A pose estimation method for fisheye cameras based on the EPnP algorithm

Chenxiao Wang$^{1,2,3[0009-0001-2623-138X]}$, Biao Wang$^{4[0000-0001-6844-153X]}$, Yulong Ding$^{*1,2,3[0000-0001-9178-9892]}$, Dunhui Xiao$^{1,2,3,5[0000-0003-2461-523X]}$, Ben M. Chen$^{6[0000-0002-3839-5787]}$, and Keping Zhang$^{7[0009-0006-6361-807X]}$

[1] Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, Shanghai, China
{chenxiaowang,dingyulong,xiaodunhui}@tongji.edu.cn
[2] National Key Laboratory of Autonomous Intelligent Unmanned Systems
[3] Frontiers Science Center for Intelligent Autonomous Systems, Ministry of Education
[4] College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, China
wangbiao@nuaa.edu.cn
[5] School of Mathematical Sciences, Key Laboratory of Intelligent Computing and Applications(Ministry of Education), Tongji University, Shanghai, China
[6] Department of Mechanical and Automation Engineering, the Chinese University of Hong Kong, Hong Kong, China
bmchen@cuhk.edu.hk
[7] State Grid Nanjing Lishui District Power Supply, Nanjing, Jiangsu, China
15951689969@163.com

Abstract. Fisheye cameras offer a wider field of view compared to traditional pinhole cameras. This paper presents a pose estimation method for fisheye cameras based on the EPnP algorithm. A unified projection model is used to analyze the imaging process, and it is shown to be applicable to fisheye cameras. A virtual pinhole camera is constructed, where the image captured by the fisheye camera is mapped onto the virtual camera's image. The resulting virtual 2D reference points are then used as input to the EPnP algorithm for estimating the fisheye camera's pose. The mapping error from the fisheye image to the virtual camera image is analyzed, revealing that the process does not amplify pixel error when extracting 2D reference points. It is recommended to select points near the image's central focus as 2D reference points. Experimental results demonstrate that the proposed method is both reliable and efficient.

Keywords: Fisheye cameras · Pose estimation · Perspective-n-Point.

1 Introduction

Camera pose estimation is a technique used to determine the position and orientation of a camera relative to its environment based on images. It plays a critical role in environmental perception for unmanned systems. The Perspective-n-Point (PnP) problem involves estimating the pose of a calibrated camera using a set of n 3D points and their corresponding 2D projections in the image. Numerous researchers have proposed methods to solve this problem. For example, Quan et al. [14] and Gao et al. [6] solved the specialized P3P problem; Fiore addressed the PnP problem for arbitrary values of n with a time complexity of $O(n^2)$[5], though this method is sensitive to noise; Ansar et al. proposed a high-accuracy set of linear solutions to the pose estimation problem for both n points and n lines[1].

In 2009, Lepetit et al. introduced the EPnP algorithm[11], a non-iterative solution to the PnP problem with linear time complexity $O(n)$. Since then, this algorithm has been widely adopted for camera pose estimation. Most non-iterative PnP algorithms first determine the depth of feature points in the camera image to calculate the 3D coordinates of the feature points in the camera coordinate system. In contrast, the EPnP algorithm represents the 3D reference points in the world coordinate system as a weighted sum of virtual control points. Typically, four virtual control points are required, which must not lie on the same plane. By solving for the coordinates of these control points in the camera coordinate system, the camera's pose can then be estimated.

Building on Lepetit et al.'s work, numerous researchers have proposed variations of pose estimation methods based on the EPnP algorithm. For instance, Penate-Sanchez et al. introduced a pose estimation algorithm for cameras with unknown focal lengths[13] in 2013; Deng et al. designed a pose estimation method for spherical panoramic images in 2016[4]; Gong et al. improved the EPnP algorithm for omnidirectional cameras in 2021[8]. In practice, applying the EPnP algorithm is typically just one step in the broader pose estimation process, as it requires the 3D coordinates of reference points in space, which usually depend on the depth information of the images.

Currently, unmanned systems commonly employ traditional cameras based on the pinhole imaging model, which offers simplicity and minimal distortion but has a limited field of view. This constraint can lead to insufficient environmental data, resulting in suboptimal motion planning and delayed control responses. In contrast, fisheye cameras, with extremely short focal lengths—usually less than 16mm—produce hemispherical images with significant visual distortion and an angle of view approaching or reaching $180°$. Compared to pinhole cameras, fisheye cameras capture a far wider range of environmental information, making them increasingly popular in unmanned systems.

Fisheye cameras project the largest possible scene onto a limited image plane using specific projection functions. Based on these functions, fisheye camera models are generally categorized into four types: equidistant, equisolid (equal area), orthographic, and stereographic. In practice, fisheye cameras do not perfectly conform to these models, leading researchers to propose specialized models for fisheye cameras. In 2006, Kannala and Brandt introduced a universal camera model where the distance from the optical center to the projection point is proportional to the angle between the projection ray and the principal axis, expressed as a polynomial[9]. That same year, Scaramuzza et al. proposed a model assuming that the camera and lens axes are perfectly aligned and the lens is rotationally symmetric about its axis[15]. This model maps points in space to corresponding points on the image plane using a Taylor series expansion. In 2007, Mei and Rives proposed a method for calibrating single omnidirectional cameras using a planar chessboard[12], introducing a spherical projection model. This model incorporates precise theoretical projection functions while adding parameters to simulate real-world errors, proving effective for fisheye and spherical cameras.

Employing fisheye cameras for pose estimation enables the utilization of environmental information over a broader spatial range, which benefits the localization of unmanned systems and thus holds significant research value. The severe radial distortion in fisheye images makes it difficult to apply the EPnP algorithm directly. This paper proposes a pose estimation method tailored for fisheye cameras, based on the EPnP algorithm with a time complexity of $O(n)$. First, we establish a fisheye camera model using the unified projection model introduced by Mei et al. in Section 2. In Section 3, we propose a pose estimation method for fisheye cameras: we construct a virtual pinhole camera, map the image captured by the fisheye camera to the virtual pinhole camera's image, and use the resulting virtual 2D reference points as input to the EPnP algorithm to estimate the pose. Section 4 analyzes the error arising from mapping fisheye camera images to virtual pinhole camera images. Finally, in Section 5, experiments validate the proposed method's stability and efficiency compared to OpenCV.

2 Fisheye Camera Model

Fisheye cameras produce images with significant radial distortion, rendering them incompatible with the traditional pinhole camera model. Mei et al. developed a spherical projection model for omnidirectional cameras, known as the Unified Projection Model [12], building on the projection models proposed by Barreto et al. and Geyer et al. [2, 7]. They also introduced a calibration method using planar grids to support this model.

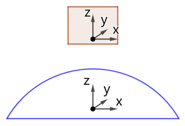


Fig. 1: Selection of the Cartesian coordinate system

Consider the Cartesian coordinate system depicted in Figure 1. As illustrated in Figure 2, a point $X(x, y, z)$ in the world can be projected onto the camera's image plane through the following steps:

(1) A 3D point $X(x, y, z)$ in the world coordinate system with origin $O_s = (0, 0, 0)$ is projected onto the unit sphere, $X \to X_s = \frac{X}{||X||} = (x_s, y_s, z_s)$, where $||X||$ represents the norm of X.

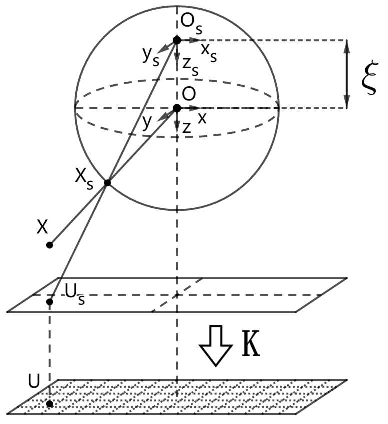


Fig. 2: Unified Projection Model

(2) In a new coordinate system with origin $O_s = (0, 0, -\xi)$, $(X_s)_{\mathcal{F}_O} \rightarrow (X_s)_{\mathcal{F}_{O_s}} = (x_s, y_s, z_s + \xi)$.

(3) The point X_s is projected onto the normalized plane to obtain the point U_s, $U_s = \hbar(X_s) = (\frac{x_s}{z_s+\xi}, \frac{y_s}{z_s+\xi}, 1)$.

(4) The final projection involves the camera projection matrix K, projecting U_s to U on the camera image plane,

$$U = k(U_s) = KU_s = \begin{bmatrix} f_1\eta & f_1\eta\alpha & u_0 \\ 0 & f_2\eta & v_0 \\ 0 & 0 & 1 \end{bmatrix} U_s, \tag{1}$$

where $[f_1, f_2]^T$ are the focal lengths in the X and Y directions, (u_0, v_0) is the principal point, and α is the skew coefficient, typically set to 0.

Thus, the relationship between a 3D point X and the corresponding point U on the camera image plane can be expressed as:

$$U = K \cdot \hbar(\frac{X}{||X||}). \tag{2}$$

In this process, ξ and η are parameters related to the camera lens. Table 1 presents the values of ξ and η for different lenses, while Table 2 lists the equations satisfied by these parameters for various lenses, where d represents the distance between the two focal points of the lens, and $4p$ denotes the latus rectum.

A generalized camera projection matrix K is used, treating the camera and its lens as a single entity rather than separate components. As a result, f and η cannot be independently estimated during calibration. We define $f_x = f_1\eta$ and $f_y = f_2\eta$ accordingly.

Mei et al. demonstrated that the Unified Projection Model is applicable to fisheye cameras, building on the work of Ying et al. and Brauer-Burchardt et

Table 1: Parameters for different lenses

Lens Type	ξ	η
Parabolic	1	$-2p$
Hyperbolic	$\frac{d}{\sqrt{d^2+4p^2}}$	$\frac{-2p}{\sqrt{d^2+4p^2}}$
Elliptical	$\frac{d}{\sqrt{d^2+4p^2}}$	$\frac{2p}{\sqrt{d^2+4p^2}}$
Planar	0	-1

Table 2: Lens Parameter Equations

Lens Type	Equation
Parabolic	$\sqrt{x^2+y^2+z^2} = z + 2p$
Hyperbolic	$\frac{(z+\frac{d}{2})^2}{a^2} - \frac{x^2}{b^2} - \frac{y^2}{b^2} = 1$
Elliptical	$\frac{(z+\frac{d}{2})^2}{a^2} + \frac{x^2}{b^2} + \frac{y^2}{b^2} = 1$
Planar	$z = -\frac{d}{2}$

al.[3, 12, 16]. By performing calibration using planar grids, the fisheye camera's intrinsic parameters, including the projection matrix K and the lens parameter ξ, can be accurately obtained.

3 Pose Estimation Method For Fisheye Cameras

After applying the Unified Projection Model for fisheye cameras, we propose a pose estimation method tailored to fisheye cameras based on the EPnP algorithm. The inputs for the algorithm are as follows:

(1) Coordinates of n 3D reference points in the world coordinate system;
(2) Coordinates of n 2D reference points on the fisheye camera image corresponding to the 3D reference points;
(3) The projection matrix K and lens parameter ξ of the fisheye camera.

First of all, n points are selected as 3D reference points in the world coordinate system. Then, n corresponding points are extracted from the fisheye camera image, resulting in n 2D reference points.

The EPnP pose estimation algorithm proposed by Lepetit et al. has a time complexity of $O(n)$ [11], making it computationally efficient. However, the camera model used in the EPnP algorithm is the conventional pinhole camera model, which is not suitable for fisheye cameras.

3.1 Mapping to Virtual Image Plane

To apply the EPnP algorithm, a virtual pinhole camera is constructed. Based on the Unified Projection Model described in Section 2, a virtual image plane u^{vir} is created, and the mapping f from the fisheye camera image plane u to u^{vir} is defined. For simplicity, we set the distance from u^{vir} to the origin O of the camera coordinate system to 1, and ensure that u^{vir} is parallel to u, meaning that it is positioned at the plane $z = 1$. The point $[0, 0, 1]^T$ is chosen as the origin of u^{vir}, as illustrated in Figure 3.

We now begin the process of mapping a point U on the fisheye camera image plane u to its corresponding point U_{vir} on the virtual image plane u^{vir}. First, map the point U to a corresponding point U_s on the normalized plane in the Unified Projection Model, denoted as f_1:

$$U_s = f_1(U) = K^{-1}U, \tag{3}$$

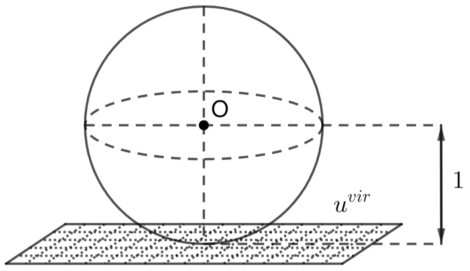


Fig. 3: Virtual image plane

where K is the projection matrix of the fisheye camera.

Then, map U_s to a point X_s on the unit sphere s, denoted as f_2. In fact, f_2 is the inverse mapping of $\hbar$ described in Section 2:

$$X_s = f_2(U_s) = \hbar^{-1}(U_s) = \begin{bmatrix} \frac{\xi+\sqrt{1+(1+\xi^2)({u_s}^2+{v_s}^2)}}{{u_s}^2+{v_s}^2+1} \cdot u_s \\ \frac{\xi+\sqrt{1+(1+\xi^2)({u_s}^2+{v_s}^2)}}{{u_s}^2+{v_s}^2+1} \cdot v_s \\ \frac{\xi+\sqrt{1+(1-\xi^2)({u_s}^2+{v_s}^2)}}{{u_s}^2+{v_s}^2+1} - \xi \end{bmatrix}, \tag{4}$$

where (u_s, v_s) are the coordinates of U_s, and ξ is the lens parameter.

Finally, map X_s to a point U_{vir} on u^{vir}, denoted as f_3:

$$U_{vir} = f_3(X_s) = \begin{bmatrix} \frac{x_s}{z_s} \\ \frac{y_s}{z_s} \\ 1 \end{bmatrix}, \tag{5}$$

where $[x_s, y_s, z_s]^T$ represents the coordinate of X_s, satisfying the condition $x_s^2 + y_s^2 + z_s^2 = 1$.

The mapping from a point U on the fisheye camera image plane u to a point U_{vir} on the virtual image plane u^{vir} has now been completed, denoted as mapping f. This mapping f is the composition of the mappings f_1, f_2, and f_3:

$$U_{vir} = f(U) = f_3(f_2(f_1(U))) = f_3(\hbar^{-1}(K^{-1}U)). \tag{6}$$

For the n 2D reference points on u, calculate the corresponding n virtual 2D reference points on u^{vir} using the mapping f. With these n 3D reference points in the world coordinate system and the n virtual 2D reference points on u^{vir} as inputs, the EPnP algorithm can be applied to determine the rotation matrix R and translation vector t of the fisheye camera relative to the world coordinate system, thus providing the pose of the fisheye camera.

3.2 Parameterization

Conventionally, four virtual control points are selected. Let the coordinates of the n 3D reference points in the world coordinate system be denoted as $p_i^w(i =$

$1, 2, ..., n$), and let the coordinates of the four virtual control points be denoted as $c_j^w (j = 1, 2, 3, 4)$, where p_i^w and c_j^w are given in non-homogeneous coordinates.

In the world coordinate system, each 3D reference point is represented as a weighted sum of the virtual control points:

$$p_i^w = \sum_{j=1}^{4} \alpha_{ij} c_j^w, \quad \sum_{j=1}^{4} \alpha_{ij} = 1, \tag{7}$$

where α_{ij} are referred to as Homogeneous Barycentric Coordinates. Once four non-coplanar virtual control points are determined, the coordinates α_{ij} are uniquely determined.

Let the coordinates of the n 3D reference points in the camera coordinate system be denoted as $p_i^c (i = 1, 2, ..., n)$, and the coordinates of the four virtual control points be denoted as $c_j^c (j = 1, 2, 3, 4)$, where p_i^c and c_j^c are given in non-homogeneous coordinates. In the camera coordinate system, a similar relationship holds:

$$p_i^c = \sum_{j=1}^{4} \alpha_{ij} c_j^c. \tag{8}$$

Converting Equation 7 to matrix form:

$$\begin{bmatrix} p_i^w \\ 1 \end{bmatrix} = C \begin{bmatrix} \alpha_{i1} \\ \alpha_{i2} \\ \alpha_{i3} \\ \alpha_{i4} \end{bmatrix} = \begin{bmatrix} c_1^w & c_2^w & c_3^w & c_4^w \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha_{i1} \\ \alpha_{i2} \\ \alpha_{i3} \\ \alpha_{i4} \end{bmatrix}, \tag{9}$$

where $\begin{bmatrix} p_i^w & 1 \end{bmatrix}$ and $\begin{bmatrix} c_j^w & 1 \end{bmatrix}$ are both homogeneous coordinates. This implies that the homogeneous coordinates of the 3D reference points are a linear combination of the homogeneous coordinates of the virtual control points.

From Equation 9, the formula for calculating the Homogeneous Barycentric Coordinates α_{ij} is given by:

$$\begin{bmatrix} \alpha_{i1} \\ \alpha_{i2} \\ \alpha_{i3} \\ \alpha_{i4} \end{bmatrix} = C^{-1} \begin{bmatrix} p_i^w \\ 1 \end{bmatrix}. \tag{10}$$

Now, choose the virtual control points. In the world coordinate system, the centroid of the n 3D reference points is selected as the first virtual control point:

$$c_1^w = \frac{1}{n} \sum_{i=1}^{n} p_i^w. \tag{11}$$

Compute the matrix

$$A = \begin{bmatrix} {p_1^w}^T - {c_1^w}^T \\ \cdots \\ {p_n^w}^T - {c_1^w}^T \end{bmatrix}. \tag{12}$$

Let $\lambda_i (i = 1, 2, 3)$ be the eigenvalues of $A^T A$, and $v_i (i = 1, 2, 3)$ be the corresponding eigenvectors. The remaining three virtual control points are then determined using the following formulas:

$$\begin{cases} c_2^w = c_1^w + \sqrt{\lambda_1} v1, \\ c_3^w = c_1^w + \sqrt{\lambda_2} v2, \\ c_4^w = c_1^w + \sqrt{\lambda_3} v3. \end{cases} \tag{13}$$

3.3 Solution of Virtual Control Points Coordinates in Camera Coordinate System

Let $u_i i = 1, ..., n$ be the virtual 2D reference points corresponding to the 3D reference points $p_i^c i = 1, ..., n$ on u^{vir}. It is important to note that the camera model used here should be the virtual pinhole camera model. Therefore, the projection matrix used should be the projection matrix K_{vir} of the virtual pinhole camera. Since the distance from the virtual image plane u^{vir} to the origin O of the camera coordinate system is 1, the focal length of the virtual camera is also 1. Additionally, the coordinates of the origin of u^{vir} in the camera coordinate system are $[0, 0, 1]^T$. Consequently, the projection matrix K_{vir} is the identity matrix I_3. According to the pinhole camera model, we have

$$\forall i, \quad w_i \begin{bmatrix} u_i \\ 1 \end{bmatrix} = K p_i^c = K \sum_{j=1}^{4} \alpha_{ij} c_j^c, \tag{14}$$

where w_i is a scalar depth parameter.

Let $c_j^c = \begin{bmatrix} x_j^c \; y_j^c \; z_j^c \end{bmatrix}^T$, $K = K_{vir} = I_3$, then:

$$\forall i, \quad w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = I_3 \sum_{j=1}^{4} \alpha_{ij} c_j^c \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix} = \sum_{j=1}^{4} \alpha_{ij} c_j^c \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix}. \tag{15}$$

By $w_i = \sum_{j=1}^{4} \alpha_{ij} z_j^c$ and Equation 15, we obtain two linear equations:

$$\begin{cases} \sum_{j=1}^{4} \left[\alpha_{ij} x_j^c - \alpha_{ij} u_i z_j^c \right] = 0, \\ \sum_{j=1}^{4} \left[\alpha_{ij} y_j^c - \alpha_{ij} v_i z_j^c \right] = 0. \end{cases} \tag{16}$$

For each virtual 2D reference point u_i on the virtual image plane, we can derive 2 linear equations. Thus, with n virtual 2D reference points $u_i (i = 1, ..., n)$, we obtain a total of $2n$ linear equations. By combining these $2n$ linear equations, we form a linear system:

$$\begin{bmatrix} \alpha_{11} & 0 & -\alpha_{11}u_1 & \ldots & \alpha_{14} & 0 & -\alpha_{14}u_1 \\ 0 & \alpha_{11} & -\alpha_{11}v_1 & \ldots & 0 & \alpha_{14} & -\alpha_{14}v_1 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \alpha_{n1} & 0 & -\alpha_{n1}u_n & \ldots & \alpha_{n4} & 0 & -\alpha_{n4}u_n \\ 0 & \alpha_{n1} & -\alpha_{n1}v_n & \ldots & 0 & \alpha_{n4} & -\alpha_{n4}v_n \end{bmatrix}_{2n\times 12} \begin{bmatrix} x_1^c \\ y_1^c \\ z_1^c \\ \ldots \\ x_4^c \\ y_4^c \\ z_4^c \end{bmatrix}_{12\times 1} = 0. \tag{17}$$

Let M denote the first matrix on the left-hand side of Equation 17. The vector $x = \left[c_1^{c^T} \; c_2^{c^T} \; c_3^{c^T} \; c_4^{c^T} \right]^T$ contains the coordinates of the 4 virtual control points in the camera coordinate system. Then,

$$Mx = 0. \tag{18}$$

Thus, x lies in the right null space of the matrix M, i.e., $x \in \ker(M)$.

Suppose that the matrix M^TM has N zero eigenvalues, and let $v_k(k = 1, ..., N)$ be the eigenvectors corresponding to the k-th zero eigenvalues of matrix M^TM. Then,

$$x = \sum_{k=1}^{N} \beta_k v_k, \tag{19}$$

where β_k are coefficients.

For the i-th virtual control point,

$$c_i^c = \sum_{k=1}^{N} \beta_k v_k^{[i]}, \quad i = 1, 2, 3, 4, \tag{20}$$

where $v_k^{[i]}$ is the i-th 3×1 block submatrix of eigenvector v_k.

To solve for v_k, compute the eigenvalues and eigenvectors of the matrix M^TM. The eigenvectors corresponding to the zero eigenvalues are v_k. The number of zero eigenvalues N of the matrix M^TM depends on the camera focal length and the number of 3D-to-2D correspondences. The EPnP algorithm addresses cases where $N = 1, 2, 3, 4$. Note that the matrix M^TM is 12×12, and calculating M^TM has a computational complexity of $O(n)$, resulting in an overall complexity of $O(n)$ for the pose estimation method.

Then, solve for β_k (for $k = 1, \ldots, N$). Since the distance between any two virtual control points is invariant in both the world coordinate system and the camera coordinate system, we have

$$||c_i^w - c_j^w||^2 = ||c_i^c - c_j^c||^2 = ||\sum_{k=1}^{N} \beta_k v_k^{[i]} - \sum_{k=1}^{N} \beta_k v_k^{[j]}||^2. \tag{21}$$

This equation is quadratic in terms of $\beta_k(k = 1, ..., N)$ and does not contain linear terms of β_k. By defining $\beta_{ij} = \beta_i\beta_j$, Equation 21 can be transformed into a linear equation in terms of $\beta_{ij}(i, j = 1, ..., N)$. With 4 virtual control points, we obtain $C_4^2 = 6$ linear equations, forming a linear system.

When the number of zero eigenvalues N of the matrix $M^T M$ is 1, 2, or 3, the number of unknowns in the linear system is less than the number of equations, making it straightforward to solve for β_{ij}. However, when $N = 4$, the number of unknowns 10 exceeds the number of equations. In this case, the "linearization" method proposed by Kipnis et al. can be used to solve for β_{ij} [10].

The $\beta_k(k = 1, ..., N)$ obtained in this process are initial estimates and can be further refined using the Gauss-Newton method.

3.4 Camera Pose Computation

First, calculate the coordinates of the virtual control points in the camera coordinate system, c_i^c for $i = 1, 2, 3, 4$, using Equation 20. Then, compute the coordinates of the 3D reference points in the camera coordinate system, p_i^c for $i = 1, 2, ..., n$, using Equation 8.

Next, determine the centroid p_0^w (c_1^w) of the n 3D reference points $p_i^w(i = 1, 2, ..., n)$ in the world coordinate system using Equation 11, and compute the matrix A using Equation 12.

Similarly, calculate the centroid p_0^c of the n 3D reference points $p_i^c(i = 1, 2, ..., n)$ in the camera coordinate system and compute the matrix B:

$$p_0^c = \frac{1}{n}\sum_{i=1}^{n} p_i^c, \quad B = \begin{bmatrix} p_1^{c^T} - p_0^{c^T} \\ \cdots \\ p_n^{c^T} - p_0^{c^T} \end{bmatrix}. \tag{22}$$

Then, compute the matrix H:

$$H = B^T A. \tag{23}$$

Perform Singular Value Decomposition (SVD) on matrix H:

$$H = U\Sigma V^T. \tag{24}$$

Then, calculate the rotation matrix R of the camera coordinate system relative to the world coordinate system:

$$R = UV^T. \tag{25}$$

If $\det(R) < 0$, adjust R by setting $R(2,:) = -R(2,:)$.

Finally, calculate the translation vector t of the camera coordinate system relative to the world coordinate system:

$$t = p_0^c - Rp_0^w. \tag{26}$$

Note that there are four possible solutions for β, resulting in four different pairs of rotation matrix R and translation vector t. Select the solution that minimizes the reprojection error as the final result. This concludes the computation of the fisheye camera's pose.

4 Error Analysis

This section presents an error analysis of the fisheye camera pose estimation method based on the EPnP algorithm introduced in Section 3. The method consists of three main steps:

(1) Calibrating the fisheye camera using a planar checkerboard;
(2) Constructing a virtual pinhole camera and mapping the fisheye camera images to virtual pinhole camera images, thus obtaining virtual 2D reference points;
(3) Applying the EPnP algorithm to solve for the fisheye camera pose.

Errors introduced during fisheye camera calibration using a planar checkerboard have been analyzed by Mei et al.[12], while errors from applying the EPnP algorithm to estimate the fisheye camera pose have been discussed by Lepetit et al.[11]. Therefore, this section focuses exclusively on analyzing the errors in the mapping function f, which maps a point U on u to a point U_{vir} on u^{vir}.

When extracting 2D reference points from fisheye camera images, errors are introduced. In the following analysis, we examine how these errors affect the mapping f through numerical computations. We use a fisheye camera with a focal length of 1.4 mm, a 180° field of view, an image resolution of 640 × 480 pixels, and a principal point at $(u_0, v_0) = (360.036, 214.209)$.

First, we analyze how varying pixel error magnitudes during the extraction of 2D reference points from fisheye camera images influence the errors when these points are mapped to the virtual image plane U_{vir} via the mapping function f. Numerical computations show that when the pixel error direction during point extraction aligns with the direction from the principal point to the true 2D reference point, the error direction in the mapping f corresponds to the direction from the origin to the 2D reference point on the virtual image plane. Thus, without loss of generality, we choose a point $U(u_0 + 100, v_0)$, located 100 pixels away from the principal point, as the 2D reference point. The error in the distance from the origin to the virtual 2D reference point, induced by varying pixel error lengths from 0 to 3, is then calculated. The results are shown in Figure 4.

The line graph shows that the error introduced by the mapping function f increases linearly with pixel error lengths when extracting 2D reference points. This indicates that the mapping f does not amplify the pixel errors incurred during the extraction of 2D reference points.

Next, we analyze how different positions of 2D reference points, each with the same pixel error length, affect the errors when these points are mapped onto U_{vir} via the mapping f. Numerical computations reveal that, for 2D reference points with the same pixel error length, if their directions align with the direction from the principal point to the 2D reference points, then the errors in pixel lengths remain consistent after mapping f. Therefore, without loss of generality, we select a set of points $\{(u, v)|u = u_0 + x, v = v_0, x \in [10, 250]\}$ as the 2D reference points. Assuming a pixel error length of 0.5 during point extraction, the errors

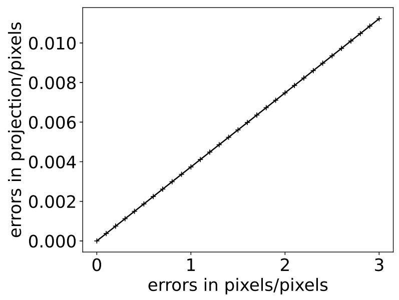


Fig. 4: Errors of mapping f versus pixel errors when extracting 2D reference point

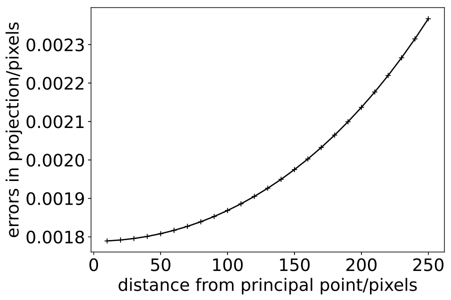


Fig. 5: Errors of mapping f versus distances between 2D reference points and the principal point

in the distances from the origin to the virtual 2D reference points are computed. These results are shown in Figure 5.

The line graph demonstrates that the error introduced by the mapping f increases with the distance between the 2D reference points and the principal point, with an accelerating trend. Consequently, selecting 2D reference points closer to the principal point in the fisheye camera image can help reduce the pixel errors induced by the mapping f. Therefore, it is advisable to choose points nearer to the principal point as 2D reference points.

5 Experiment

We used a fisheye camera with a focal length of 1.4 mm and a 180° field of view, as shown in Fig. 6.

Fig. 6: Fisheye camera

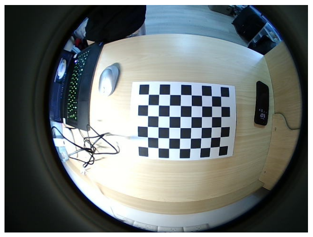


Fig. 7: Fisheye camera photo

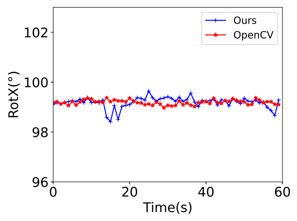
The fisheye camera was initially calibrated using planar grids to determine its projection matrix K and lens parameter ξ. The calibration results are shown in Table 3.

A 7×9 flat chessboard was placed in the environment. The fisheye camera was positioned above the chessboard and kept still for one minute, with the

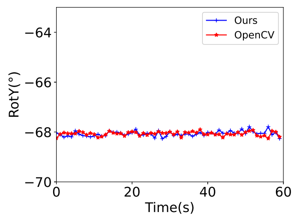
Table 3: Calibration result of fisheye camera

Parameter	Value
K	$\begin{bmatrix} 608.499 & 0.654 & 360.036 \\ 0 & 607.862 & 214.209 \\ 0 & 0 & 1 \end{bmatrix}$
ξ	1.177

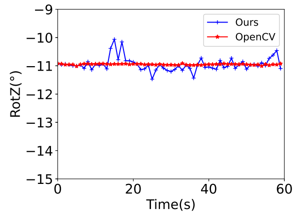
pose estimated every second using the method described in Section 3. An example photo of the setup is shown in Fig. 7. The results are compared with the calibration results obtained using OpenCV 3.0, as depicted in Fig. 8.



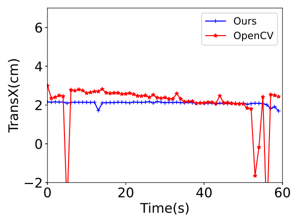
(a) Euler angle around x axis versus time



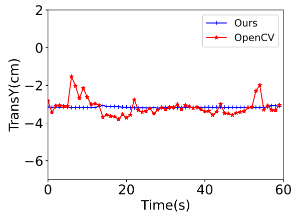
(b) Euler angle around y axis versus time



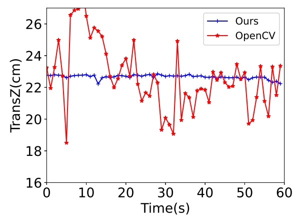
(c) Euler angle around z axis versus time



(d) Translation in x direction versus time



(e) Translation in y direction versus time



(f) Translation in z direction versus time

Fig. 8: Comparison results for the fisheye camera pose estimation

In this experiment, the pose of the fisheye camera remained constant, so the Euler angles and translation vectors should also be consistent. The standard deviations of the Euler angles around the x, y, and z axes, and those of the three components of the translation vectors estimated by our method are 0.221, 0.098, 0.242 and 0.092, 0.028, 0.128, respectively. In comparison, the standard deviations estimated by OpenCV are 0.093, 0.082, 0.019 and 1.250, 0.436, 2.099, respectively. This indicates that our method is significantly more stable than

OpenCV in estimating translation vectors, but slightly less stable in estimating Euler angles. Additionally, our method took 0.927 seconds to estimate the poses of 60 images, which is 13.039% faster than the 1.066 seconds required by OpenCV.

6 Conclusion

In this article, we proposed a pose estimation method for fisheye cameras based on the EPnP algorithm. Our method has a time complexity of $O(n)$ and fully leverages the broader environmental information captured by fisheye cameras. An error analysis was conducted, and an experiment was carried out demonstrating that our method is more stable in estimating translation vectors and more efficient compared to OpenCV.

Future work will involve conducting more comprehensive experiments to further validate the proposed pose estimation method. Additionally, we aim to develop a pose estimation technique for stereo fisheye cameras.

Acknowledgments. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFE0193900, in part by the National Natural Science Foundation of China under Grants 62088101 and 62303354, and in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0100.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Ansar, A., Daniilidis, K.: Linear pose estimation from points or lines. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(5), 578–589 (May 2003)
2. Barreto, J., Araujo, H.: Issues on the geometry of central catadioptric image formation. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001. vol. 2, pp. II–II (Feb 2001)
3. Brauer-Burchardt, C., Voss, K.: A new algorithm to correct fish-eye- and strong wide-angle-lens-distortion from single images. In: Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205). vol. 1, pp. 225–228 vol.1 (Oct 2001)
4. Deng, F., Wu, Y., Hu, Y., Cui, H.: Position and Pose Estimation of Spherical Panoramic Image with Improved EPnP Algorithm. Acta Geodaetica et Cartographica Sinica **45**(6), 677–684 (Jun 2016)
5. Fiore, P.: Efficient linear solution of exterior orientation. IEEE Transactions on Pattern Analysis and Machine Intelligence **23**(2), 140–148 (Feb 2001)
6. Gao, X.S., Hou, X.R., Tang, J., Cheng, H.F.: Complete solution classification for the perspective-three-point problem. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(8), 930–943 (Aug 2003)

7. Geyer, C., Daniilidis, K.: A Unifying Theory for Central Panoramic Systems and Practical Implications. In: Vernon, D. (ed.) Computer Vision — ECCV 2000. pp. 445–461. Springer, Berlin, Heidelberg (2000)
8. Gong, X., Lv, Y., Xu, X., Wang, Y., Li, M.: Pose Estimation of Omnidirectional Camera with Improved EPnP Algorithm. Sensors **21**(12), 4008 (Jan 2021)
9. Kannala, J., Brandt, S.: A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(8), 1335–1340 (Aug 2006)
10. Kipnis, A., Shamir, A.: Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization. In: Wiener, M. (ed.) Advances in Cryptology — CRYPTO' 99. pp. 19–30. Springer, Berlin, Heidelberg (1999)
11. Lepetit, V., Moreno-Noguer, F., Fua, P.: EPnP: An Accurate O(n) Solution to the PnP Problem. International Journal of Computer Vision **81**(2), 155–166 (Feb 2009)
12. Mei, C., Rives, P.: Single View Point Omnidirectional Camera Calibration from Planar Grids. In: Proceedings 2007 IEEE International Conference on Robotics and Automation. pp. 3945–3950 (Apr 2007)
13. Penate-Sanchez, A., Andrade-Cetto, J., Moreno-Noguer, F.: Exhaustive Linearization for Robust Camera Pose and Focal Length Estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(10), 2387–2400 (Oct 2013)
14. Quan, L., Lan, Z.: Linear N-point camera pose determination. IEEE Transactions on Pattern Analysis and Machine Intelligence **21**(8), 774–780 (Aug 1999)
15. Scaramuzza, D., Martinelli, A., Siegwart, R.: A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion. In: Fourth IEEE International Conference on Computer Vision Systems (ICVS'06). pp. 45–45 (Jan 2006)
16. Ying, X., Hu, Z.: Can We Consider Central Catadioptric Cameras and Fisheye Cameras within a Unified Imaging Model. In: Pajdla, T., Matas, J. (eds.) Computer Vision - ECCV 2004. pp. 442–455. Springer, Berlin, Heidelberg (2004)