

Pose Estimation for Omni-directional Cameras using Sinusoid Fitting

Haofei Kuang¹, Qingwen Xu¹, Xiaoling Long¹ and Sören Schwertfeger¹

Abstract—We propose a novel pose estimation method for geometric vision of omni-directional cameras. On the basis of the regularity of the pixel movement after camera pose changes, we formulate and prove the sinusoidal relationship between pixels movement and camera motion. We use the improved Fourier-Mellin invariant (iFMI) algorithm to find the motion of pixels, which was shown to be more accurate and robust than the feature-based methods. While iFMI works only on pin-hole model images and estimates 4 parameters (x , y , yaw, scaling), our method works on panoramic images and estimates the full 6 DoF 3D transform, up to an unknown scale factor. For that we fit the motion of the pixels in the panoramic images, as determined by iFMI, to two sinusoidal functions. The offsets, amplitudes and phase-shifts of the two functions then represent the 3D rotation and translation of the camera between the two images. We perform experiments for 3D rotation, which show that our algorithm outperforms the feature-based methods in accuracy and robustness. We leave the more complex 3D translation experiments for future work.

I. INTRODUCTION

Omni-directional cameras have been widely used in mobile robots. Visual cues from panorama images help the robot to achieve homing in [1], which uses omni-cameras' advantage of the large field of view (FOV). In [2], panorama images are exploited to implement a bearings-only SLAM system, which can provide rich feature points. In addition, several meaningful applications with panorama cameras are mentioned in [3]. In the above cases, pose estimation is one of the important topics. Optimization and geometric methods are two common ways to achieve this task. The former is usually used in feature-based methods [4], while the latter is one of the basic parts in direct methods [5]. These two approaches are usually combined together to achieve better performance. For example, Engel et. al combined both methods to realize real-time robust visual odometry in [6]; [7] uses geometric methods for the front-end of a SLAM system and the optimization method for the back-end. In addition, recently deep learning gained attention in pose estimation [8], [9]. Since the deep learning methods heavily depend on training images, which may be difficult to adapt to different environments within limited training datasets, we do not take the deep learning method into account.

Optimization methods are usually used to minimize the photometric and geometry errors. In direct methods based on photometric consistency, the motion between different frames is estimated by minimizing the photometric error, as used in [5], [6], [10]. Geometry error optimization helps to

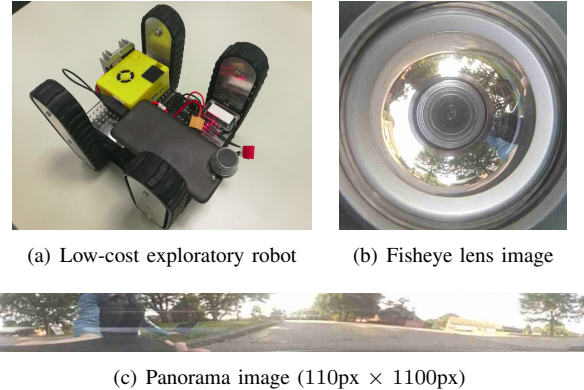


Fig. 1: A tracked robot with a smartphone as control unit. An omni-directional lens is placed over the front-camera of the phone. The captured image as well as the unwrapped panoramic image are shown.

boost pose estimation, such as bundle adjustment [11] and graph optimization [12], [13].

Geometric methods include two main groups: pose estimation by 2D to 2D correspondences and 3D to 2D correspondences. The eight-point [14] and five-point algorithms [15] are widely used to estimate the relative pose between two frames with given 2D-2D correspondences. Firstly, the essential matrix or fundamental matrix is calculated by epipolar geometry. Then relative rotation and up-to-scale translation are estimated from the matrix. Methods like [16], that estimate relative pose by 3D-2D correspondences, compensate the scaling case and calculate the absolute pose directly, because 3D points provide the real scale information.

In addition to feature-based and direct methods, spectral methods are also used for motion estimation. On 2D images, the iFMI algorithm [17], [18] uses Fast Fourier Transform (FFT) to calculate the x and y translation, yaw and scaling between two images. It is applied to image mosaicking [19] and motion detection [20]. A related approach is also used for registration of 3D range data [21].

If we only consider pose estimation between two frames, optimization is used with direct methods, while the geometric vision helps feature-based methods to calculate the camera's motion. Our proposed method belongs to the latter, but it differs from the two approaches mentioned above by directly exploiting the properties of catadioptric omni-directional cameras. Figure 1 shows the omni-directional camera that we use to collect omni-images and one omni-image, mounted on a simple robot. We also show the image as captured by

¹All authors are with the School of Information Science Technology of ShanghaiTech University, China <kuanghf, xuqw, longxl, soerensch>@shanghaitech.edu.cn

the smart phone and its unwrapped panorama image. From Figure 1(c), we can see that the catadioptric panorama image has the disadvantage of low resolution, as mentioned in [3], which will increase the difficulty for feature-based methods. Thus it is important to process these images with more robust algorithms. To meet this requirement, we combined the improved Fourier-Mellin invariant (iFMI) algorithm with our proposed method, since it proves to be more robust than SIFT. Our main contributions are summarized as:

- 1) proposing a novel relative pose estimation method based on geometric vision and fitting of pixel displacement values to sinusoidal functions;
- 2) exploiting the 2D frequency-based algorithm to estimate 3D pose of the omni-camera;
- 3) comparing our algorithm with commonly used epipolar geometry methods together with different features.

The rest of this paper is organized as follows: the theoretical foundation of our proposed method is introduced in Sec II, including motion model, proof and parameter estimation; then we explain out implementation detail in Sec III; the experiment and analysis is displayed in Sec IV; finally, we conclude our work in Sec V.

II. PROBLEM FORMULATION

A. Notations

- u, v -axis: image coordinates
- x, y, z -axis: camera coordinates
- I_o : an omni-image; $^k I_o$: superscript k means k^{th} frame
- I_p : a panorama image
- u_p, v_p : coordinates in a panorama image;
- u_{max}, v_{max} : width and height of the panorama image;
- $\Delta u(u_p)$: the translation (in pixel) in u -axis versus a column (u_p) between two panorama images;
- $\Delta v(u_p)$: the translation (in pixel) in v -axis versus a column (u_p) between two panorama images;
- R : 3D rotation of the camera between two panorama images;
- t : 3D translation of the camera between two panorama images;
- $P_{xy}(t)$: projection of t in x - y plane;
- $P_z(t)$: projection of t in z axis;
- $P_{xy}(R)$: projection of R around x and y axis;
- $P_z(R)$: projection of R around z axis;
- \hat{t}_{xy} : the angle between the axis of the x, y translation and x axis;
- λ : the unknown scale factor of the translation;
- γ : the known opening angle of a pixel. Suppose pixels are square, then $\gamma = \frac{2\pi}{u_{max}}$;
- r : the radius of the cylinder;
- H : the height of the cylinder; thus the height of corresponding panorama image;
- T : transformation matrix in 3D space.

B. Modeling of Pose Estimation

1) *Cylinder Camera Model*: As mentioned in [22], an image captured by a catadioptric omni-directional camera is usually unwrapped into a cylindric panorama image. Thus

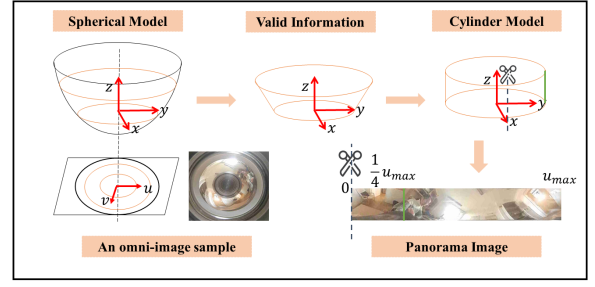


Fig. 2: Cylindric Camera Model

we model the pose estimation problem base on the cylindric camera model in this work. Figure 2 gives an intuitive description from spherical to cylinder model, which can be implemented by interpolation. The transformation between the cylinder and the panorama image can be described as

$$u = r \cdot \theta = r \cdot \arctan\left(\frac{y}{x}\right) \quad (1a)$$

$$v = \frac{H}{2} - z \quad (1b)$$

and

$$x = r \cdot \cos \theta = r \cdot \cos \frac{u}{r} \quad (2a)$$

$$y = r \cdot \sin \frac{u}{r} \quad (2b)$$

$$z = \frac{H}{2} - v \quad (2c)$$

Note that in the panorama image there will be a u associated with the positive x -axis of the camera (e.g. front of the robot) and another $u, \frac{1}{4}u_{max}$ pixels to the right, which is associated with the positive y -axis of the camera (e.g. left of the robot). The negative sides of the axes will be on the opposite side of the panorama image, so $\frac{1}{2}u_{max}$ pixels away. In order to simplify the formulation, this paper assumes that the x -axis will always located at $u = 0$ (grey dash line in Figure 2), and the y -axis is thus at $\frac{1}{4}u_{max}$ (green line in Figure 2).

2) *Resolution Consistency*: When unwrapping omni-images to panorama images with Eq. 1, 2, we cannot make sure that each pixel is square, i.e. there may be resolution inconsistency. In other words, the incident angle could be different with same pixels in u - and v -axis. Thus we use a calibration method described in [23] to find the ratio between angles per pixel in u - and v - direction. For the remainder of this paper we assume a perfectly calibrated cylinder model with square pixels, while the detailed discussion of such calibration is out of the scope of this paper.

3) *Motion Model*: We propose a sinusoidal function to describe the motion model of catadioptric omni-directional cameras as follows,:

$$y = B + A \sin(\omega x + \phi) \quad (3a)$$

$$\Delta v(u_p) = \lambda t_z + \gamma \|P_{xy}(R)\| \cdot \sin\left(\gamma u_p + \frac{P_{xy}(R)}{\|P_{xy}(R)\|}\right) \quad (3b)$$

$$\Delta u(u_p) = \gamma P_z(R) + \lambda \|P_{xy}(t)\| \cdot \sin(\gamma u_p + \hat{t}_{xy}) \quad (3c)$$

We model the motion of the camera using the sinusoid function Eq 3a with offset B , amplitude A and phase shift ϕ , while ω is just a calibration factor (field of view of a pixel). In order to recover the full six degree of freedom (DoF) motion of the camera we apply this function twice: once to the vertical motion $\Delta v(u_p)$ of the columns of the panorama image (Eq. 3b) and once to the horizontal motion $\Delta u(u_p)$ of the columns of the panorama image (Eq. 3c).

Then we analyze the motion model in four different cases to explain why we choose the sinusoidal function to represent the model:

- 1) **Translation along z-axis** (t_z): the shift along v -axis of each column u_p , which is the same for all u_p ;
- 2) **Rotation around x- and y- axis** (R_{xy}): (Roll and pitch, respectively, *using pitch=0 and some roll as example*) the closer u_p is to the (positive or negative) x -axis of the cylinder model, the smaller the shift along the v -axis; the closer u_p is to the (positive or negative) y -axis, the larger the shift along the v -axis;
- 3) **Rotation around z-axis** (R_z): (Yaw) the shift along the u -axis in each column u_p , which is the same for all u_p ;
- 4) **Translation along x- and y-axis** ($t_{x,y}$): (*using translation along x-axis as an example*) the closer u_p is to the (positive or negative) x -axis of the cylinder model, the smaller the shift along u -axis of the center in each column; the closer u_p is to the (positive or negative) y -axis, the larger the shift along the u -axis.

When the camera rotates around an arbitrary axis \vec{c} in the x - y plane, the shift along v of the columns u_p follows a certain pattern: the closer the column u_p is to \vec{c} , the smaller the shift is. Thus we use the phase shift $\frac{P_{xy}(R)}{\|P_{xy}(R)\|}$ to model the angle of \vec{c} to the x -axis in Eq 3b. The phase shift \hat{t}_{xy} in Eq 3c is derived following the same argument.

In the following we give a rigorous proof of the motion model. We have two images I_{p1} and I_{p2} and a transform (R, t) between them. Assume there is an arbitrary point ${}^2p_P = [u_{p2}, v_{p2}]$ in the second panorama image I_{p2} , its cylinder coordinate 2P is

$${}^2P = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} r \cos \frac{u}{r} \\ r \sin \frac{u}{r} \\ \frac{H}{2} - v \end{bmatrix} \quad (4)$$

Then we analyze the shifts of each row or column when the camera moves like the above four cases. Firstly, the 3D point 2P is transformed to ${}^1P = [x_1, y_1, z_1]^T$ with specified transformation matrix T ; secondly, we find the intersection ${}^1P' = [x'_1, y'_1, z'_1]^T$ between the vector O^1P and the cylinder $\{C : x^2 + y^2 = r^2\}$, which is then unwrapped into the point 1p_P in the panorama image I_{p1} ; finally, we calculated the shift between 1p_P and 2p_P in row and column direction, respectively.

- 1) **Translation along z-axis**: The transformation matrix T is:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & t_z \end{bmatrix} \quad (5a)$$

the transformed point 1P is

$${}^1P = \begin{bmatrix} r \cos \frac{u}{r} \\ r \sin \frac{u}{r} \\ \frac{H}{2} - v + t_z \end{bmatrix} \quad (5b)$$

the intersection ${}^1P'$ is

$${}^1P' = \begin{bmatrix} r \cos \frac{u}{r} \\ r \sin \frac{u}{r} \\ \frac{H}{2} - v + \lambda_z t_z \end{bmatrix} \quad (5c)$$

finally we get the shift Δv in column direction:

$$\Delta v = \left(\frac{H}{2} - z_2\right) - \left(\frac{H}{2} - z'_1\right) = \lambda_z t_z \quad (5d)$$

- 2) **Rotation around x- and y- axis**: (for example x - axis (roll))

The transformation matrix T is:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta_x & -\sin \theta_x & 0 \\ 0 & \sin \theta_x & \cos \theta_x & 0 \end{bmatrix} \quad (6a)$$

the transformed point 1P is

$${}^1P = \begin{bmatrix} r \cos \frac{u}{r} \\ r \sin \frac{u}{r} \cos \theta_x - \left(\frac{H}{2} - v\right) \sin \theta_x \\ r \sin \frac{u}{r} \sin \theta_x + \left(\frac{H}{2} - v\right) \cos \theta_x \end{bmatrix} \quad (6b)$$

the intersection ${}^1P'$ is

$${}^1P' = \begin{bmatrix} \frac{r}{\sqrt{1+k^2}} \\ \frac{r \sin \frac{u}{r} \sin \theta_x + \left(\frac{H}{2} - v\right) \cos \theta_x}{r \cos \frac{u}{r}} \\ \frac{r \cos \frac{u}{r}}{\sqrt{1+k^2}} \end{bmatrix} \quad (6c)$$

where $k = \frac{r \sin \frac{u}{r} \cos \theta_x - \left(\frac{H}{2} - v\right) \sin \theta_x}{r \cos \frac{u}{r}}$; finally we get the shift Δv in column direction:

$$\Delta v = \frac{r(r \sin \frac{u}{r} \sin \theta_x + \left(\frac{H}{2} - v\right) \cos \theta_x)}{\sqrt{(r \cos \frac{u}{r})^2 + (r \sin \frac{u}{r} \cos \theta_x - \left(\frac{H}{2} - v\right) \sin \theta_x)^2}} - \left(\frac{H}{2} - v\right) \quad (6d)$$

The denominator in Eq 6d can expand into $r^2 - r \sin \frac{u}{r} \left(\frac{H}{2} - v\right) \sin 2\theta_x + \left(\frac{H}{2} - v\right)^2 \sin^2 \theta_x$, which can be approximated to r^2 when θ_x is small. Under this condition, $\sin \theta_x \approx \theta_x$ and $\cos \theta_x \approx 1$. Thus the shift will be

$$\Delta v = \left(\frac{H}{2} - z_2\right) - \left(\frac{H}{2} - z'_1\right) = \theta_x R \sin \frac{u}{R} \quad (6e)$$

- 3) **Rotation around z-axis**:

The transformation matrix T is:

$$T = \begin{bmatrix} \cos \theta_z & -\sin \theta_z & 0 & 0 \\ \sin \theta_z & \cos \theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (7a)$$

the transformed point 1P is

$${}^1P = \begin{bmatrix} r \cos\left(\frac{u}{r} + \theta_z\right) \\ r \sin\left(\frac{u}{r} + \theta_z\right) \\ \frac{H}{2} - v \end{bmatrix} \quad (7b)$$

the intersection ${}^1P'$ remains

$${}^1P' = {}^1P = \begin{bmatrix} r \cos(\frac{u}{r} + \theta_z) \\ r \sin(\frac{u}{r} + \theta_z) \\ \frac{H}{2} - v \end{bmatrix} \quad (7c)$$

finally we get the shift Δu in row direction:

$$\Delta u = r \arctan \frac{y_2}{x_2} - r \arctan \frac{y'_1}{x'_1} = -r\theta_z \quad (7d)$$

- 4) **Translation along x- and y-axis:** (for example along x-axis)

The transformation matrix T is:

$$T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (8a)$$

the transformed point 1P is

$${}^1P = \begin{bmatrix} r \cos \frac{u}{r} + \lambda t_x \\ r \sin \frac{u}{r} \\ \frac{H}{2} - v \end{bmatrix} \quad (8b)$$

the intersection ${}^1P'$ is

$${}^1P' = \begin{bmatrix} \frac{r}{\sqrt{1+k^2}} \\ \frac{kr}{\sqrt{1+k^2}} \\ \frac{\frac{H}{2}-v}{r \cos \frac{u}{r} + \lambda t_x} \frac{r}{\sqrt{1+k^2}} \end{bmatrix} \quad (8c)$$

where $k = \frac{r \sin \frac{u}{r}}{r \cos \frac{u}{r} + \lambda t_x}$; finally we get the shift Δu in row direction:

$$\Delta u = r \arctan \frac{y_2}{x_2} - r \arctan \frac{y'_1}{x'_1} \quad (8d)$$

$$= r \arctan \frac{\lambda t_x \sin \frac{u}{r}}{\lambda t_x \cos \frac{u}{r} + r} \quad (8e)$$

In Eq 8e, since $r \gg \lambda t_x \cos \frac{u}{r}$, then $\lambda t_x \cos \frac{u}{r} + r$ approximately equals to r and $\frac{\lambda t_x \sin \frac{u}{r}}{\lambda t_x \cos \frac{u}{r} + r}$ is small enough to take the first-order Taylor approximation. Thus, the shift of each row can be described as:

$$\Delta u \approx r \frac{\lambda t_x \sin \frac{u}{r}}{r} = \lambda t_x \sin \frac{u}{r} \quad (8f)$$

Based on the proof, we can tell that Eq 3c and Eq 3b is correct when the translation is small enough in x-y plane or the rotation component along axis in x-y plane is small enough.

C. Curve Fitting

There are some unknown parameters $\Phi = \{A, \phi, B\}$ in Eq. 3 that we estimate using curve fitting. We solve it through modeling it as a nonlinear least-squared problem and the parameters are estimated by using a nonlinear optimization algorithm. Then the rotation R and translation t can be calculated from Φ .

To find the corresponding parameters Φ_v, Φ_u in Eq. 3b and Eq. 3c we build the following two objective functions:

$$C_v(u_p, \Phi_v) = \Delta v(u_p; \Phi_v) - y_v \quad (9a)$$

$$C_u(u_p, \Phi_u) = \Delta u(u_p; \Phi_u) - y_u \quad (9b)$$

$$\min_{\Phi_v} L_v(u_p; \Phi_v) = \min_{\Phi_v} \frac{1}{2} \|C_v(u_p, \Phi_v)\|_2^2 \quad (9c)$$

$$\min_{\Phi_u} L_u(u_p; \Phi_u) = \min_{\Phi_u} \frac{1}{2} \|C_u(u_p, \Phi_u)\|_2^2 \quad (9d)$$

where C_v and C_u are cost functions and y_v and y_u are the measured shifts in v - and u - direction by using the iFMI algorithm [19]. L_v and L_u are loss functions in the standard least-squared form. Afterwards, the nonlinear optimization Levenberg-Marquardt algorithm [24] is used to minimize the objective functions, which could also be replaced with other optimization methods.

$$L_v(u_p; \Phi_v, \delta) = \delta \cdot \left(\sqrt{1 + \left(\frac{C_v(u_p, \Phi_v)}{\delta} \right)^2} - 1 \right) \quad (10a)$$

$$L_u(u_p; \Phi_u, \delta) = \delta \cdot \left(\sqrt{1 + \left(\frac{C_u(u_p, \Phi_u)}{\delta} \right)^2} - 1 \right) \quad (10b)$$

We use two methods to handle the outliers problem. First we employ a median filter of data to remove some obvious outliers. Then we use the Huber loss function to reduce the influence of outliers, which was proven to be less sensitive to outliers in data than the L2 loss [25]. Because the Pseudo-Huber loss function [26] combines the best properties of L2 loss and Huber loss by being strongly convex when close to the minimum and less steep for outliers, we use it to replace the standard least-squared loss form with Eq. 10a and Eq. 10b.

III. IMPLEMENTATION

The implementation of our proposed method is described in Algorithm 1, where we have $W = 1100, L = H = 110, d = 20$.

We use a square window to slide along the u -direction on the panorama images I_{p1} and I_{p2} . For each window pair, we use the iFMI method to find the 2D motion. Then we get the set of shifts Δu and Δv versus column index $u_p =$

Algorithm 1 Proposed pose estimation for omni-cameras

- 1: **Input:** Omni images I_{o1}, I_{o2} ;
Sliding window size $L \times L$ and step d
 - 2: Obtain panorama images I_{p1}, I_{p2} of size $W \times H$
by cartesian-to-polar transformation
 - 3: **for** $L + k \times d \leq W, k \in \mathbb{N}$ **do**
 - 4: Compute the scaling, rotation and translation t_{uv}
for k^{th} window between I_{p1} and I_{p2} by iFMI[17]
 - 5: Push t_{uv} to motion set \mathbb{M}
 - 6: **end for**
 - 7: Estimate parameter Φ_v and Φ_u by optimization on \mathbb{M} (Eq. 9)
 - 8: Calculate transformation T from Φ_v, Φ_u (Eq. 3)
 - 9: **Output:** T
-

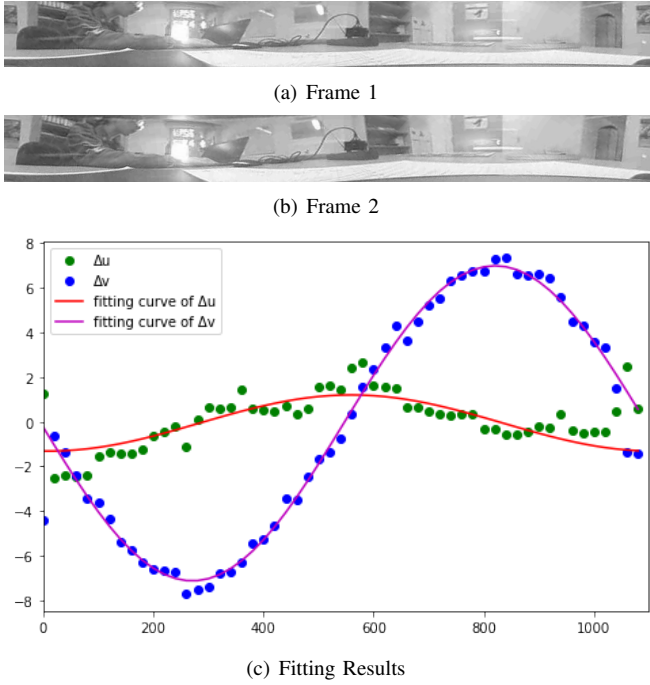


Fig. 3: Δu and Δv is shift of u and v direction, respectively, which are measured using iFMI. The results of function fitting by non-linear least squared method through the Δu and Δv values are shown. The motion corresponds to a roll (column 0 is the x -axis).

$\frac{1}{2}L + k \times d$. Afterwards, we fit the values with the sinusoidal function to estimate parameter Φ as line 7 of Algorithm 1 describes. $\delta = 2$ of Eq 10 is selected in our implementation. Figure 3(c) displays an example of curve fitting between two frames. The optimization algorithm fits the sinusoids effectively.

In Figure 3(c) we see maxima in the Δv curve at around $\frac{1}{4}u_{max}$ and $\frac{3}{4}u_{max}$. Following the definition from above, that the x -axis is at $u = 0$, this means there is a big motion in the image where the y -axis is. It thus follows that there has been a rotation around the x -axis of the camera (roll).

IV. EXPERIMENTS

In this section we compare our method with feature-based algorithms using experiments on three different datasets. We use images from indoor (office) and outdoor (street) scenes as well as another dataset with ambiguous features (grass), which poses big challenges to feature-matching algorithms (Samples are shown in Figure 4). All the images are captured by the phone (Oneplus 5) covered with a low-cost omni-lens (Kogeto Dot Lens), as shown in Figure 1(a). In addition, all the computations are conducted on a PC with an Intel Core i7-6700 CPU and 16 GB memory.

In this experiment, we choose ORB and AKAZE feature detectors to implement the feature-based algorithm, since ORB is one of the fastest detectors and AKAZE is designed for detecting features in non-linear space according to [27]. Besides, the STEWENIUS five-point algorithm is exploited

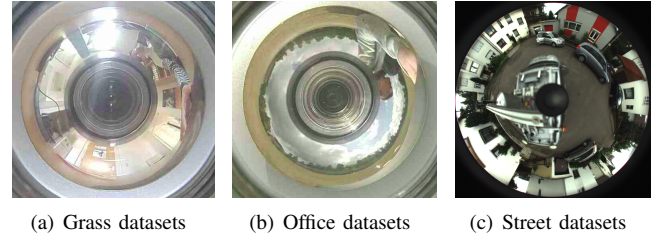


Fig. 4: Experimental Image Samples

to estimate the relative pose, which is widely used and performs robustly.

To simplify the comparison, the experiment is done with controlled variables, that is, we only rotate the camera around one axis (x -, y - or z -axis) at one single test. Experiments for the translation are omitted in this work for two reasons: For estimating the correct translation, the sinusoid fitting model assumes an equal distance to the camera of all points. This restriction is easily broken and thus the translation estimation is error prone. The rotation doesn't require this assumption. Secondly, the translation estimate is anyways up to an unknown scale factor, which adds another difficulty. We use the IMU of the smartphone as ground truth. In the following, we analyze the results in both qualitative and quantitative ways.

A. Results Analysis

Figure 5 demonstrates an intuitive performance of the pose estimation in the three datasets. We find that our method achieves a better performance than feature-based methods in general. Thanks to the robust iFMI algorithm, our method also works well on the grass dataset, while the feature-based methods suffer from describing the ambiguous features. Especially, the ORB-based algorithm fails to detect roll on the grass datasets. Both the proposed algorithm and feature-based methods show a good performance on office datasets, where the latter gets help from rich features in this scenario, e.g. chairs, books and so on. Moreover, the proposed method also outperforms the two feature-based ones on the office dataset. Lastly, the street dataset shows that our method also performs well regarding the accuracy and robustness in the general outdoor scenario.

Table I provides a rigorous comparison among three methods, where the root mean square error (RMSE) is used as the error metrics. This table shows that the total average error ϵ of our proposed method is less than half of the AKAZE-based algorithm and about one-quarter of the ORB-based algorithm. It also shows that the AKAZE-based method outperforms our method with a slight advantage in the yaw test. But we suspect, that the ground truth yaw (from the smartphone IMU) might not be reliable even for the short interval between the frames and thus the yaw-experiments shouldn't be trusted too much. The run-time of the AKAZE-based algorithm is more than that of our proposed and ORB-based methods, as the last column of Table I shows.

In summary, the our method is the most robust among

	grass			office			street			average error			$\epsilon[rad]$	time[s]
	roll	pitch	yaw	roll	pitch	yaw	roll	pitch	yaw	roll	pitch	yaw		
ORB	0.196	0.683	0.239	0.225	0.163	0.191	0.313	0.339	0.328	0.245	0.395	0.253	0.300	0.11
AKAZE	0.130	0.414	0.113	0.171	0.110	0.073	0.164	0.312	0.106	0.155	0.279	0.097	0.177	0.75
Ours	0.088	0.123	0.129	0.054	0.047	0.021	0.027	0.032	0.143	0.056	0.067	0.098	0.074	0.40

TABLE I: RMSE and average run-time of our method and feature-based approaches.

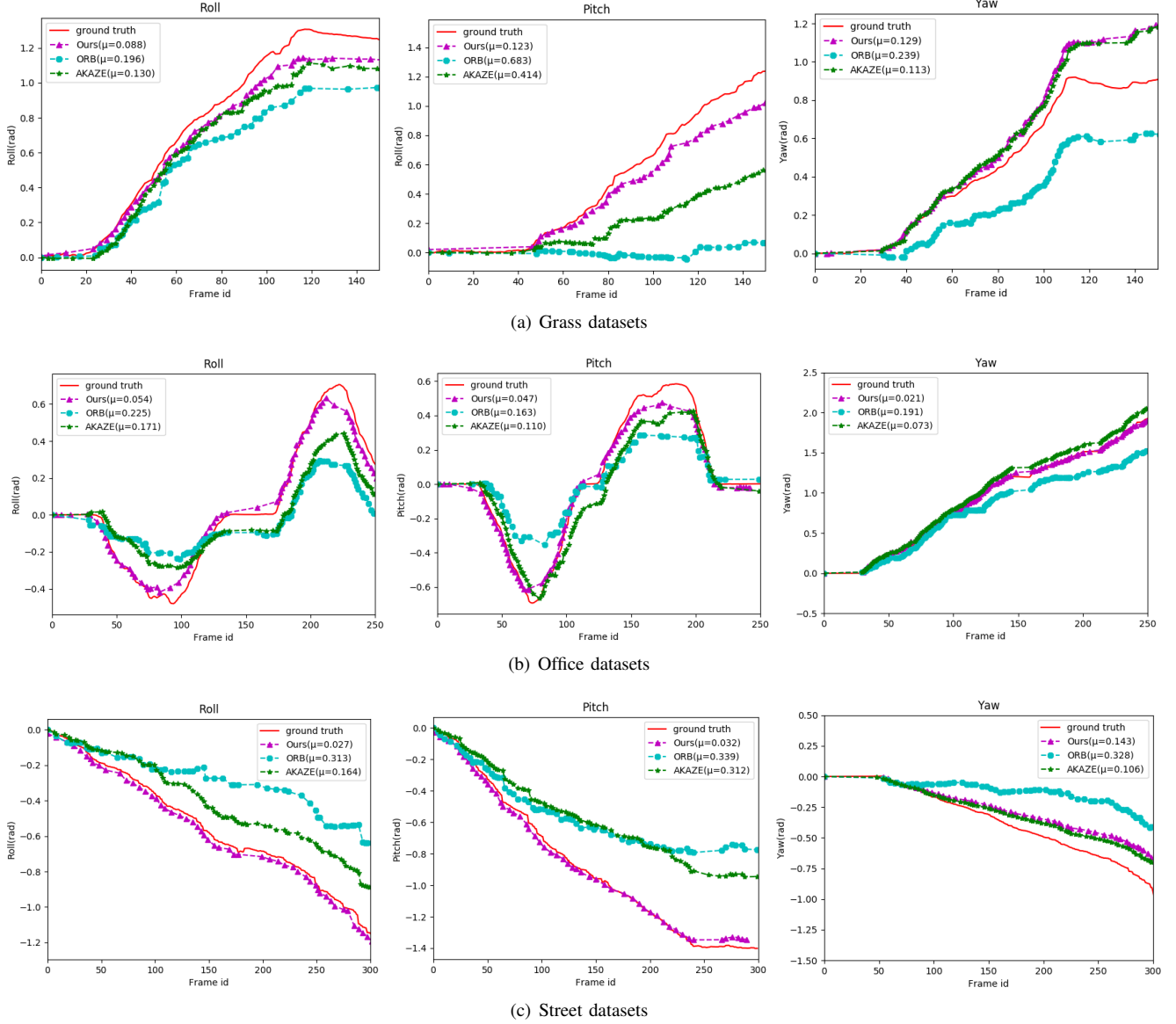


Fig. 5: The rotation estimation experiments with different datasets. The μ in the legend represents RMSE for each method.

the three algorithms in the different application scenarios and performing especially well in environments which are feature-deprived or have repetitive features, such as grass. Even though our method is not the fastest one, it is cost-effective when considering the balance between accuracy and run-time.

V. CONCLUSION

In this paper we proposed a novel pose estimation method by fitting sinusoid functions of pixel displacements in panoramic images of omni-directional cameras. We calculated the average pixel displacements of columns in the image using the iFMI algorithm. Experimental results for the 3D rotation part show that our method outperforms feature-

based approaches, which we attribute to the facts that we use a spectral method and then fit many results to (sinusoid) functions, which thus takes care of outliers of the spectral method. The iFMI method can only estimate 2D transforms of pin-hole images, while our approach can, in principal, estimate the full 6 DoF 3D camera pose (up to scale factor) from panoramic images.

From the experiment results, we can see that the accuracy of our proposed method is about two to four times better than the other feature-based approaches for roll and pitch. The speed of our method is almost twice as fast than the accurate AKAZE-based algorithm. Thus our method is the most cost-effective one among the three methods.

We envision our algorithm to be used more for fast visual-odometry rather than visual SLAM with big displacements between frames. In this case we estimate that also the translation part of our sinusoid fitting model will perform reasonably well. Soon we will do experiments also regarding the translation part of our algorithm, and use a tracking system to generate more precise ground truth data. Furthermore, we will attempt to incorporate also the scaling and rotation of the sliding window in our model, so we can use all results that iFMI is giving us, to maybe get even better and more robust results. We then also plan to investigate other means of generating the pixel displacements per column (e.g. optical flow, 1D spectral methods). Finally, we aim to integrate our method into a full omni-visual odometry and SLAM framework.

REFERENCES

- [1] A. A. Argyros, K. E. Bekris, S. C. Orphanoudakis, and L. E. Kavraki, "Robot homing by exploiting panoramic vision," *Autonomous Robots*, vol. 19, no. 1, pp. 7–25, 2005.
- [2] T. Lemaire and S. Lacroix, "Slam with panoramic vision," *Journal of Field Robotics*, vol. 24, no. 1-2, pp. 91–111, 2007.
- [3] R. Benosman, S. Kang, and O. Faugeras, *Panoramic vision*. Springer-Verlag New York, Berlin, Heidelberg, 2000.
- [4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 6, pp. 1052–1067, 2007.
- [5] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.
- [6] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2018.
- [7] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.
- [8] A. Handa, M. Bloesch, V. Pătrăucean, S. Stent, J. McCormac, and A. Davison, "gvnn: Neural network library for geometric computer vision," in *European Conference on Computer Vision*. Springer, 2016, pp. 67–82.
- [9] G. Costante, M. Mancini, P. Valigi, and T. A. Ciarfuglia, "Exploring representation learning with cnns for frame-to-frame ego-motion estimation," *IEEE robotics and automation letters*, vol. 1, no. 1, pp. 18–25, 2016.
- [10] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtm: Dense tracking and mapping in real-time," in *2011 international conference on computer vision*. IEEE, 2011, pp. 2320–2327.
- [11] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," in *International workshop on vision algorithms*. Springer, 1999, pp. 298–372.
- [12] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3607–3613.
- [13] M. Pfingsthorn and A. Birk, "Generalized graph slam: Solving local and global ambiguities through multimodal and hyperedge constraints," *International Journal of Robotics Research (IJRR)*, vol. 35, 2016.
- [14] R. I. Hartley, "In defence of the 8-point algorithm," in *Proceedings of IEEE international conference on computer vision*. IEEE, 1995, pp. 1064–1070.
- [15] H. Stewénius, D. Nistér, F. Kahl, and F. Schaffalitzky, "A minimal solution for relative pose with unknown focal length," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005, pp. 789–794.
- [16] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi, "Revisiting the pnp problem: A fast, general and optimal solution," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2344–2351.
- [17] H. Bülow and A. Birk, "Fast and robust photomapping with an unmanned aerial vehicle (uav)," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 3368–3373.
- [18] S. Schwertfeger, H. Bülow, and A. Birk, "On the effects of sampling resolution in improved fourier mellin based registration for underwater mapping," in *7th Symposium on Intelligent Autonomous Vehicles (IAV)*, IFAC, IFAC, 2010.
- [19] A. Birk, B. Wiggerich, H. Bülow, M. Pfingsthorn, and S. Schwertfeger, "Safety, security, and rescue missions with an unmanned aerial vehicle (uav): Aerial mosaicking and autonomous flight at the 2009 european land robots trials (elrob) and the 2010 response robot evaluation exercises (rree)," *Journal of Intelligent and Robotic Systems*, vol. 64, no. 1, pp. 57–76, 2011.
- [20] S. Schwertfeger, A. Birk, and H. Bülow, "Using ifmi spectral registration for video stabilization and motion detection by an unmanned aerial vehicle (uav)," in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, IEEE Press. IEEE Press, 2011, pp. 1–6.
- [21] H. Bülow and A. Birk, "Spectral 6-dof registration of noisy 3d range data with partial overlap," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 35, pp. 954–969, 2013.
- [22] D. Scaramuzza, "Omnidirectional camera," *Computer Vision: A Reference Guide*, pp. 552–560, 2014.
- [23] T. L. Conroy and J. B. Moore, "Resolution invariant surfaces for panoramic vision systems," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 1. IEEE, 1999, pp. 392–397.
- [24] D. W. Marquardt, "An algorithm for least-squares estimation of non-linear parameters," *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.
- [25] P. J. Huber, "Robust estimation of a location parameter," in *Breakthroughs in statistics*. Springer, 1992, pp. 492–518.
- [26] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud, "Deterministic edge-preserving regularization in computed imaging," *IEEE Transactions on image processing*, vol. 6, no. 2, pp. 298–311, 1997.
- [27] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*. IEEE, 2018, pp. 1–10.