An Enhanced Unified Camera Model

Bogdan Khomutenko, Gaëtan Garcia, Philippe Martinet

An Enhanced Unified Camera Model

Bogdan Khomutenko[1], Gaëtan Garcia[2], and Philippe Martinet[1]

***Abstract*— This paper describes a novel projection model based on the so-called unified projection model. The new model applies to catadioptric systems and wide-angle fish-eye cameras, it does not require additional mapping to model distortions, and it takes just two projection parameters more than a simple pinhole model to represent radial distortion (one parameter more than the unified model). Here we provide a study of different mathematical aspects of the model, its application limits, and explicit closed-form inversion. The latter allows to apply all the notions of epipolar geometry with no difficulties. Also we introduce a concept of projection surface, which is a useful notion to study and compare different projection models with radial distortion. Using developed software, several different lenses were calibrated using the proposed model, and in all cases sub-pixel precision was achieved.**

I. INTRODUCTION

Most computer vision problems in robotic applications involve camera modeling and calibration. Also fisheye cameras are of great interest in the field because they allow to equip the robot with 360° stereo vision using few cameras.

But wide-angle cameras do not obey a simple pinhole projection model. Multiple models have been proposed to approximate the projection process in such cameras. The authors of [1] propose to compute the distance from the projection center to the projected point using a polynomial function of the angle between the optical axis and the ray direction (so-called *capture ray*-based model). The so-called unified model is of particular interest to us because our own model is based on it. [2] introduces the unified model and shows that it describes all central catadioptric systems. In [3] the model was augmented with an additional distortion mapping and a calibration technique was proposed. Equivalence between the unified model and the captured ray-based model as well as the pinhole model was shown in [4]. The model was successfully applied to model fisheye cameras as well.

In [5] the projection is modeled using an intermediate surface. The surface is defined using a polynomial function of image points. The notion is somewhat similar to one that we propose in this paper, but it is the other way around: the projection surface is defined using an equation in 3D space and allows to compute projections of spatial points.

Important properties of a distortion model are simplicity and flexibility. That is, the model should have few parameters but still allow us to model different types of cameras. Also it is important for a model to have a simple mathematical form, which allows to study analytically the properties of the model.

Tangential distortion is not considered in this work. Though it can easily be added, we argue that without it the model is precise enough and a closed-form inversion of the projection function can be calculated.

In section II, we describe the unified model, from which the proposed one is derived. In section III, the proposed model is presented and its properties are studied by means of projection surfaces; also the inverse of the model in closed form is calculated. In section IV, the calibration results are presented and the efficiency of the model in case of different cameras is shown.

II. RELATED WORK

The simplest camera model is represented by this projection relation:

$$\boldsymbol{p} = \frac{1}{z} K \boldsymbol{X} \tag{1}$$

Here $\boldsymbol{p} = (u\ v)^T$ is an image point, $\boldsymbol{X} = (x\ y\ z)^T$ is a spatial point to be projected. K is a projection matrix:

$$K = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \end{pmatrix} \tag{2}$$

The distortion model is usually added in the following manner. First we project all the points onto the projection plane:

$$\boldsymbol{X}_n = (x_n\ y_n\ 1)^T = \frac{\boldsymbol{X}}{z} \tag{3}$$

Then we apply radial distortions:

$$\boldsymbol{X}_d = \boldsymbol{D}(\boldsymbol{X}_n) = \begin{pmatrix} D(r)x_n \\ D(r)y_n \\ 1 \end{pmatrix} \tag{4}$$

Here $r = \sqrt{x_n^2 + y_n^2}$; $\boldsymbol{D} : \mathbb{P}^2 \to \mathbb{P}^2$ is a radial distortion mapping ($\mathbb{P}^n$ is a n-dimensional projective space on $\mathbb{R}$). It is defined by $D : \mathbb{R}_{+0} \to \mathbb{R}_+$, that represents the deflection of

[1]B. KHOMUTENKO and P. MARTINET are with Institut de Recherche en Communications et Cybernétique de Nantes (IRCCYN), Ecole Centrale de Nantes (ECN), Nantes, France bogdan.khomutenko@irccyn.ec-nantes.fr philippe.martinet@irccyn.ec-nantes.fr

[2]G. GARCIA is with Ecole Centrale de Nantes (ECN), Nantes, France gaetan.garcia@ec-nantes.fr

a ray from its pinhole trajectory; $D(0) = 1$. The last thing is to apply the projection matrix K.

Theoretically we can model any projection mapping for $\boldsymbol{X}$ with $z > 0$ using these relations. But there are two problems. The first is how to choose the distortion function D. In the OpenCV the following form is used:

$$D(r) = \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} \tag{5}$$

Obviously these models approximate the distortions just in a limited range of r.

This fact is also related to the second problem, which arises when $z \to 0$. In this case the model is still defined but farther we go from the center of projection, the less precise the result, due to the distortion model limitations and numerical problems.

A. Unified Camera Model

This model aims to model correctly projection of points with zero and even negative z. It is needed to model fisheye lenses with more than 180° angle of view. This is done by changing the normalization equation from (3) to the following [2]:

$$\boldsymbol{m} = \begin{pmatrix} x_m & y_m & 1 \end{pmatrix}^T = \begin{pmatrix} \frac{x}{z + \xi\rho} & \frac{y}{z + \xi\rho} & 1 \end{pmatrix}^T \tag{6}$$

where $\rho = \sqrt{x^2 + y^2 + z^2}$; ξ is a projection parameter. When $\xi = 0$ we are back to the pinhole model. But the larger we take ξ, the wider the allowed angle between the optical axis and the point to be projected. So, for narrow-angle low-distortion cameras we expect this parameter to be about 0, while for fisheye cameras it should be large (in the order of 1). We shall notice that (6) generally does not map straight lines to straight lines. So, it introduces some distortion, which is in fact similar to that of a real fisheye camera.

Unfortunately these distortions are not flexible enough to model a real camera. That is why another mapping is introduced in [3]:

$$\begin{aligned} \boldsymbol{x}_d &= \boldsymbol{D}(\boldsymbol{m}) + \boldsymbol{\delta} \\ D(r) &= 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 \\ \boldsymbol{\delta}(\boldsymbol{m}) &= \begin{pmatrix} 2k_4 x_m y_m + k_5(r^2 + 2x_m^2) \\ 2k_5 x_m y_m + k_4(r^2 + 2y_m^2) \\ 0 \end{pmatrix} \end{aligned} \tag{7}$$

Where $r = \sqrt{x_m^2 + y_m^2}$; $\boldsymbol{\delta}$ models tangential distortions due to a misalignment of the retina normal and the lens optical axis in the camera; $\boldsymbol{D}$ is defined as in (4), D represents radial distortions.

Finally we apply the projection matrix K:

$$\boldsymbol{p} = K\boldsymbol{X}_d \tag{8}$$

Overall we have 10 projection parameters: ξ, $k_{1..5}$, f_u, f_v, u_0, v_0.

B. Distortion Model Issues

Even though the model provides high precision, it has some drawbacks. We can rewrite (6) as $\boldsymbol{m} = \phi(r)\boldsymbol{X}_n$, where $\boldsymbol{X}_n = \boldsymbol{X}/z$, and $r = \sqrt{x_n^2 + y_n^2}$. Here ϕ represents the nonlinear part of the projection:

$$\phi(r) = \frac{1}{1 + \xi\sqrt{1 + r^2}} \tag{9}$$

We see that the function $\phi(r)$ is even, so, its Taylor expansion contains just even degrees of r. Hence:

$$\phi(r) = \phi(0) + \frac{\phi''(0)}{2} r^2 + o(r^2) \tag{10}$$

So, the argument here is that in the neighborhood of the projection center the distortion caused by the nonlinear projection model is well-approximated by a second order polynomial with $\phi^* = \phi(0) + \frac{\phi''(0)}{2} r^2$. If after we apply another distortion model that contains a second-order term, we still get a distortion that is well-approximated with a second-order polynomial. Hence, the second order term in the distortion model (7) is redundant and it does not improve the model precision.

III. PROPOSED MODEL

Here are the proposed projection relations:

$$\begin{aligned} \boldsymbol{m} &= \begin{pmatrix} \frac{x}{\alpha\rho + (1-\alpha)z} \\ \frac{y}{\alpha\rho + (1-\alpha)z} \\ 1 \end{pmatrix} \quad \rho = \sqrt{\beta(x^2 + y^2) + z^2} \\ \boldsymbol{p} &= K\boldsymbol{m} \end{aligned} \tag{11}$$

The two parameters are $\alpha \in [0, 1]$ and $\beta > 0$. They allow us to better approximate the properties of lenses with strong distortion. This model assumes that the denominator $\alpha\rho + (1 - \alpha)z > 0$. K is the same matrix as in (2).

A. Projection Surfaces

To analyze the model, let us introduce the notion of *projection surface*. For the sake of simplicity let us consider only projection relations with radial distortion. That is, every projected point can be written as $\boldsymbol{m} = (x/\eta(\boldsymbol{X}) \;\; y/\eta(\boldsymbol{X}) \;\; 1)^T$ where x and y are the components of the spatial point $\boldsymbol{X}$, and $\eta : \mathbb{R}^3 \to \mathbb{R}_+$ is a function of $\boldsymbol{X}$, moreover $\forall \boldsymbol{X}_1, \boldsymbol{X}_2 \in \mathbb{R}^3$:

$$\begin{cases} x_1^2 + y_1^2 = x_2^2 + y_2^2 \\ z_1 = z_2 \end{cases} \implies \eta(\boldsymbol{X}_1) = \eta(\boldsymbol{X}_2) \tag{12}$$

This means that η can be represented by a function:

$$\hat{\eta} : \mathbb{R}^2 \to \mathbb{R}_+ \tag{13}$$

so that:

$$\eta(\boldsymbol{X}) = \hat{\eta}(\sqrt{x^2+y^2}, z) = \hat{\eta}(r, z) \tag{14}$$

$$\hat{\eta}(r, z) = \hat{\eta}(-r, z), \quad \forall r \in \mathbb{R} \tag{15}$$

(15) allows us to operate on $\mathbb{R}^2$, rather than $\mathbb{R}_{+0} \times \mathbb{R}$. In this case $y = 0 \implies \eta(\boldsymbol{X}) = \hat{\eta}(x, z)$, or $\hat{\eta}$ represents η in xz plane.

Also we require:

$$\forall \lambda \in \mathbb{R}_+ \quad \eta(\lambda \boldsymbol{X}) = \lambda \eta(\boldsymbol{X}) \tag{16}$$

that is, η is a homogeneous function of degree 1. The same holds for $\hat{\eta}$ as long as both r and z are homogeneous functions of $\boldsymbol{X}$ of degree 1.

Let us define the projection surface P by the following equation:

$$\eta(\boldsymbol{X}) = 1 \tag{17}$$

Projection surface is a surface of revolution. It is generated by rotating the curve $\hat{\eta}(x, z) = 1$ about z-axis. Let us call this curve a *projection curve*. The geometric meaning of (17) is that all the points of the projection surface are projected orthogonally to the image plane. So, we can think of the projection process as scaling the point $\boldsymbol{X}$ by $\eta(\boldsymbol{X})$ and then projecting it orthogonally into $\boldsymbol{m} \in M$ — the normal plane (see Fig. 1):

$$\begin{gathered} \boldsymbol{X}_p = \frac{\boldsymbol{X}}{\eta(\boldsymbol{X})} \\ \boldsymbol{m} = (x_p \; y_p \; 1) \end{gathered} \tag{18}$$

Using (16) we can deduce:

$$\eta(\boldsymbol{X}_p) = \eta\left(\frac{\boldsymbol{X}}{\eta(\boldsymbol{X})}\right) = \frac{\eta(\boldsymbol{X})}{\eta(\boldsymbol{X})} = 1 \tag{19}$$

Hence, all the points $\boldsymbol{X}_p$ belong to the projection surface. In fact $\boldsymbol{X}_p$ is the intersection between the projection surface P and the $O\boldsymbol{X}$ ray (Fig. 1).

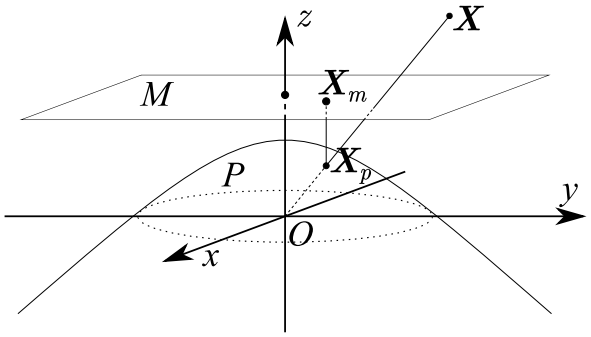

Fig. 1: Illustration of the projection surface notion. z is the optical axis; O is the center of projection; $\boldsymbol{X}_p$ is obtained by projecting $\boldsymbol{X}$ to P along $O\boldsymbol{X}$ ray. Then this point is transformed into $\boldsymbol{m}$ by projecting it orthogonally onto an intermediate projection plane M which is defined as $z = 1$

One convenience of the notion is that it is relatively easy to see for which spatial points the projection is defined. For example, (3) corresponds to $\eta(\boldsymbol{X}) = z$. So, the projection surface in this case is defined by $z = 1$. It is a plane. And all the points with $z \leq 0$ do not define rays that intersect the surface.

Let us apply the notion to the proposed model. In this case:

$$\eta(\boldsymbol{X}) = \alpha\sqrt{\beta(x^2+y^2)+z^2} + (1-\alpha)z \tag{20}$$

So, $\eta(\boldsymbol{X}) = 1$ leads to:

$$\alpha\sqrt{\beta(x^2+y^2)+z^2} + (1-\alpha)z = 1 \tag{21}$$

Let us replace $1-\alpha$ by γ and x^2+y^2 by r^2:

$$\alpha\sqrt{\beta r^2+z^2} = 1 - \gamma z \tag{22}$$

By squaring both sides we get:

$$\alpha^2\beta r^2 + \alpha^2 z^2 = 1 - 2\gamma z + \gamma^2 z^2 \tag{23}$$

We should remember that as we have squared both sides, we may get some solutions that do not satisfy (22). We may notice that $\gamma^2 - \alpha^2 = \gamma - \alpha$, hence:

$$\alpha^2\beta r^2 = 1 - 2\gamma z + (\gamma - \alpha)z^2 \tag{24}$$

$$\alpha^2\beta r^2 + (\alpha - \gamma)z^2 + 2\gamma z = 1 \tag{25}$$

This equation defines a second-order projection curve. $\alpha = 0.5$ leads to $\alpha = \gamma$ and

$$z = 1 - 0.25\beta r^2 \tag{26}$$

That is, the projection curve is a parabola. If $\alpha < 0.5$, (25) defines a hyperbola and if $\alpha > 0.5$, it is an ellipse (because the coefficient in front of z^2 is negative and positive respectively). We can see that $r = 0,\ z = 1$ always satisfies (22). So, the projection surface is a surface of revolution, which is defined by a conic projection curve that passes through $(0 \; 0 \; 1)^T$.

That is the difference between the proposed model and (6). The latter allows to get just one parabola as a projection curve when $\xi = 1$, while the former allows to scale this parabola along the x-axis.In the case, when $\xi \neq 1$ the same is true, but it is just less obvious. β allows us to adjust the projection surface, while α defines its shape.

B. Completeness of the Model

In fact we can show that (25) describes all the possible conics that pass through $(0 \; 1)^T$ and are symmetric with respect to z-axis (here we consider rz coordinate plane). To make a sketch of a proof let us consider a general conic equation:

$$Ar^2 + Brz + Cz^2 + Dr + Ez = 1 \tag{27}$$

To make it symmetric with respect to z-axis we have to have $B = 0$ and $D = 0$. Then, by substituting $z = 1$, $r = 0$ we get:

$$C + E = 1 \tag{28}$$

If we check (25), we see that all these conditions are satisfied. Indeed:

$$\begin{aligned} &B = 0 \\ &D = 0 \\ &C + E = \alpha - \gamma + 2\gamma = \alpha + \gamma = 1 \end{aligned} \tag{29}$$

But what if there is a projection curve which does not pass through $(0\ 1)^T$? Actually we can scale it so that it does (using parameters f_u and f_v — see Fig. 2). So, we can say that the proposed model is complete in the sense that it can fit any projection whose projection curve is a conic section.

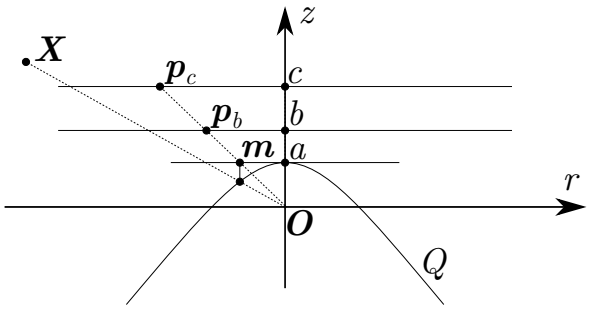

Fig. 2: Using projection curve Q, point $\boldsymbol{X}$ is projected to $\boldsymbol{m}$, which z-coordinate is a. If $a = 1$ and $f = c$ then the final projection is $\boldsymbol{p}_c = f\boldsymbol{m}$. But if $a < 1$, let us say that $b = 1$, then $\boldsymbol{p}_b = \frac{\boldsymbol{m}}{a}$ and the final projection $\boldsymbol{p}_c = f\boldsymbol{p}_b = \frac{f}{a}\boldsymbol{m} = f'\boldsymbol{m}$. So, we see that the projection defined by Q and f' is equivalent to one defined by f and Q scaled such that it pass through 1. The same argument is true when $a > 1$.

C. Inverse Model

Let us denote by $f : \mathbb{R}^3 \backslash 0 \to \mathbb{R}^2$ the mapping defined in (11). Here by inverse we mean an injective mapping $g : \mathbb{R}^2 \to \mathbb{R}^3$ such that:

$$f(g(\boldsymbol{m})) = \boldsymbol{m} \tag{30}$$

or, in other words, $f \circ g = I$. We know that the points of the projection surface are projected orthogonally. So, g may be defined as:

$$g : \begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \\ z(x,y) \end{pmatrix} \tag{31}$$

Where $z(x, y)$ is an explicit solution of (22) with $r = \sqrt{x^2 + y^2}$. To do that we can solve (25) and then choose the proper solution. It is a quadratic equation:

$$\begin{aligned} &Az^2 + Bz + C = 0 \\ &A = \alpha - \gamma \\ &B = 2\gamma \\ &C = \alpha^2\beta r^2 - 1 \end{aligned} \tag{32}$$

This is solved as follows:

$$\begin{aligned} &D = B^2 - 4AC = \\ &= 4\left(\gamma^2 - (\alpha - \gamma)(\alpha^2\beta r^2 - 1)\right) \\ &z = \frac{-B \pm \sqrt{D}}{2A} = \\ &= \frac{-\gamma \pm \sqrt{D/4}}{\alpha - \gamma} \end{aligned} \tag{33}$$

We can choose the solution by using the fact that $z(0, 0) = 1$. Notice that $r = 0 \implies D/4 = \alpha^2$. Hence, the solution must be defined by:

$$z = \frac{\sqrt{\gamma^2 - (\alpha - \gamma)(\alpha^2\beta r^2 - 1)} - \gamma}{\alpha - \gamma} \tag{34}$$

We see that (34) is not defined when $\alpha = 0.5$. It is so because (25) is no longer a quadratic equation.

We can avoid the singularity by multiplying both numerator and denominator of (34) by $\sqrt{\gamma^2 - (\alpha - \gamma)(\alpha^2\beta r^2 - 1)} + \gamma$:

$$\begin{aligned} z &= \frac{\gamma^2 - (\alpha - \gamma)(\alpha^2\beta r^2 - 1) - \gamma^2}{(\alpha - \gamma)\left(\sqrt{\gamma^2 - (\alpha - \gamma)(\alpha^2\beta r^2 - 1)} + \gamma\right)} = \\ &= \frac{1 - \alpha^2\beta r^2}{\sqrt{\gamma^2 - (\alpha - \gamma)(\alpha^2\beta r^2 - 1)} + \gamma} \end{aligned} \tag{35}$$

Also let us consider the expression under the square root in the denominator:

$$\begin{aligned} &\gamma^2 - (\alpha - \gamma)(\alpha^2\beta r^2 - 1) = \\ &= (\alpha - 1)^2 + 2\alpha - 1 - (\alpha - \gamma)\alpha^2\beta r^2 = \\ &= \alpha^2 - 2\alpha + 1 + 2\alpha - 1 - (\alpha - \gamma)\alpha^2\beta r^2 = \\ &= \alpha^2(1 - (\alpha - \gamma)\beta r^2) \end{aligned} \tag{36}$$

hence, we can rewrite the solution as:

$$z = \frac{1 - \alpha^2\beta r^2}{\alpha\sqrt{1 - (\alpha - \gamma)\beta r^2} + \gamma} \tag{37}$$

Due to the square root, z is defined as a real value when:

$$1 - (\alpha - \gamma)\beta r^2 \geq 0 \tag{38}$$

If $\alpha \leq 0.5$ then $\alpha - \gamma \leq 0$ and (38) is always true. On the other hand if $\alpha \geq 0.5$ then z is defined for:

$$r^2 \leq \frac{1}{(\alpha - \gamma)\beta} \tag{39}$$

It is so because the projection curve for $\alpha > 0.5$ is an ellipse, so the projection relation is not surjective.

D. Homography matrix

In [6] it was shown that the unified model allows us to use homography matrix to perform 3D reconstruction. The proposed model also allows us to do it. Suppose that we have set of points $\boldsymbol{X}_{1..N}$ belonging to a plane and we observe it with two calibrated cameras a and b (with projection functions defined by η_a and η_b respectively). Let us define the plane by the following equation:

$$\boldsymbol{n}^a \cdot \boldsymbol{X}^a = d_a \tag{40}$$

Here superscript a stands for the first camera's frame; we suppose $d_a \neq 0$. Then we can apply the projection model:

$$\boldsymbol{X}_p^a = \frac{\boldsymbol{X}^a}{\eta_a(\boldsymbol{X}^a)} \tag{41}$$

On the other hand we can reconstruct $\boldsymbol{X}_p$ from a point $\boldsymbol{m} = (x_m \;\; y_m \;\; 1)$ from the normalized pale M (see Fig. 1) using (37):

$$\boldsymbol{X}_p = \begin{pmatrix} x_m \\ y_m \\ z(x_m, y_m) \end{pmatrix} \tag{42}$$

By computing the scalar product with $\boldsymbol{n}^a$ for both sides of (41) and applying (40) we get:

$$\frac{1}{\eta_a(\boldsymbol{X}^a)} = \frac{\boldsymbol{n}^a \cdot \boldsymbol{X}_p^a}{d_a} \tag{43}$$

The transformation between frames a and b is defined as:

$$\boldsymbol{X}^b = R_a^b \boldsymbol{X}^a + \boldsymbol{t}_a^b \tag{44}$$

By replacing $\boldsymbol{X}$ by $\boldsymbol{X}_p$ and using (43) we obtain:

$$\frac{\eta_b(\boldsymbol{X}^b)}{\eta_a(\boldsymbol{X}^a)} \boldsymbol{X}_p^b = R_a^b \boldsymbol{X}_p^a + \frac{\boldsymbol{n}^a \cdot \boldsymbol{X}_p^a}{d_a} \boldsymbol{t}_a^b = \left(R_a^b + \frac{\boldsymbol{t}_a^b (\boldsymbol{n}^a)^T}{d_a} \right) \boldsymbol{X}_p^a \tag{45}$$

We can rewrite the last equation as:

$$\boldsymbol{X}_p^b \propto H_a^b \boldsymbol{X}_p^a \tag{46}$$

where $H_a^b = R_a^b + \frac{\boldsymbol{t}_a^b (\boldsymbol{n}^a)^T}{d_a}$ is a 3×3 homography matrix. Reconstructing $\boldsymbol{X}_{p1..N}$ for both cameras a and b we can get the following system of equations:

$$\begin{cases} \lambda_1 \boldsymbol{X}_{p1}^b = H_a^b \boldsymbol{X}_{p1}^a \\ \vdots \\ \lambda_N \boldsymbol{X}_{pN}^b = H_a^b \boldsymbol{X}_{pN}^a \end{cases} \tag{47}$$

Here $\lambda_{1..N}$ are some unknown scale factors. As matrix H_a^b defined up to a scale factor, we have $8 + N$ unknowns and $3N$ equations. So, we can estimate H_a^b using just 4 points.

E. Jacobian Matrix

Our projection relation is:

$$\begin{aligned} \boldsymbol{m} &= \begin{pmatrix} \frac{x}{\eta} & \frac{y}{\eta} \end{pmatrix}^T \\ \eta &= \gamma z + \alpha \rho \\ \rho &= \sqrt{\beta(x^2 + y^2) + z^2} \end{aligned} \tag{48}$$

First let us compute the partial derivatives of ρ:

$$\frac{\partial \rho}{\partial x} = \frac{\beta x}{\rho} \qquad \frac{\partial \rho}{\partial y} = \frac{\beta y}{\rho} \qquad \frac{\partial \rho}{\partial z} = \frac{z}{\rho} \tag{49}$$

Then by applying the chain rule and noticing that $\frac{\partial \eta}{\partial \rho} = \alpha$ and $\frac{\partial \eta}{\partial z} = \gamma$ we can compute the Jacobian matrix:

$$\begin{aligned} \frac{\partial \boldsymbol{m}}{\partial \boldsymbol{X}} &= \begin{pmatrix} \frac{1}{\eta} - \frac{\alpha\beta x^2}{\eta^2 \rho} & -\frac{\alpha\beta xy}{\eta^2 \rho} & -\frac{x}{\eta^2}\left(\gamma + \frac{\alpha z}{\rho}\right) \\ -\frac{\alpha\beta xy}{\eta^2 \rho} & \frac{1}{\eta} - \frac{\alpha\beta y^2}{\eta^2 \rho} & -\frac{y}{\eta^2}\left(\gamma + \frac{\alpha z}{\rho}\right) \end{pmatrix} \\ \frac{\partial \boldsymbol{p}}{\partial \boldsymbol{X}} &= \begin{pmatrix} f_u & 0 \\ 0 & f_v \end{pmatrix} \frac{\partial \boldsymbol{m}}{\partial \boldsymbol{X}} \end{aligned} \tag{50}$$

This Jacobian matrix allows us to speedup a bundle adjustment process in a SLAM system because it is more efficient than numeric or automatic differentiation.

IV. CALIBRATION USING THE MODEL

This section aims to show that the model can adequately approximate the projection relations of a wide range of different lenses. We don't try to show that the model performs better than the existing ones. Moreover, The unified model is supposed to fit the data better because it has more complex projection relation.

To test the model, a calibration of several different lenses was performed. In all cases sub-pixel precision was achieved. All the example images here are taken with a Fujinon FE185C057HA-1 lens (in Table I it goes second). Below we describe the calibration methodology.

The calibration was done using a check-board calibration pattern. In order to perform the calibration a software was developed . It was written in C++ in a manner such that it is easy to reuse it to perform calibration of a different model. To extract the image coordinates of the pattern a standard function of the *OpenCV* library was used.

Actually it is a practical way to extract the pattern, alternative to the manual corner selection. But it has its drawbacks, first of all poor precision. Relatively high reprojection errors for some corners are explained by this fact. But the overall quality of extraction together with a large dataset (about 100 images) allow us to obtain a precise parameter estimation.

A. Non-linear Optimization Problem

Given a reference model of the calibration board $\{\boldsymbol{X}_i \in \mathbb{R}^3\}$ and a set of projections of the model $\{\boldsymbol{p}_{ij} \in \mathbb{R}^2\}$, (where i stands for the corner index in the board, j is the index of the positions from which the image was taken), and a projection model:

$$f : \mathbb{R}^3 \times \mathbb{R}^N \to \mathbb{R}^2 \tag{51}$$

where N is the number of projection model parameters, find a set of transformation $\{T_j^c \in SE(3)\}$ (superscript c stands for "camera") and a vector of numeric parameters $\boldsymbol{\alpha} \in \mathbb{R}^N$ such that it minimizes the error function:

$$E = \sum_i \sum_j \|\boldsymbol{p}_{ij} - f_k(T_j^c \boldsymbol{X}_i, \boldsymbol{\alpha})\|^2 \tag{52}$$

Here $\boldsymbol{\alpha}$ is the vector of intrinsic parameters, and the transformations represent the extrinsic ones.

The *ceres-solver* library [7] was used to carry out the optimization part. The library provides so-called automatic derivation which allows to compute the Jacobian matrix of the error vector exactly with no explicit symbolic definition of it. (for example, see [8]). This allows us to use the same calibration software to calibrate another camera model by simply changing the projection relations.

B. Initial Approximation

To achieve better convergence the initial extrinsic parameters of the calibration board were estimated using fixed projection parameters.

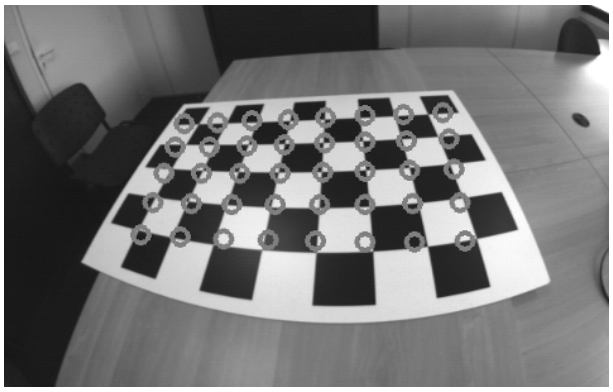

Fig. 3: Initial approximation of the transformation between the board and the camera — the reprojected grid is represented by gray circles.

As long as initial projection parameter are arbitrary, the position estimation is also poor and cannot be considered as a real estimation. But it brings the initial parameter vector of the global optimization problem to a "valley" from where it will converge to the global optimum.

The extrinsic parameters can be estimated independently for each image, hence, the time consumption of this part is negligible in comparison to the global optimization.

C. Calibration Results

The overall optimization (that is, bundle adjustment) is done in a fraction of a second, while corner extraction takes about one minute in total. The standard deviations of the reprojection errors along x and y axes as well as the number of images used in the calibration are given in Table I. The first three lenses are fisheye, and have high α value. The second and the third are of the same model and their projection parameters are quite close. The last two lenses are low-distortion, narrow-angle, and their α are significantly smaller than for fisheye. Fig. 4 shows projection curves corresponding to the calibrated lenses.

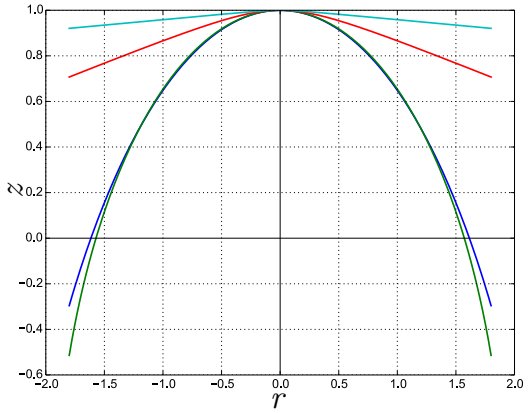

Fig. 4: Projection curves for the calibrated lenses. Blue — first, green — second and third, red — fourth, cyan — fifth.

Fig. 5 is the reprojection of the grid after the calibration. Fig. 6–7 show the undistortion using the model. Fig. 7 shows the undistortion of a region on the border of the image: lines that are straight on the calibration board appear straight. To perform this undistortion we have to rotate the virtual camera with respect to the real camera. Otherwise the region would not be in the field of view of the virtual pinhole camera.

TABLE I: Calibration results. Second and third lines represent different lenses of the same model. The number of images is conditioned by the fact that the *OpenCV* extractor fails to extract the grid from some images and they have to be discarded. The resolution in all cases is 1296×966. σ_x and σ_y represent the reprojection error distribution after the calibration.

Lense model	Images	σ_x, px	σ_y, px	α	β
CF5M1414	88	0.14	0.14	0.571	1.18
FE185C057HA-1	138	0.17	0.16	0.629	1.02
FE185C057HA-1	86	0.31	0.32	0.626	1.03
DF6HA-1B	40	0.20	0.18	0.082	5.50
COMP-M0814-MP	66	0.31	0.33	0.007	46.2

The comparison between the proposed model (11) and unified camera model (6) (with and without the distortion

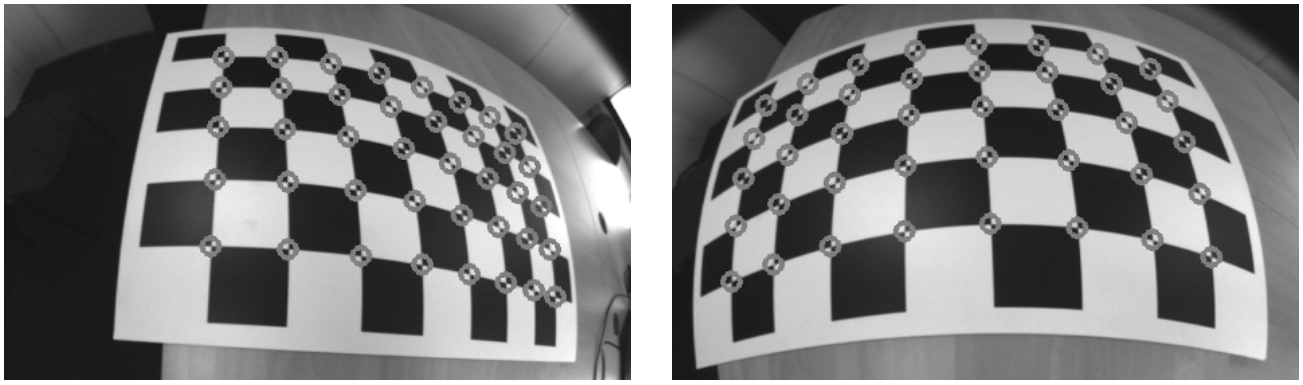

Fig. 5: Grid projection after calibration.

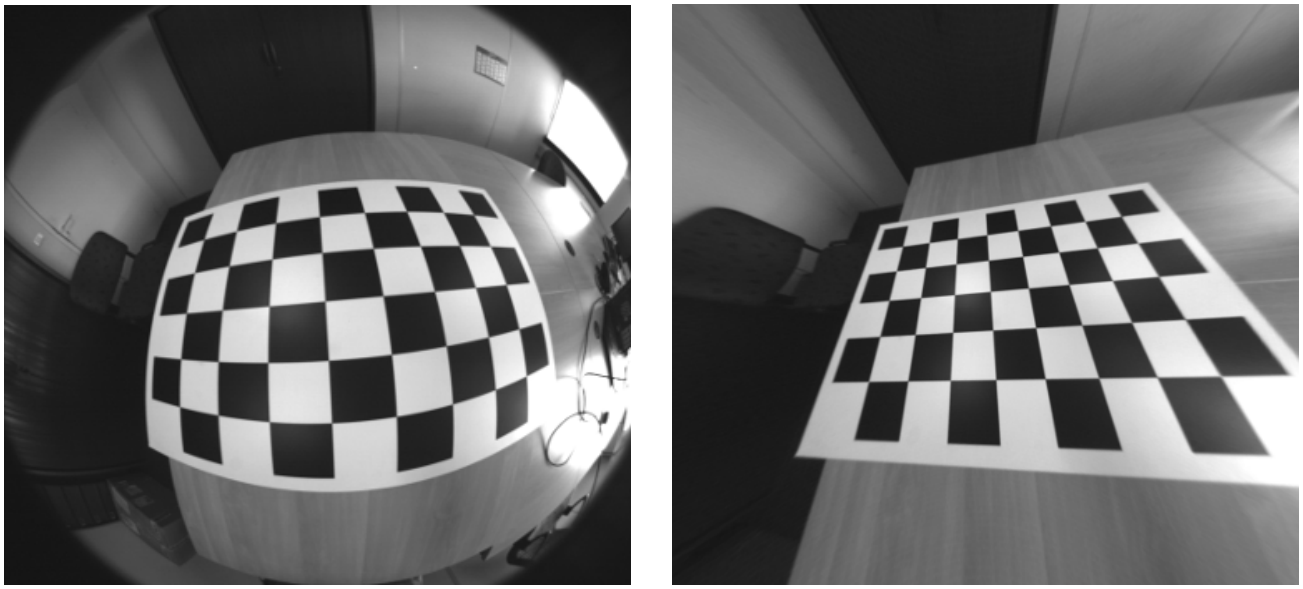

Fig. 6: Undistortion using the calibrated model — all the straight lines after undistortion become straight. Here the board is in the middle of the image.

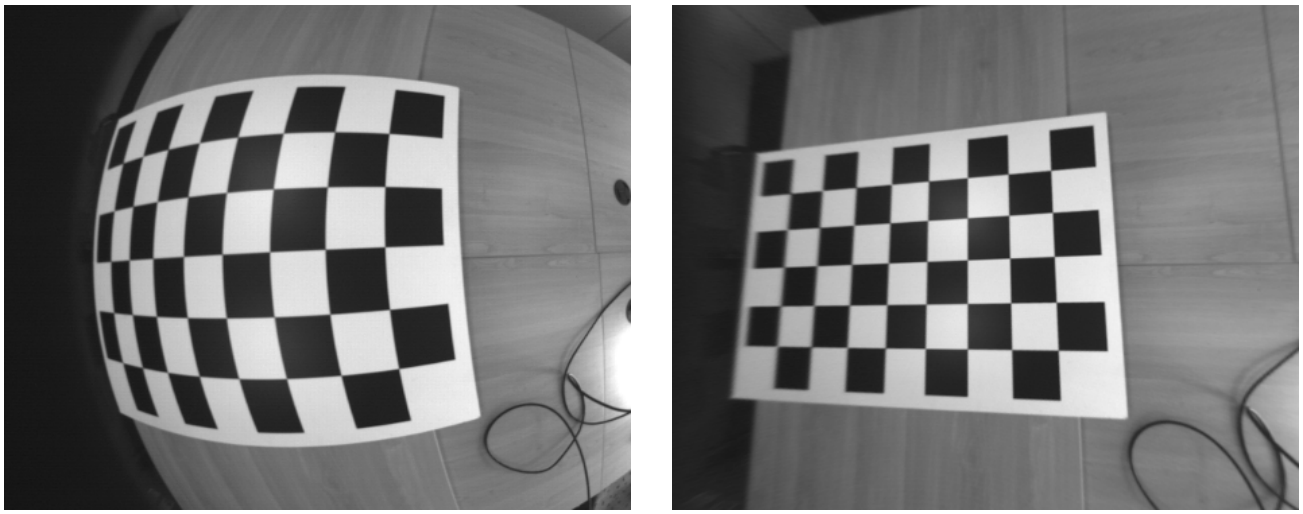

Fig. 7: Undistortion using the calibrated model. Before reprojection the 3D points were rotated. The model works well even on the very border of projection.

layer (7)) is given in Table II. The first thing that we see is that UCM-D does not improve the error. The change in σ_x from EUCM to UCM-D is negligible. On the other hand the computation time increases by an order of magnitude. If we compare UCM and EUCM, then we see that there is no significant change in computation time. But there is a significant improvement of precision for CF5M1414, while the performance of either model is almost the same for FE185C057HA-1. It becomes clear by looking at Table I where for FE185C057HA-1 β is almost 1, while for CF5M1414 it is not. And β is the parameter that makes the difference between two models. We can see a significant improvement for DF6HA-1B (σ_x changes from 0.27 to 0.20).

TABLE II: Calibration results. Here N is the number of images in a dataset; σ_x is the standard deviation of the final error; T is the computation time. The models are: UCM — unified camera model (6), EUCM — enhanced unified camera model (11), UCM-D — unified camera model with distortions (7). The resolution in all cases is 1296×966.

Lense model	N	UCM 5 parameters		EUCM 6 parameters		UCM-D 10 parameters	
		σ_x, px	T, s	σ_x, px	T, s	σ_x, px	T, s
CF5M1414	88	0.52	0.070	0.14	0.086	0.13	0.591
FE185C057HA-1	139	0.18	0.092	0.17	0.220	0.17	1.594
FE185C057HA-1	86	0.32	0.062	0.31	0.079	0.31	0.577
DF6HA-1B	40	0.27	0.021	0.20	0.025	0.19	0.176
COMP-M0814-MP	66	0.32	0.029	0.31	0.100	0.31	1.109

V. CONCLUSIONS

The notion of projection curves and projection surfaces seems to be useful in projection model analysis. The calibration tests show us that the model has a wide application field. It shows the same performance as the unified model with distortions for all the lenses that were used for the experiments. Still, the computation complexity remains at the level of pure unified model.

A general calibration and visual geometry framework (written in C++) has been developed and is available on GitHub: `https://github.com/BKhomutenko/visgeom`. The framework includes the camera model and general calibration tools, so it may be useful in any project that involves camera calibration (regardless of the model to be used).

The model is also successfully used in a stereo visual SLAM project using fisheye cameras. Due to explicit inverse model it allows to rapidly compute stereo reconstruction from a single stereo image, while any model without inversion would require to solve an optimization problem in order to do it.

This model projects straight lines into conics. So, it is possible to formulate epipolar constraints as it is presented in [9]. It may allow us to perform 3D reconstruction and execute RANSAC algorithms efficiently without applying inverse projection mapping.

Also, studying straight line projections may lead us to some visual servoing [10], [11] and pose estimation [12] applications using the model. Finally we strongly believe that the model may improve the modeling of catadioptric systems, and an extension of this work in this way can be done.

REFERENCES

[1] Kannala, J.; Brandt, S.S., "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," in Pattern Analysis and Machine Intelligence, IEEE Transactions on , vol.28, no.8, pp.1335-1340, Aug. 2006

[2] Christopher Geyer, Kostas Daniilidis, "A Unifying Theory for Central Panoramic Systems and Practical Applications," ECCV '00 Proceedings of the 6th European Conference on Computer Vision-Part II Pages 445-461, 26 Jul. 2000

[3] Mei, C.; Rives, P., "Single View Point Omnidirectional Camera Calibration from Planar Grids," Robotics and Automation, 2007 IEEE International Conference on , vol., no., pp.3945,3950, 10-14 Apr. 2007

[4] Courbon, J.; Mezouar, Y.; Eck, L.; Martinet, P., "A generic fisheye camera model for robotic applications," Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on , vol., no., pp.1683,1688, 29 Oct. 2007

[5] Scaramuzza, D.; Martinelli, A.; Siegwart, R., "A Toolbox for Easily Calibrating Omnidirectional Cameras," in Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on , vol., no., pp.5695-5701, 9-15 Oct. 2006

[6] Courbon, J.; Mezouar, Y.; Martinet, P., "Evaluation of the Unified Model of the Sphere for Fisheye Cameras in Robotic Applications," Advanced Robotics, 26:8-9, 947-967, 2012

[7] Sameer Agarwal, Keir Mierle, Others, "Ceres Solver", `http://ceres-solver.org`

[8] Dan Piponi, "Automatic Differentiation, C++ Templates, and Photogrammetry," Journal of Graphics Tools Vol. 9, Iss. 4, 2004

[9] Svoboda T.; Pajdla T., "Epipolar Geometry for Central Catadioptric Cameras," Int. J. Comput. Vision, Vol. 49, No. 1. (August 2002), pp. 23-37

[10] Andreff N.; Espiau B.; and Horaud R., "Visual servoing from lines," International Journal of Robotics Research, 21(8):679700, August 2002.

[11] Hadj-Abdelkader, H.; Mezouar, Y.; Andreff, N.; Martinet, P., "Omnidirectional visual servoing from polar lines," in Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on , vol., no., pp.2385-2390, 15-19 May 2006

[12] Stphane Christy; Radu Horaud, "Iterative Pose Computation from Line Correspondences," Journal of Computer Vision and Image Understanding, Elsevier Science Inc., New York, Volume 73 Issue 1, Pages 137-144, Jan. 1999,