}'})}{u'} = \frac{Z}{X}. \tag{20}$$

Then, by substituting $u'$ in (20) with its expression from the Unified Central Model (2):

$$u' = \alpha_u \frac{X_S}{Z_S + \xi}, \tag{21}$$

and noting $\rho_{\mathbf{u}'} = |u'|$ lead to:

$$\frac{r(|u'|)}{u'} = \frac{Z}{X} = \frac{Z_S}{X_S}, \tag{22}$$

that we develop and re-organize to obtain:

$$\begin{aligned} a_0 + |u'|^2 a_2 + |u'|^3 a_3 + |u'|^4 a_4 &= \alpha_u \frac{X_S}{Z_S+\xi}\frac{Z_S}{X_S} \\ &= \alpha_u \frac{Z_S}{Z_S+\xi} \end{aligned} \tag{23}$$

Considering a second degree polynomial for more compactness, a minimum of two instances ($X \neq 0$) of the above equation is necessary to solve for the two unknowns. We rewrite the above equation for degree two with matrices as:

$$\begin{bmatrix} \vdots & \vdots \\ 1 & \left(\alpha_u \frac{X_{S_i}}{Z_{S_i}+\xi}\right)^2 \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} a_0 \\ a_2 \end{bmatrix} = \begin{bmatrix} \vdots \\ \alpha_u \frac{Z_{S_i}}{Z_{S_i}+\xi} \\ \vdots \end{bmatrix}, \tag{24}$$

that is solved with the matricial inverse or pseudo-inverse if more than two points $[X_{S_i}, Y_{S_i}, Z_{S_i}]^\top$ are considered.

### C. Rational polynomial distortion model to Brown-Conrady's

Using Brown-Conrady's distortion model (7), (8) and applying (3), we obtain the full expression of distorted coordinates $\mathbf{u}_{d_B} = (u_{d_B}, v_{d_B})^\top \in \mathbb{R}^2$ in the digital image plane:

$$\begin{cases} u_{d_B} = \alpha_u \left(d_B x + 2p_1 xy + p_2(\rho_{\mathbf{x}}^2 + 2x^2)\right) + u_0 \\ v_{d_B} = \alpha_v \left(d_B y + p_1(\rho_{\mathbf{x}}^2 + 2y^2) + 2p_2 xy\right) + v_0 \end{cases}. \tag{25}$$

The distorted coordinates $\mathbf{u}_{d_R} = (u_{d_R}, v_{d_R})^\top \in \mathbb{R}^2$ in the digital image plane are obtained with the rational polynomial model (9) similarly to (25) but with $d_R$ instead of $d_B$.
Then, with either coordinate, we solve for $u_{d_B} = u_{d_R}$ or $v_{d_B} = v_{d_R}$, leading to the direct identification of parameters $\alpha_u$, $\alpha_v$, $u_0$, $v_0$, $p_1$ and $p_2$, the same for both the Brown-Conrady and the rational polynomial model. Thus, the remaining equation is $d_B = d_R$, that is:

$$1 + k_{B_1}\rho_{\mathbf{x}}^2 + k_{B_2}\rho_{\mathbf{x}}^4 + k_{B_3}\rho_{\mathbf{x}}^6 = \frac{1 + k_{R_1}\rho_{\mathbf{x}}^2 + k_{R_2}\rho_{\mathbf{x}}^4 + k_{R_3}\rho_{\mathbf{x}}^6}{1 + k_{R_4}\rho_{\mathbf{x}}^2 + k_{R_5}\rho_{\mathbf{x}}^4 + k_{R_6}\rho_{\mathbf{x}}^6},$$

that we re-organise as a linear equation in the unknowns $k_{B_1}$, $k_{B_2}$, $k_{B_3}$:

$$\rho_{\mathbf{x}}^2 k_{B_1} + \rho_{\mathbf{x}}^4 k_{B_2} + \rho_{\mathbf{x}}^6 k_{B_3} = d_R - 1. \tag{26}$$

Following a similar solving methodology than in Section III-A, one can solve for the three parameters of (26) by a minimum of three non-distorted normalized image point instances $j \in \mathbb{N} \setminus \{1, 2\}$, each leading to a unique $\rho_{\mathbf{x}_j}$, by stacking instances of (26):

$$\begin{bmatrix} \vdots & \vdots & \vdots \\ \rho_{\mathbf{x}_j}^2 & \rho_{\mathbf{x}_j}^4 & \rho_{\mathbf{x}_j}^6 \\ \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} k_{B_1} \\ k_{B_2} \\ k_{B_3} \end{bmatrix} = \begin{bmatrix} \vdots \\ d_{R_j} - 1 \\ \vdots \end{bmatrix}, \tag{27}$$

and using the matricial inverse, or pseudo-inverse if $j > 3$.

## IV. RESULTS

This section reports first the results of camera projection model conversions (Sec. IV-A) obtained using the methods developed in Section III and then, an application to visual SLAM with a color-depth camera, which distortion parameters are the result of a conversion, onboard a robot with ground truth captured with motion capture (Sec. IV-B).

### A. Conversions

In this section, we evaluate first for a fisheye camera the conversion from the equidistant projection model to the UCM (Sec. IV-A.1), then for a catadioptric camera the conversion from the UCM to the Cartesian polynomial model (Sec. IV-A.2), and finally for the conversion of the distortion model of a conventional FoV color-depth camera from the rational polynomial to the polynomial distortion model (Sec. IV-A.3). All these camera model conversion methods are implemented within the libPeR library open-sourced at: https://github.com/PerceptionRobotique/libPeR_base.

To evaluate the model conversion results, we compare them with the calibration results considering the destination camera model. To calibrate cameras with either the UCM or the pinhole model with polynomial distortions, we use the latest version of [24], named *MIXEDVISION*, which source code is available at: https://github.com/PerceptionRobotique/MIXEDVISION.

On the other hand, to calibrate a camera with the Cartesian polynomial model, we used the *OCamCalib* calibration toolbox[1] for Matlab that shipped with the seminal Cartesian polynomial model calibration method [16].

*1) Equidistant to UCM:* We used a Prophesee Gen3.1 event camera of $640 \times 480$ pixels with $k = 15$ $\mu$m pixel pitch. The lens is a Fujinon FE185C086HA-1 equidistant fisheye lens of $f_f = 2.7$ mm focal length. Its FoV is 185 degrees. In our experiments, we solve (24) by arbitrarily computing $N = 2 \times \lfloor 185/2 \rfloor = 184$ angles $\phi_i$, hence leading (24) to be solved with the pseudo-inverse. Hence, the approximate mapping to the UCM, leads to $\alpha_u = \alpha_v \approx 499.4629$ and $\xi = 1.7841$. $u_0$ and $v_0$ are the same for both models (if the optical axis is perfectly perpendicular at the exact center of the image, we would have $u_0 = 320$, $v_0 = 240$ pixels). On the 184 $\phi_i$, the average residual error in the digital image plane is 0.06 pixels, with most of the errors on the outer ring of the 185 degrees FoV (Fig. 1).

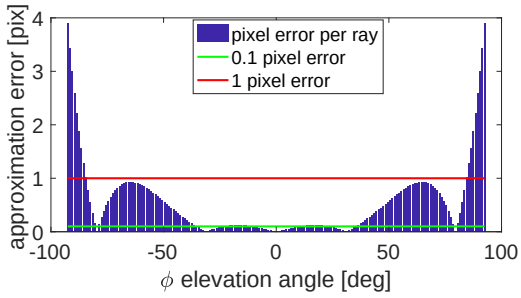
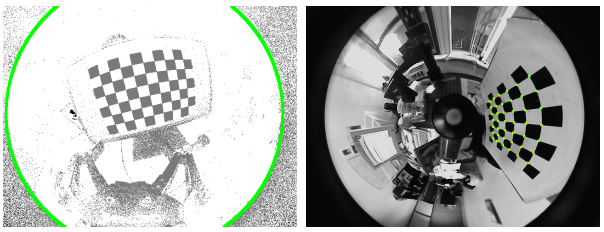



Fig. 1: Equidistant fisheye to UCM conversion: in digital image plane residual errors.

Of course, in practice the optical axis of the lens is rarely perfectly aligned with the exact center of the image. For instance, Fig. 2a highlights the outer ring of the fisheye event camera FoV in the digital image which center $u_c = 308$, $v_c = 234$ pixels is obviously shifted by several pixels from the exact image center. Thus, assuming the perpendicularity of the lens optical axis to the image plane, the principal point $u_0, v_0$ would tend to $u_c, v_c$. This is confirmed by calibrating the fisheye event camera with the UCM using *MIXEDVISION* that lead to the optimal intrinsic parameters $\hat{\alpha}_u = 489.0459$, $\hat{\alpha}_v = 490.3241$, $\hat{u}_0 = 307.1954$, $\hat{v}_0 = 228.9119$ and $\hat{\xi} = 1.7665$ (the hat denotes the optimum). $\hat{u}_0$ and $\hat{v}_0$ are way closer to the fisheye FoV circle than the exact image center as expected. However, the other parameters $\hat{\alpha}_u$, $\hat{\alpha}_v$ and $\hat{\xi}$ are close to the ones computed by converting the theoretical equidistant model to the UCM, which confirms the interest of the approach.

To get rid of a possible bias related to a single calibration example, we ran 8 distinct calibration procedure by detaching and re-attaching the fisheye lens to the event camera, capturing each time new calibration targets at different poses and running the optimization of the intrinsic parameters. Furthermore, 4 calibrations among the 8 done were conducted after dismounting the image sensor itself from the camera box to highlight the principal point can change significantly depending on the camera assembling. As a statistical result, the mean $\mu$ and standard deviations $\sigma$ of optimal intrinsic parameters $\overline{\hat{\alpha}_u}$, $\overline{\hat{\alpha}_v}$, $\overline{\hat{u}_0}$, $\overline{\hat{v}_0}$ and $\overline{\hat{\xi}}$ are reported in Table I. Interestingly, $\hat{u}_0$ is very stable (less than 1% of deviation). $\hat{v}_0$ is also very stable before (1.4% deviation) or after (0.2% deviation) dismounting/re-assembling the image sensor even if the computed $\overline{\hat{v}_0}$ before and after dismounting/re-assembling the image sensor highlights the sensor is not exactly at the same location in the camera box. Another interesting thing is that always $\hat{\alpha}_u \approx \hat{\alpha}_v$ with, on average, less than 3% different from the converted ones. Similarly, $\overline{\hat{\xi}}$ (computed for the 8 calibrations) deviates of less than 3% from the converted $\xi$. Ignoring the global $\overline{\hat{v}_0}$, one may note $\overline{\hat{\alpha}_u}$, $\overline{\hat{\alpha}_v}$ and $\overline{\hat{\xi}}$ show greater standard deviations than other parameters. This is not surprising since these parameters are known to be coupled leading to various (to some extent) possible correct solutions



(a) (b)

Fig. 2: Examples of calibration images used for evaluating projection model conversions. (a) Image of a fisheye event camera accumulating events while a monitor is blinking, displaying a calibration chessboard. The green circle shows an approximation of the fisheye FoV in the image plane. (b) Image of a catadioptric camera from the *OCamCalib* [16] dataset, superimposed with detected (yellow disks) and reprojected (green crosses) calibration target corners after calibration with *MIXEDVISION* [24].

[1] https://rpg.ifi.uzh.ch/software_datasets.html (accessed in Sept. 2023)

TABLE I: Mean $\mu$, standard deviation $\sigma$ and their ratio $\sigma/\mu$ (given as percentage) of intrinsic parameters calibrated for two sets of four calibration procedures: 1-4 before and 5-8 after disassembling/reassembling the image sensor, each one with dismounting/mounting the fisheye lens (*all* gathers 1-8).

| # | | $\overline{\hat{\alpha}_u}$ | $\overline{\hat{\alpha}_v}$ | $\overline{\hat{u}_0}$ | $\overline{\hat{v}_0}$ | $\overline{\hat{\xi}}$ |
|---|---|---|---|---|---|---|
| all | $\mu$ | 487.67 | 487.88 | 307.01 | 220.78 | 1.73 |
| | $\sigma$ | 18.81 | 19.04 | 1.11 | 13.39 | 0.08 |
| | $\frac{\sigma}{\mu}$ | 3.9% | 3.9% | 0.4% | 6.1% | 4.4% |
| 1-4 | $\mu$ | 481.04 | 481.21 | 306.91 | 233.15 | 1.71 |
| | $\sigma$ | 8.62 | 8.88 | 1.51 | 3.20 | 0.05 |
| | $\frac{\sigma}{\mu}$ | 1.8% | 1.8% | 0.5% | 1.4% | 2.9% |
| 5-8 | $\mu$ | 494.30 | 494.55 | 307.11 | 208.41 | 1.75 |
| | $\sigma$ | 25.18 | 25.48 | 0.77 | 0.33 | 0.10 |
| | $\frac{\sigma}{\mu}$ | 5.0% | 5.0% | 0.3% | 0.2% | 5.5% |

to the calibration problem [17].

*2) UCM to Cartesian polynomial:* For this camera model conversion evaluation, we use the images set provided with *OCamCalib*. There are 10 images provided, captured with a catadioptric lens by moving a checkerboard around to cover the whole FoV of approximately 180 degrees (Fig. 2b). Using *OCamCalib* to calibrate the camera with a Cartesian polynomial model of order 2, refining several times the locations of checkerboard corners, the principal point and the whole set of intrinsic and extrinsic parameters, we obtain the results, considered as reference (hence the star): $a_0^*$ = 131.0074, $a_2^*$ = -0.0018, $u_0^*$ = 516.4379 and $v_0^*$ = 383.0140, for an average reprojection error of 0.39 pixels.

On the other hand, we used the same images to calibrate the camera with the UCM using *MIXEDVISION*. This time the average reprojection error is 0.31 pixels and optimal intrinsic parameters are: $\hat{\alpha}_u$ = 259.889, $\hat{\alpha}_v$ = 259.335, $\hat{u}_0$ = 514.168, $\hat{v}_0$ = 382.797, $\hat{\xi}$ = 0.975. The reprojection error reached with the UCM compared to the Cartesian polynomial is slightly lower but it is not surprising since *MIXEDVISION* optimizes all the UCM parameters simultaneously whereas *OCamCalib* optimizes separately for $a_0$ and $a_2$ on the one hand and for $u_0$ and $v_0$ on the other hand.

Then, we apply our conversion method from the UCM to the Cartesian polynomial model and obtain: $\hat{a}_0$ = 131.4600, $\hat{a}_2$ = -0.0018 ($\hat{u}_0$ and $\hat{v}_0$ do not change). Clearly $\hat{a}_2 = a_2^*$ and $\hat{a}_0 \approx a_0^*$ (less than half a percent of difference) which validate our approach that easily allows to use cameras calibrated with the UCM with vision software considering the Cartesian polynomial model.

*3) Rational polynomial model to Brown-Conrady's:* We used an Azure Kinect DK color-depth (RGBD) camera which the color camera image capture resolution is set to 2048×1536 pixels for a FoV of 90°×74.3°. It has four depth modes: a narrow FoV of 75°×65° (NFOV) and a wide FoV of 120°×120° (WFOV), each with two possible resolutions, either 640×576 pixels (*Unbinned*) or 320×288 pixels (*Binned*) for NFOV and either 1024×1024 pixels (*Unbinned*) or 512×512 (*Binned*) pixels for WFOV. Whatever depth mode, the depth image is always transformed to the color image space such that a pixel has matching color and depth data (if the depth could be measured). So, in this paper, we consider only the calibration of the color camera.

TABLE II: Intrinsic parameters of an Azure Kinect RGBD camera implementing the pinhole and distortion models: rational polynomial distortions model for the factory parameters; polynomial distortion model for the converted and calibrated parameters. In Section IV-B, parameters with a † are set to 0 for the *factory 4 parameters tests* and those with a ‡ are set to 0 for the *factory 6 parameters tests*.

| **Parameters** | **Factory** | **Calibrated** | **Converted** |
|---|---|---|---|
| $\alpha_u$ | 967.548 | 975.605 | 967.548 |
| $\alpha_v$ | 967.409 | 975.408 | 967.409 |
| $u_0$ | 1025.603 | 1031.959 | 1025.603 |
| $v_0$ | 777.720 | 776.158 | 777.720 |
| $k_1$ | $0.399^{\dagger,\ddagger}$ | 0.117 | 0.112 |
| $k_2$ | $-2.589^{\dagger,\ddagger}$ | -0.113 | -0.110 |
| $p_1$ | $-1.526e\text{-}6^{\dagger}$ | 3.343e-4 | -1.526e-06 |
| $p_2$ | $-3.088e\text{-}4^{\dagger}$ | 1.791e-3 | -3.088e-4 |
| $k_3$ | $1.528^{\dagger,\ddagger}$ | 5.000e-2 | 5.145e-2 |
| $k_4$ | $0.276^{\dagger,\ddagger}$ | - | - |
| $k_5$ | $-2.402^{\dagger,\ddagger}$ | - | - |
| $k_6$ | $1.448^{\dagger,\ddagger}$ | - | - |

The Azure Kinect DK is comes from the factory with the parameters of a pinhole camera extended with the rational polynomial distortion model (Sec. II-C). We thus consider its conversion to the Brown-Conrady model and its calibration with the same model for comparison. We repeated the calibration procedure three times and kept the parameters with lowest residual error for fair comparison. Every intrinsic parameter are summarized in Table II.

Recall that the converted $\alpha_u$, $\alpha_v$, $u_0$, $v_0$ and $p_1$ and $p_2$ parameters are identical to the factory values (Sec. III-C). The calibrated $\alpha_u$, $\alpha_v$, $u_0$, $v_0$ are also very close to those obtained at the factory, the former deviating from the latter by $0.2\%$ to $0.8\%$. Parameters $k_1$ to $k_3$ obtained at the factory are not comparable to those converted or calibrated because the distortion models are different. But, on the one hand, the converted and the calibrated $k_1$, $k_2$, $k_3$ are very close to each other, though this time deviating by $3.3\%$ on average. On the other hand, the calibrated tangential distortion parameters $p_1$ and $p_2$ are however very different. Since all these parameters are linked, the best way to evaluate their relevance is to use them for a vision-based application, that we chose to be visual SLAM due to its broad interest in robotics.

### *B. Application to RGB and RGBD visual SLAM*

To be able to compare the impact of the different sets of RGBD camera distorsion parameters (Sec. IV-A.3), we compare the trajectories resulting from visual SLAM using the open-source software[2] StellaVSLAM [25] applied to our new CD-MaJ dataset (Color-Depth vision dataset of MIS and JRL laboratories), publicly shared at: https://extra.u-picardie.fr/nextcloud/index.php/s/yLG72QH46tsSiea (15 GB).

StellaVSLAM, close to ORB-SLAM2 [26], can use monocular, stereo, RGBD and equirectangular cameras. StellaVSLAM considers RGB and RGBD cameras follow the pinhole

[2]https://github.com/stella-cv/stella_vslam

TABLE III: Average values of relative and absolute errors in translation (*Trans* in mm) and rotation (*Rot* in degree) of trajectories. In bold, the smallest absolute (*A* stands for APE) and relative (*R* stands for RPE) rotation and translation error values for each trajectory. *F-4* stands for *factory-4p*, *F-6* for *factory-6p*, *Con* for *converted-9p* and *Cal* for *calibrated-9p*.

| | | NFOV Binned 15Hz | | | | NFOV Binned 30Hz | | | | NFOV Unbinned 15Hz | | | | NFOV Unbinned 30Hz | | | | WFOV Binned 15Hz | | | | WFOV Binned 30Hz | | | | WFOV Unbinned 15Hz | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Trans | | Rot | | Trans | | Rot | | Trans | | Rot | | Trans | | Rot | | Trans | | Rot | | Trans | | Rot | | Trans | | Rot | |
| | | A | R | A | R | A | R | A | R | A | R | A | R | A | R | A | R | A | R | A | R | A | R | A | R | A | R | A | R |
| RGB align and scale | F_4 | 168 | 17 | 9.3 | 2.8 | 100 | 10 | 9.2 | 2.1 | 132 | 8 | 8.9 | 2.4 | 100 | 13 | 9.0 | 3.0 | 97 | 13 | 8.6 | 2.8 | 84 | 15 | 9.0 | 3.0 | 85 | 26 | 8.1 | 3.1 |
| | F_6 | 112 | 13 | 8.7 | **2.8** | 111 | 13 | 9.1 | 2.1 | 108 | 7 | 7.4 | **2.2** | 155 | 13 | 9.1 | 3.0 | 69 | 16 | 9.0 | **2.6** | 67 | 16 | 9.5 | **2.7** | 70 | 25 | 7.8 | **2.3** |
| | Con | **22** | **8** | 7.9 | 3.1 | **21** | 7 | 6.7 | 2.2 | **21** | 7 | **6.9** | 2.6 | 24 | **9** | 6.8 | **2.7** | **20** | **8** | 6.6 | 2.9 | **22** | **8** | 6.8 | 2.8 | 28 | **11** | 6.3 | 3.0 |
| | Cal | 24 | 9 | **7.7** | 3.5 | 28 | **6** | **6.4** | **2.0** | 31 | **6** | 7.3 | **2.2** | **22** | **9** | **6.5** | 3.1 | 23 | 9 | **5.8** | 2.9 | **22** | 9 | **6.3** | 2.8 | **20** | 12 | **5.4** | 3.2 |
| RGBD align and scale | F_4 | 89 | 10 | 10.8 | 3.1 | 60 | 8 | 9.7 | **2.2** | 106 | 8 | 7.0 | 2.8 | 81 | 10 | 9.1 | 3.1 | 68 | **8** | 9.1 | 2.7 | 114 | 10 | 7.8 | 3.0 | 106 | 14 | 9.1 | 3.7 |
| | F_6 | 93 | 10 | 10.2 | 3.0 | 68 | 8 | 9.8 | 2.3 | 154 | 10 | 8.3 | **2.4** | 100 | 11 | 9.5 | **2.9** | 76 | **8** | 9.3 | **2.6** | 80 | 10 | 8.9 | **2.8** | 70 | 16 | 9.1 | **3.1** |
| | Con | 23 | 8 | 7.7 | 1.9 | **19** | **7** | 7.2 | 2.3 | **19** | **7** | 7.3 | 2.8 | **21** | **8** | 7.0 | 3.3 | **19** | **8** | 6.9 | 3.2 | **22** | **8** | 6.5 | **2.8** | **21** | **4** | **4.8** | 3.5 |
| | Cal | **19** | **7** | **7.1** | **1.8** | 22 | **7** | **6.8** | 2.4 | **19** | **7** | **6.4** | **2.4** | 22 | **8** | **6.9** | 3.1 | 20 | **8** | **6.5** | 3.1 | **22** | 9 | **6.2** | 2.9 | 25 | 12 | 5.5 | **3.1** |
| RGBD align | F_4 | 87 | 10 | 10.8 | **3.0** | 59 | 8 | 9.7 | **2.2** | 118 | 8 | 7.0 | 2.8 | 79 | 10 | 9.0 | 3.1 | 86 | **8** | 9.1 | 2.7 | 140 | 10 | 7.8 | 3.0 | 114 | 14 | 9.1 | 3.7 |
| | F_6 | 92 | 10 | 10.2 | **3.0** | 68 | 8 | 9.8 | 2.3 | 117 | 8 | **6.2** | 2.7 | 100 | 11 | 9.5 | **2.9** | 88 | **8** | 9.3 | **2.6** | 85 | 10 | 8.9 | **2.8** | 108 | 16 | 9.1 | 3.1 |
| | Con | **45** | **9** | 8.3 | 3.2 | **41** | **7** | 7.2 | 2.3 | **42** | **7** | 7.3 | 2.8 | **44** | **8** | 7.0 | 3.3 | **38** | **8** | 6.9 | 3.2 | **43** | **8** | 6.5 | **2.8** | **33** | 14 | 6.3 | 3.5 |
| | Cal | 66 | **9** | **7.6** | 3.1 | 54 | **7** | **6.8** | 2.4 | 59 | **7** | 6.4 | **2.4** | 55 | **8** | **6.9** | 3.1 | 50 | **8** | **6.5** | 3.1 | 67 | 9 | **6.2** | **2.8** | 60 | **12** | **5.5** | **3.1** |

and Brown-Conrady's distorsion models. Hence, conversion is needed for the Azure Kinect that comes calibrated from the factory with the rational polynomial distortions model.

Our dataset is created using a Pioneer 3 AT mobile robot equipped with an Azure Kinect camera using a mast (height to the ground: 122 cm). Approximately circular trajectories of roughly 12 m length are carried out by manual control in the measurement volume of an Optitrack *motion capture* system in order to obtain precise ground truth and to compare the localization results of StellaVSLAM. All necessary data is saved via ROS as a rosbag file. A trajectory is created for each depth mode, available at 15Hz and 30Hz each, except for the WFOV Unbinned mode that only runs at 15Hz. The dataset is therefore composed of 7 distinct trajectories.

StellaVSLAM is run with 4 intrinsic parameter sets: *factory-4p* (factory's $\alpha_u$, $\alpha_v$, $u_0$, $v_0$, the rest set to zero), *factory-6p* (*factory-4p*'s parameters and factory's $p_1$ and $p_2$, the rest set to zero), *calibrated-9p* (calibrated pinhole and polynomial distortion parameters) and *convert-9p* (converted pinhole and polynomial distortion parameters). Each of the latter four sets is used within StellaVSLAM with RGB only and with RGBD data. *factory_4p* and *factory_6p* are of course doomed to fail but they are reported as very naive use of the Azure Kinect factory parameters in StellaVSLAM for comparison purpose. Every estimated trajectory are aligned and scaled to the motion capture ground truth, though the estimations with the RGBD data are also considered non-scaled for the sake of generality. Example estimated trajectories with the four parameter sets considering the RGB data of the Azure Kinect in NFOV, Binned, 15Hz mode for a single real trajectory are shown in Figure 3. The trajectories obtained with *convert_9p* and *calibrated_9p* parameters are way closer to the ground truth than the other two.

The average absolute pose errors (APE) and the per image relative pose errors (RPE) in translation and rotation for all the trajectories are reported in Table III. On the one hand, the example of Figure 3 is confirmed since every estimation error with parameter sets *factory-4p* and *factory-6p* are always significantly greater than those with the converted and calibrated parameters (average APE: +80mm on RGB trajectories; +70mm on scaled RGBD trajectories). On the other hand, *converted-9p* and *calibrated-9p* always show very similar estimation errors: on average, considering the scaled trajectories, the difference in APE is approximately 3 mm in translation and 0.5 degree in rotation. Actually, *converted-9p* even leads 18 times over 21 to lower error estimates than *calibrated-9p*. By showing trajectories the closest to the ground truth, these results show the effectiveness of the proposed conversion from the rational polynomial distortions model to Brown-Conrady's model.

## V. CONCLUSION

This paper has formulated several state-of-the-art projection models in a unified notation. It also presents an efficient methodology and mathematical developments for converting camera models for three conversions involving three different camera types. Experimental results obtained with calibration datasets and visual SLAM confirm the efficiency and interest of these conversions, enabling to avoid camera recalibration when this is not possible or difficult.

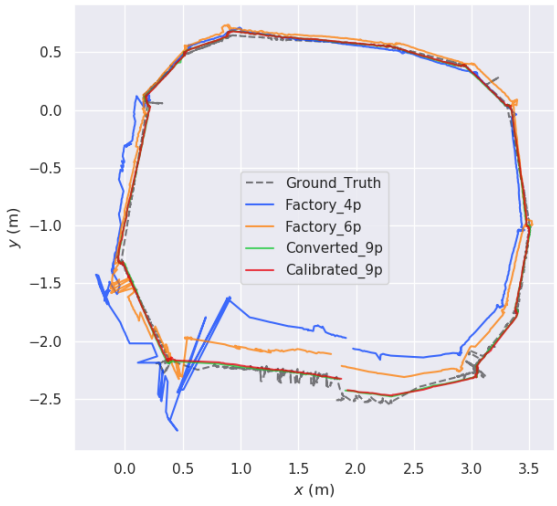



Fig. 3: Camera trajectories estimated with StellaVSLAM using the Azure Kinect (mode *NFOV-Binned-15Hz*) for the four sets of intrinsic parameters. The trajectories are aligned and scaled to the motion capture system. The trajectory is plotted using the EVO python package [27].

## REFERENCES


[1] P. Sturm, S. Ramalingam, J.-P. Tardif, S. Gasparini, and J. Barreto, "Camera Models and Fundamental Concepts Used in Geometric Computer Vision," Foundations and Trends in Computer Graphics and Vision, vol. 6, no. 1-2, pp. 1–183, Jan. 2011.

[2] G. Fahim, K. Amin, and S. Zarif, "Single-view 3d reconstruction: A survey of deep learning methods," Computers & Graphics, vol. 94, pp. 164–190, 2021.

[3] W. Förstner and B. P. Wrobel, Photogrammetric Computer Vision: Statistics, Geometry, Orientation and Reconstruction. Springer Publishing Company, Incorporated, 2016.

[4] D. Sharafutdinov, M. Griguletskii, P. Kopanev, M. Kurenkov, G. Ferrer, A. Burkov, A. Gonnochenko, and D. Tsetserukou, "Comparison of modern open-source visual slam approaches," J Intell Robot Syst, vol. 107, p. 43, 2023.

[5] V. Larsson, T. Sattler, Z. Kukelova, and M. Pollefeys, "Revisiting radial distortion absolute pose," in Proc. of IEEE/CVF Int. Conf. on Computer Vision (ICCV), Oct. 2019.

[6] D. Brown, "Decentering distortion of lenses," Photogrammetric Engineering, vol. 32, pp. 444–462, 1966.

[7] A. Fitzgibbon, "Simultaneous linear estimation of multiple view geometry and lens distortion," in Proc. of IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, vol. 1, 2001, pp. I–I.

[8] K. Miyamoto, "Fish eye lens," Journal of the Optical Society of America, vol. 54, no. 8, pp. 1060–1061, 1964.

[9] S. Gao, K. Yang, H. Shi, K. Wang, and J. Bai, "Review on panoramic imaging and its applications in scene understanding," IEEE Trans. on Instrumentation and Measurement, vol. 71, pp. 1–34, 2022.

[10] G. Caron, "Models and Calibration Methods," in Omnidirectional Vision: From Theory to Applications. ISTE, Dec. 2023.

[11] J. Kannala and S. S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 8, pp. 1335–1340, 2006.

[12] S. Baker and S. K. Nayar, "A theory of single-viewpoint catadioptric image formation," Int. Journal on Computer Vision, vol. 35, no. 2, pp. 175–196, 1999.

[13] C. Geyer and K. Daniilidis, "A unifying theory for central panoramic systems and practical applications," in Proc. of European Conf. on Computer Vision, Dublin, Irelande, Jul. 2000.

[14] J. Barreto, F. Martin, and R. Horaud, "Visual servoing/tracking using central catadioptric images," in Experimental Robotics VIII, 2003, pp. 245–254.

[15] C. Mei and P. Rives, "Single view point omnidirectional camera calibration from planar grids," in Proc. of IEEE Int. Conf. on Robotics and Automation, Rome, Italy, Apr. 2007, pp. 3945–3950.

[16] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easily calibrating omnidirectional cameras." in Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Pekin, China, Oct. 2006, pp. 5695–5701.

[17] V. Usenko, N. Demmel, and D. Cremers, "The double sphere camera model," in Proc. of Int. Conf. on 3D Vision, Verone, Italy, Sep. 2018.

[18] P.-A. Brousseau and S. Roy, "Calibration of axial fisheye cameras through generic virtual central models," in Proc. of IEEE/CVF Int. Conf. on Computer Vision, Seoul, South Korea, Oct. 2019.

[19] S. Ramalingam and P. Sturm, "A Unifying Model for Camera Calibration," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 39, no. 7, pp. 1309–1319, 2017.

[20] K. Di and R. Li, "Cahvor camera model and its photogrammetric conversion for planetary applications," Journal of Geophysical Research: Planets, vol. 109, no. E4, 2004.

[21] P. Wolf, Elements of Photogrammetry: With Air Photo Interpretation and Remote Sensing, ser. Civil Engineering Series. McGraw-Hill, 1983.

[22] D. B. Gennery, Least-Squares Camera Calibration Including Lens Distortion and Automatic Editing of Calibration Points. Springer Berlin Heidelberg, 2001, pp. 123–136.

[23] S. Peng and P. Sturm, "Calibration wizard: A guidance system for camera calibration based on modelling geometric and corner uncertainty," in Proc. of IEEE Int. Conf. on Computer Vision, 2019.

[24] G. Caron and D. Eynard, "Multiple camera types simultaneous stereo calibration," in Proc. of IEEE Int. Conf. on Robotics and Automation, Shanghai, China, May 2011, pp. 2933–2938.

[25] S. Sumikura, M. Shibuya, and K. Sakurada, "OpenVSLAM: A Versatile Visual SLAM Framework," in Proc. of ACM Int. Conf. on Multimedia, Nice, France, 2019, pp. 2292–2295.

[26] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras," IEEE Trans. on Robotics, vol. 33, no. 5, pp. 1255–1262, 2017.

[27] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.