Variational Fisheye Stereo

Menandro Roxas[1] and Takeshi Oishi[2]

***Abstract*—Dense 3D maps from wide-angle cameras is beneficial to robotics applications such as navigation and autonomous driving. In this work, we propose a real-time dense 3D mapping method for fisheye cameras without explicit rectification and undistortion. We extend the conventional variational stereo method by constraining the correspondence search along the epipolar curve using a trajectory field induced by camera motion. We also propose a fast way of generating the trajectory field without increasing the processing time compared to conventional rectified methods. With our implementation, we were able to achieve real-time processing using modern GPUs. Our results show the advantages of our non-rectified dense mapping approach compared to rectified variational methods and non-rectified discrete stereo matching methods.**

***Index Terms*—Omnidirectional Vision, Mapping**

I. INTRODUCTION

WIDE-ANGLE (fisheye) cameras have seen significant usage in robotics applications. Because of the wider field-of-view (FOV) compared to the pinhole camera model, fisheye cameras pack more information in the same sensor area which are advantageous especially for object detection, visual odometry, and 3D reconstruction. In applications that require 3D mapping, using fisheye cameras have several advantages especially for navigation and autonomous driving. For example, the wide FOV allows for simultaneous visualization and observation of objects in multiple directions.

Several methods have addressed the 3D mapping problem for fisheye cameras. The most common approach performs rectification of the images to perspective projection which essentially removes the main advantage of such cameras - wide FOV. Moreover, information closer to the edge of the image are highly distorted while objects close the center are highly compressed, not to mention adding unnecessary degradation of image quality due to spatial sampling. Other rectification methods that retain the fisheye's wide FOV involve reprojection on a sphere, which suffers from similar degradation especially around the poles.

We address these issues by directly processing the distorted images without rectification and undistortion. We embed our method in a variational framework, which inherently produces smooth dense maps in contrast to discrete stereo matching methods. However, directly applying existing variational stereo methods on unrectified and distorted fisheye images is not straightforward. The main obstacle is that solving the image gradient that guides the linearized correpondence search needs to be constrained along the epipolar curves instead of lines. This requires finding the function of the curve in every optimization step as determined by the camera projection and distortion model. Furthermore, adapting the gradient calculation to be restricted along the curve is also not simple since the gradient can only be estimated (in discretized form) along the tangential of the curve and this direction is valid only if the distance between corresponding pixels is very small.

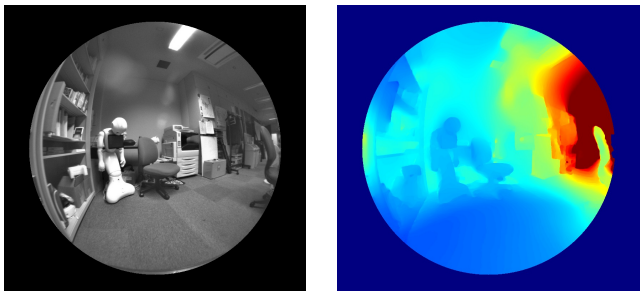

Fig. 1. Non-rectified variational stereo method result on a fisheye stereo camera.

Instead of solving for the epipolar curve function, we propose to use a trajectory field which represents the direction of the epipolar curve for every pixel. We also propose a fast way of generating the trajectory field that does not require additional processing time compared to conventional variational methods.

One advantage of using a trajectory field image is that it allows us to use simple linear interpolation to approximate the tangential of the epipolar curve which is faster than performing a direct calculation. Furthermore, since our method is founded on a variational framework, it produces smooth and dense depth maps and has high subpixel accuracy.

Our results show additional accurate measurements when compared to conventional rectified methods, and more accurate and dense estimation compared to non-rectified discrete methods. Finally, with our implementation, we were able to achieve real-time processing on a consumer fisheye stereo camera system and modern GPUs.

II. RELATED WORK

Dense stereo estimation in perspective projection consists of a one-dimensional correspondence search along the epipolar lines. In a variational framework, the search is akin to linearizing the brightness constancy constraint along the epipolar lines. In [1], a differential vector field induced by arbitrary camera motion was used for linearization. However, their method, as with other variational stereo methods in perspective projection such as [2], requires undistortion and/or rectification

Manuscript received: September, 10, 2019; Revised November 30, 2019; Accepted January, 2, 2020.

This paper was recommended for publication by Eric Marchand upon evaluation of the Associate Editor and Reviewers' comments. This work was supported in part by the social corporate program (Base Technologies for Future Robots) sponsored by NIDEC corporation and in part by JSPS KAKENHI under Grants JP16747698 and JP17923471.

Both authors are with The University of Tokyo, Tokyo, Japan [1]roxas, [2]oishi@cvl.iis.u-tokyo.ac.jp

Digital Object Identifier (DOI): see top of this page.

(in case of binocular stereo) to be applicable for fisheye cameras [3].

Instead of perspective rectification, some methods reproject the images to spherical or equirectangular projection [4] [5] [6] [7]. However, this approach suffers greatly from highly distorted images along the poles which makes estimation less accurate especially when using the variational framework. Similar to our image linearization approach, [6] generates differential vectors induced by variations on a 2-sphere in which the variational stereo method was applied. However, their graph-based formulation is a solution to the self-induced problem arising from reprojecting the image on a spherical surface. In contrast, our method does not require reprojection on a 2-sphere and works directly on the distorted images without undistortion, reprojection or rectification. We do this by evaluating the variations directly from the epipolar curve.

Other methods also directly work on the distorted fisheye images. In [8], the unified camera model [9] was used to determine the path of the search space, which are incrementally shifted (akin to differential vectors) from a reference pixel to the maximum disparity. At each point, the projection function is re-evaluated which the authors claim was costly compared to linear search. However, their mapping method, while real-time, only produces semi-dense depth maps. In [10], a similar parameterization of the epipolar curve was done, but only applied on window-based stereo matching. Other methods adapts linear matching algorithms to omni-directional cameras such as semi-global matching [11], plane-sweeping [12] and a variant called sphere-sweeping [13]. Sparse methods were also adapted to handle fisheye distortion such as [14] among others. Since our method is based on a variational framework, it produces smoother and denser disparity map and has an inherent subpixel accuracy compared to direct matching and sparse methods.

III. VARIATIONAL FISHEYE STEREO

In this section, we will first introduce the problem of image linearization in fisheye camera systems in Sec. III-A. We will then propose our trajectory field generation method in Sec. III-B. Finally, we will show our warping technique in III-C.

A. Image Linearization Problem in Fisheye Cameras

Classical variational stereo methods consist of finding a dense disparity map between a pair of images that minimizes a convex energy function which includes a data term and a smoothness or regularizer term. This energy is often expressed as:

$$E(u) = E_{data}(u) + E_{smooth}(u) \tag{1}$$

where $u \in \mathbb{R}$ is the one-dimensional disparity that indicates the Euclidean distance (in pixels) between two corresponding points in an image pair. For fisheye cameras, these correspondences are constrained along the epipolar curve, $\gamma : \mathbb{R} \rightarrow \mathbb{R}^2$ and finding them constitutes a one-dimensional search [8][11][12] along γ. In our case, we solve the correspondences in a variational framework.

In general, the data term penalizes the difference in value (e.g. brightness, intensity gradient, non-local transforms [15], etc.) between the corresponding pixels through a residual function ρ. Given two images, I_0 and I_1, with known camera transformation (non-zero translation) and intrinsic parameters, we can express the set of corresponding pixels along the epipolar curve as $\{(\mathbf{x}, \pi(\exp(\hat{\xi}_1) \cdot \mathbf{X}(\mathbf{x}, u)) : \mathbf{x} \in \mathbb{R}^2\}$, where $\pi : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ is the projection of the 3D point $\mathbf{X}$ on the image plane $\Omega_1 \in \mathbb{R}^2$ of I_1. The camera pose $\xi_1 \in \mathbb{R}^6$ is the pose of I_1 relative to I_0 such that the *twist* $\hat{\xi}_1 \in \mathbf{se}(3)$ represents the 4x4 matrix parameterized by the exponential coordinates ξ_1. The residual is then defined as:

$$\rho(\mathbf{x}, u) = I_1\left(\pi\left(\exp(\hat{\xi}_1) \cdot \mathbf{X}(\mathbf{x}, u)\right)\right) - I_0\left(\mathbf{x}\right) \tag{2}$$

Assuming that I_0 and I_1 is linear along the curve, we can approximate Eq. (2) with $\bar{\rho}$ using the first-order Taylor expansion. Using a simplified notation $I_1\left(\pi\left(\exp(\hat{\xi}_1) \cdot \mathbf{X}(\mathbf{x}, u)\right)\right) = I_1(\mathbf{x}, u)$, the residual can be expressed as:

$$\bar{\rho}(\mathbf{x}, u) = I_1(\mathbf{x}, u_\omega) + (u - u_\omega)\frac{d}{du} I_1(\mathbf{x}, u)\Big|_{u_\omega} - I_0(\mathbf{x}) \tag{3}$$

where u_ω is a known disparity (solved from a prior iteration). Minimizing Eq. (3) results in the incremental disparity which we will designate from here on as $\delta u_\omega = (u - u_\omega)$.

Since the linearity assumption for I is only valid for a small disparity, we embed Eq. (3) in an iterative warping framework [16]. That is, for every warping iteration ω, we update $u_{\omega+1} = u_\omega + \delta u_\omega$.

Solving Eq. (3) also requires the evaluation of the derivative $\frac{d}{du} I_1(\mathbf{x}, u)$ which can be expressed as the dot product of the gradient of $I_1(\mathbf{x}, u)$ and a differential vector at $\mathbf{x}$:

$$\frac{d}{du} I_1(\mathbf{x}, u) = \nabla I_1(\mathbf{x}, u) \cdot \underbrace{\frac{d}{du}\pi\left(\exp(\hat{\xi}_1) \cdot \mathbf{X}(\mathbf{x}, u)\right)}_{\text{differential vector}} \tag{4}$$

However, in practice, we directly solve for the variations of I_1 along the epipolar curve. In discrete form, we can express this as:

$$\frac{d}{du} I_1(\mathbf{x}, u) = I_1(\mathbf{x} + \gamma') - I_1(\mathbf{x}) \tag{5}$$

where the differential vector is simplified as γ'. For small disparities, this differential vector is equivalent to the tangential vector of the epipolar curve $\gamma' = \nabla\gamma$.

This formulation for minimizing the residual raises two issues when used in a fisheye camera system.

- First, the warping technique requires a re-evaluation of γ at every iteration to find the tangential vectors $\nabla\gamma$ at u_ω. While this process can be performed using an unprojection-projection step such as in [8], we argue that this is unnecessarily time consuming and tedious depending on the camera model used.
- Second, even if we assume that the image is perfectly linear along the epipolar curve, ∇I will only be valid along the direction of the tangential vectors. In a perspective projection, this is not a problem since the tangential vectors indicates the exact direction of the epipolar lines. In our case, the gradient will need to be evaluated exactly along the curve.

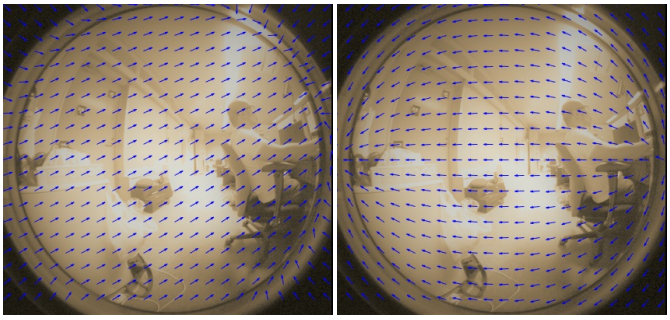

Fig. 2. Calibration (left) and trajectory (right) field for a binocular fisheye stereo.

In the following sections, we will elaborate on our approach to address these two issues.

B. Trajectory Field Representation for Epipolar Curves

As pointed out in the previous section, one way to estimate the differential vectors is to solve for the tangent of the epipolar curve at the exact point in the image determined by u_ω. This is necessary because the disparity u_ω does not point exactly to a pixel (non-integer value). However, since the function of the curve is already fixed at known points, i.e. pixels, why not solve for it once and then interpolate the values in between?

With that said, we propose to generate a trajectory field image that represents the tangential vectors γ' at every pixel $\mathbf{x}$. As a result, γ' at the next iteration step can be simply solved using bicubic interpolation.

First, instead of solving for the parameterized curve function for every pixel as in [17], we programmatically generate the trajectory field. We first assume a known transformation ξ_1 between two camera positions with non-zero translation ($|t| \neq 0$) and known projection π. Note, however, that our method is not restricted on any type of camera model [9] [18] [19] and is adaptable as long as the projection function π is defined.

Using π, we project a surface of arbitrary depth onto the two cameras: $\mathbf{x}_0 = \pi(\mathbf{X})$, $\mathbf{x}_1 = \pi(\exp(\hat{\xi}_1) \cdot \mathbf{X})$ which gives us the exact correspondence $\mathbf{w}(\mathbf{x}_0, \mathbf{x}_1) = \mathbf{x}_1 - \mathbf{x}_0$. Note that in a perspective projection, this mapping or the optical flow already signifies the slope of the epipolar lines. Assuming pre-rotated images, i.e. R=identity, the direction of the optical flow, $\frac{\mathbf{w}}{|\mathbf{w}|}$, will be dependent only on the direction of the camera translation t and independent of its magnitude $|t|$ and the surface depth $|\mathbf{X}|$. However, for fisheye projection, $\frac{\mathbf{w}}{|\mathbf{w}|}$ is still also affected by the camera distortion.

To handle the distortion, we can represent the optical flow as the sum of the tangential vectors along the path of the epipolar curve between the two corresponding points. Let the parameterization variable for γ be $s = [0, 1]$. In continuous form, we can express $\mathbf{w}(\mathbf{x}_0, \mathbf{x}_1)$ as:

$$\mathbf{w}(\mathbf{x}_0, \mathbf{x}_1) = \int_0^c \gamma'(s) ds \Big|_{c=1} \tag{6}$$

By scaling the camera translation such that $|t| \rightarrow 0$, the projected surface produces an optical flow field with very small magnitudes. In this case, the left hand side of (6) approaches $\mathbf{0}$. It follows that the right hand side becomes:

$$\lim_{c \to 0} \int_0^c \gamma'(s) ds = \gamma'(0) \tag{7}$$

which finally allows us to approximate $\gamma'(0) \approx \frac{\mathbf{w}}{|\mathbf{w}|}$. In short, $\frac{\mathbf{w}}{|\mathbf{w}|}$ gives us the normalized trajectory field. An example trajectory field image generated for a binocular stereo system is shown in Figure 2.

C. Warping Technique

The iterative warping framework requires the determination of $I_1(\mathbf{x}, u_\omega)$ in Eq: (3) with the given u_ω. The direct way of solving this is to find the 2D coordinate $\pi\left(\exp(\hat{\xi}_1) \cdot \mathbf{X}(\mathbf{x}, u_\omega)\right)$ which requires unprojection to find $\mathbf{X}(\mathbf{x}, u_\omega)$ and then reprojecting $\mathbf{X}$ to I_1 using π.

As an alternative, we can instead use the trajectory field to find the warping vector or the optical flow, $\mathbf{w}_\omega$. In this case, we can find the warped I_1 using $I_1(\mathbf{x}, u_\omega) = I_1(\mathbf{x} + \mathbf{w}_\omega)$. To do this, we need to understand how the trajectory field relates to the optical flow.

The trajectory field discretizes the epipolar curve by assigning finite vector values for every pixel. We can think of this approach as decomposing the epipolar curve as a piecewise linear function (see Figure 3) which allows us to express the disparity u as:

$$u = \sum_{\omega=0}^{N} \delta u_\omega \tag{8}$$

where N is the total number of warping iterations.

Clearly, we can better approximate the epipolar curve by making the incremental δu_ω small. We can do this by setting a magnitude limit such that $\delta u_\omega = \min(\delta u_\omega, \delta u^{max})$. By assigning a limiting value δu^{max}, we prevent missing the trajectory of the correct epipolar curve (see Figure 3). This approach consequently solves the problem of constraining the image linearization along the curve and allows us to continue using the discrete derivative for I_1 in Eq. (5).

The warping vector $\mathbf{w}_\omega$ can now be defined as the sum of the vectors whose magnitudes are equal to the incremental disparities and directions as the tangent of the epipolar curve. We can express this as:

$$\mathbf{w} = \sum_{\omega=0}^{N} \mathbf{w}_\omega = \sum_{\omega=0}^{N} \delta u_\omega \gamma'_\omega \tag{9}$$

IV. IMPLEMENTATION

In this section, we discuss our implementation choices to achieve accurate results and real-time processing, which includes image pre-processing, large displacement handling and our selected optimization parameters and hardware considerations.

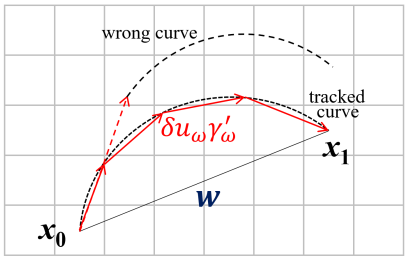

Fig. 3. Epipolar curve as a piecewise linear function. Large incremental δu_ω results in wrong tracked curve.

A. Anisotropic TGV-L1 Optimization

Our proposed algorithm can be applied on any regularized variational stereo method that uses the image linearization step described in Sec. III-A such as [1][2] and [20] among others. In this work, we followed the anisotropic tensor-guided total generalized variation (TGV) regularizer with L1 data penalty term described in [2]. We chose this method because it produces smooth surfaces while maintaining sharp object boundaries and can be implemented in real-time. The TGV-L1 energy term is summarized as:

$$E(u) = \lambda \int_\Omega |\rho(\mathbf{x}, u)| \, d^2\mathbf{x} + \alpha_0 \int_\Omega |\nabla v| \, d^2\mathbf{x} + \alpha_1 \int_\Omega \left| T^{\frac{1}{2}} \nabla u - v \right| d^2\mathbf{x} \quad (10)$$

where $\mathrm{T}^{\frac{1}{2}}$ is an anisotropic diffusion tensor. Eq. (10) allows the disparity u to be smooth by imposing a small variation (∇v) through the relaxation variable v while maintaining the natural object boundaries described by the image gradients and guided by the diffusion tensor.

We can minimize Eq. (10) using primal-dual algorithm, which consists of a gradient-ascent on the dual variables p and q, followed by a gradient-descent and over-relaxation refinement step on the primal variables u and $v : \mathbb{R}^2$. The dual variables p and q compose the convex sets P and Q, respectively, such that:

$$\begin{aligned} P &= \{p \in \mathbb{R}^2 : |p|_\infty \leq 1\} \\ Q &= \{q \in \mathbb{R}^4 : |q|_\infty \leq 1\} \end{aligned} \quad (11)$$

The primal-dual algorithm can be summarized as:

$$\begin{cases} p_{k+1} = \mathcal{P}\left(p_k + \sigma_p \alpha_1 (\mathrm{T}^{\frac{1}{2}} \nabla \bar{u}_k - \bar{v}_k)\right) \\ q_{k+1} = \mathcal{P}\left(q_k + \sigma_q \alpha_0 (\nabla \bar{v}_k)\right) \\ u_{k+1} = (I + \tau_u \partial G)^{-1}(u_n + \tau_u div(\mathrm{T}^{\frac{1}{2}} p_{k+1})) \\ v_{k+1} = v_k + \tau_v (div q_{k+1} + p_{k+1}) \\ \bar{u}_{k+1} = u_{k+1} + \theta(u_{k+1} - \bar{u}_k) \\ \bar{v}_{k+1} = v_{k+1} + \theta(v_{k+1} - \bar{v}_k) \end{cases} \quad (12)$$

where $\mathcal{P}(\phi) = \frac{\phi}{max(1, \|\phi\|)}$ is a fixed-point projection operator. The step sizes $\tau_u > 0, \tau_v > 0, \sigma_u > 0, \sigma_v > 0$ are solved using a pre-conditioning scheme following [21] while the relaxation variable θ is updated for every iteration as in [22]. The tensor $\mathrm{T}^{\frac{1}{2}}$ is calculated as:

$$\mathrm{T}^{\frac{1}{2}} = \exp(-\beta \, |I_0|^\eta) nn^T + n^\perp n^{\perp T} \quad (13)$$

where $n = \frac{\nabla I_0}{|\nabla I_0|}$ and $n^\perp$ is the vector normal to ∇I_0, while β and η are scalars controlling the magnitude and sharpness of the tensor. This tensor guides the propagation of the disparity information among neghboring pixels, while considering the natural image boundaries as encoded in n and $n^\perp$.

The so-called *resolvend* operator [22] $(I + \tau_u \partial G)^{-1}(\hat{u})$ is evaluated using the thresholding scheme:

$$(I + \tau_u \partial G)^{-1}(\hat{u}) = \hat{u} + \begin{cases} \tau_u \lambda I_u & \text{if } \bar{\rho} < -\tau_u \lambda I_u^2 \\ -\tau_u \lambda I_u & \text{if } \bar{\rho} > \tau_u \lambda I_u^2 \\ \bar{\rho}/I_u & \text{if } |\bar{\rho}| \leq \tau_u \lambda I_u^2 \end{cases}$$

where $I_u = \frac{d}{du} I_1(\mathbf{x}, u)$. In our tests, we used the parameter values: $\beta = 9.0$, $\eta = 0.85$, $\alpha_0 = 17.0$ and $\alpha_1 = 1.2$. The solved disparity is converted to depth by triangulating the unprojection rays using the unprojection function π^{-1}. This step is specific for the camera model used, hence we will not elaborate on methods to address this. Nevertheless, some camera models have closed-form unprojection function [9] [18] while others require non-linear optimizations [19].

B. Pre-rotation and calibration

We perform a calibration and pre-rotation of the image pairs before running the stereo estimation. In the same manner as the trajectory field generation, we create a calibration field which contains the rotation information as well as the difference in camera intrinsic properties (for binocular stereo case).

Again, we project a surface of arbitrary depth on the two cameras with projection function π_0 and π_1 while setting the translation vector $t = \mathbf{0}$. We then solve for the optical flow $\mathbf{w} = \mathbf{x}_1 - \mathbf{x}_0$. In this case the optical flow exactly represents the calibration field (see Figure 2). In case where $\pi_0 \neq \pi_1$, such as in binocular stereo, the calibration field will also contain the difference in intrinsic properties. For example, the difference in the image center results in the diagonal warping on our binocular camera system as seen in Figure 2. Using the calibration field, we warp the second image I_1 once, resulting in a translation only transformation.

C. Coarse-to-Fine Approach

Similar to most variational framework, we employ a coarse-to-fine (pyramid) technique to handle large displacement. Starting from a coarser level of the pyramid, we run N warping iterations and upscale both the current disparity and the warping vectors and carry the values on to the finer level.

One caveat of this approach on a fisheye image is the boundary condition especially for gradient and divergence calculations. To address this, we employ the Neumann and Dirichlet boundary conditions applied on a circular mask that

rejects pixels greater than the desired FOV. Specifically:

$$\nabla_x I = \begin{cases} I_{x+1,y} - I_{x,y} & \text{if } (x,y) \in M \text{ and} (x+1,y) \in M \\ I_{x,y} - I_{x-1,y} & \text{if } (x,y) \in M \text{ and} (x+1,y) \notin M \\ 0 & otherwise \end{cases}$$

$$\nabla_y I = \begin{cases} I_{x,y+1} - I_{x,y} & \text{if } (x,y) \in M \text{ and} (x,y+1) \in M \\ I_{x,y} - I_{x,y-1} & \text{if } (x,y) \in M \text{ and} (x,y+1) \notin M \\ 0 & otherwise \end{cases}$$

$$\nabla \cdot I = \begin{cases} I_{x,y} - I_{x-1,y} + I_{x,y} - I_{x,y-1}, & \text{if } (x,y) \in M \\ 0 & otherwise \end{cases} \tag{14}$$

where M indicates the region inside the circular mask. The mask is scaled accordingly using nearest-neighbor interpolation for every level of the pyramid. Moreover, by applying a mask, we also avoid the problem of texture interpolation with a zero-value during upscaling when the sample falls along the boundary of the fisheye image. We summarize our approach in Algorithm 1.

D. Timing Considerations

We implemented our method with C++/CUDA on an i7-4770 CPU and NVIDIA GTX 1080Ti GPU. We fix the iteration values based on the desired timing and input image size. For an 800x800 image, we found that the primal-dual iteration of 10 is sufficient for our application, with pyramid size $= 5$ and scaling $= 2.0$ (minimum image width = 50).

For the warping iteration, we plot the trade-off between accuracy and processing time in Figure 4 with respect to the percentage of erroneous pixels $> 1\,\text{px}$ with fixed $\delta u^{max} = 0.2\,\text{px}$. From the plot, we can see that the timing linearly increases with the number of iterations, but the accuracy exponentially decreases. Choosing a proper value for N needs careful considerations according to the application and scene.

Algorithm 1 Algorithm for variational fisheye stereo.

Require: I_0, I_1, $\hat{\xi}_1$, π
  Generate calibration field (Sec. IV-B)
  Generate trajectory field (Sec. III-B)
  Warp I_1 using the calibration field
  $\omega = 0$, $\mathbf{w}_\omega = 0$, $u_\omega = 0$, pyrlevel$= 0$
  while pyrlevel<pyrLevels **do**
    while $\omega < N$ **do**
      Warp I_1 using $\mathbf{w}_\omega$
      while $k <$nIters **do**
        TGV-L1: primal-dual optimization (12)
      end while
      Clip δu_ω (Sec. III-C)
      $u_{\omega+1} = u_\omega + \delta u_\omega$
      $\mathbf{w}_{\omega+\mathbf{1}} = \mathbf{w}_\omega + \delta u_\omega \gamma'_\omega$
    end while
    Upscale u_ω, $\mathbf{w}_\omega$, $\omega = N$
  end while
  Output: u

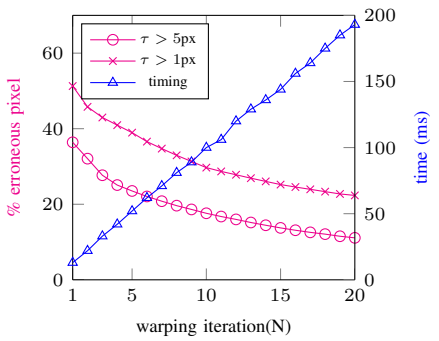

Fig. 4. Trade-off between accuracy and processing time for choosing the warping iteration (better viewed in color)

V. RESULTS AND COMPARISON

We present our results in the following sections. First, we show the effect of limiting the magnitude of the incremental disparity solution per warping iteration to the accuracy of the estmation. Then, we present the advantage and disadvantages of using the trajectory field compared to actually solving the epipolar curve for each pixel in every warping iteration. We also compare our method with an existing rectified variational stereo method and a discrete non-rectified stereo matching method.

For our comparisons, we use both synthetic and real datasets with ground truth depth. The synthetic dataset consists of a continuous sequence with 300 frames and four arbitrary stereo pairs re-rendered from [24] using Blender [25]. The real dataset consists of 144 image pairs with arbitrary camera motion (randomized rotation and non-zero translation) taken using a FARO Focus S 3D laser scanner [26] from a mixture of indoor and outdoor scenes. We also show some sample qualitative results on a commercial-off-the-shelf stereo camera fisheye system. Our dataset is available from our project page: https://www.github.com/menandro/vfs.

A. Limiting Incremental Disparity

To test the effect of limiting the incremental disparity, we measure the accuracy of our method on varying warping iteration and disparity limits. In Figure 5, we show the photometric error (absolute normalized intensity difference between I_0 and warped I_1) when $\delta u^{max} = 1.0\,\text{px}$ and $\delta u^{max} = 0.2\,\text{px}$. From the images, we can see that the photometric error is larger in areas with significant information (e.g. intensity edges and occlusion boundaries) when $\delta u^{max} = 1.0\,\text{px}$ compared to $\delta u^{max} = 0.2\,\text{px}$. This happens because it is faster for the optimization to converge in highly textured surfaces which results in overshooting from the tracked epipolar curve, as shown in Figure 3.

However, limiting the magnitude of δu has an obvious drawback. If the warping iteration is not sufficient, the estimated δu will not reach to its correct value which will result in higher error. We show this effect in Figure 6 where we plot various warping iterations N and show the accuracy of estimation with increasing δu^{max} using percentage of erroneous pixel measure

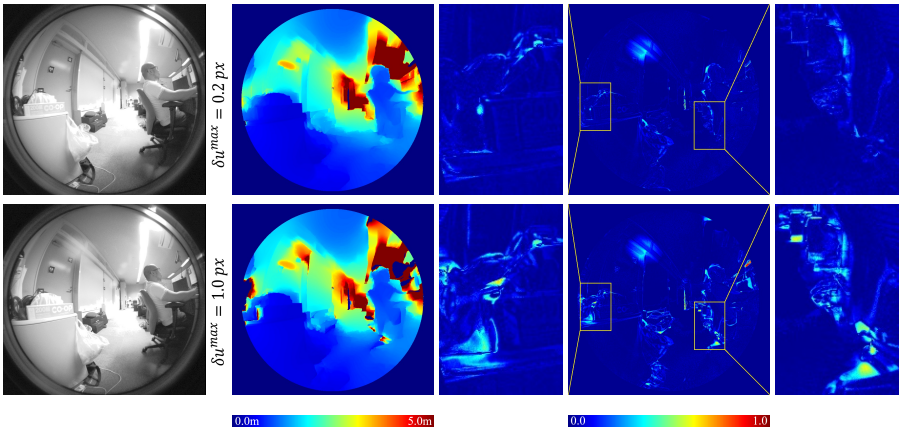

Fig. 5. Depth image and photometric error (absolute normalized intensity difference between I_0 and warped I_1) for varying values δu. Limiting the magnitude of δu per warping iteration reduces the error around sharp image gradients and occlusion boundaries.

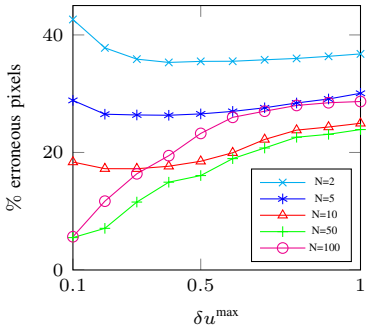

Fig. 6. Accuracy of disparity (percentage of erroneous pixels, $\tau > 1$ px) with limiting the magnitude of δu for different warping iteration values N.

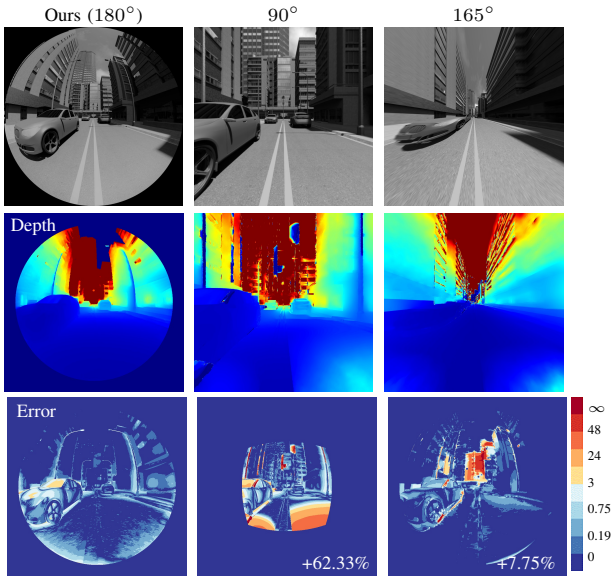

Fig. 7. Comparison between [2] with different field-of-view (90° and 165°) and our method. We compare the disparity error [23] as well as percentage of accuracy improvement by using our method.

($\tau > 1$ px) [23]. Clearly, increasing the iteration N and using smaller δu^{max} results in a more accurate estimation.

B. Trajectory Field Sampling vs. Epipolar Curve

Using the trajectory field allows us to perform GPU texture sampling and take advantage of hardware interpolation speed. In this section, we compare the trajectory sampling method (**VFS**) with actually solving the epipolar curve for every warping iteration with different camera models such as the unified camera model with [9](**VFS-ucm**) and without distortion [27] (**VFS-ucmd**), Kannala-Brandt model [19] (**VFS-kb**) and the equidistant model (**VFS-eq**). For analysis, we reduce the enhanced unified camera model [18] to **VFS-ucm** by assuming a spherical projection surface.

1) Time Consumption: The function defining the epipolar curve is dependent on the camera model being used which needs to be calculated per warping iteration. We do this by first unprojecting the 2D pixels of I_0 to an arbitrary 3D surface and then reprojecting the 3D points onto I_1 after applying the camera transformation. Methods that have closed form unprojection function such as **VFS-ucm** and **VFS-eq** are straightforward and fast, but do not handle real-world fisheye distortions [27].

TABLE I
ADDITIONAL TIME CONSUMPTION (PER 100 WARPING ITERATIONS) USING THE EPIPOLAR CURVE WITH DIFFERENT MODELS W.R.T. TRAJECTORY FIELD SAMPLING .

Method	Time(ms)
VFS-ucm	+20.4
VFS-ucmd	+139.6
VFS-kb	+219.6
VFS-eq	+30.4

On the other hand, more complete models such as **VFS-ucmd** and **VFS-kb** that discretely model the distortion are significantly slower due to the non-linear optimization needed in solving the unprojection function. In our implementation, we perform a few iteration of the Newton's method for **VFS-ucmd** and **VFS-kb** and achieved a convergence error (variation in the final value) of less than 2%. In our experiments, we observe a convergence around the inner regions of the image after four iterations. We show the additional consumed time compared to trajectory field sampling **VFS** per 100 warping iterations in Table I.

2) Accuracy: While the trajectory sampling approach allows us to perform a fast linear interpolation to determine the subpixel tangential vectors, it will still introduce additional errors because the path connecting two pixels is a curve and not a line. We determine this additional error experimentally by comparing **VFS** and **VFS-eq**. We use the equidistant model for comparison to remove the dependency on calibration accuracy when using other models and limit the resulting error to come only from the interpolation.

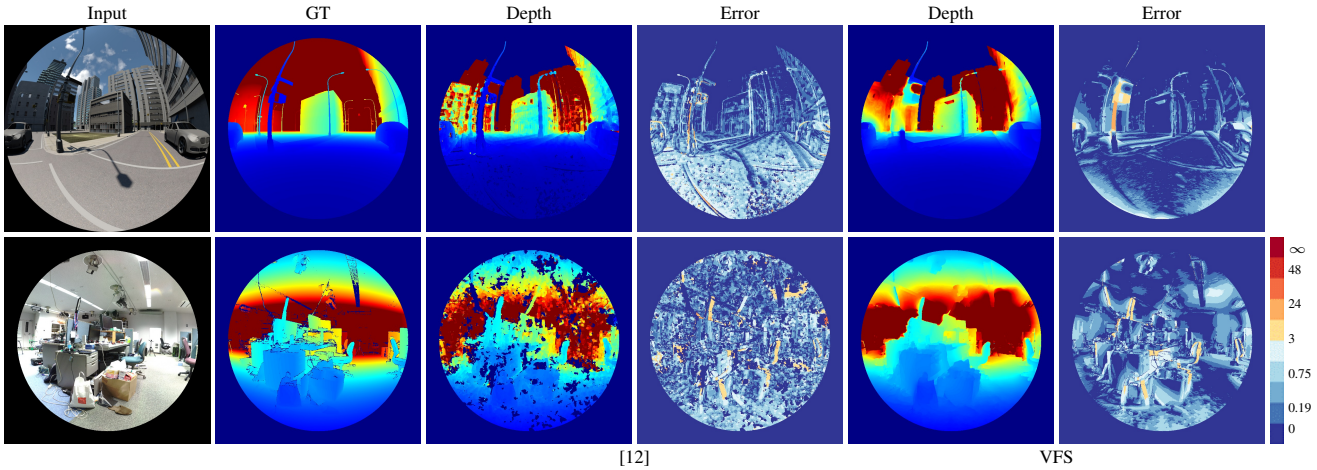

Fig. 8. Sample results on real and synthetic data with [12] and our method with disparity error [23].

TABLE II
AVERAGE DISPARITY ERROR (% OF ERRONEOUS PIXELS > 3PX), MEAN ABSOLUTE ERROR AND STANDARD DEVIATION COMPARISON ON REAL AND SYNTHETIC DATASET WITH RECTIFIED, NON-RECTIFIED, AND OUR METHOD.

method	syn-4				syn-Seq				real			
	Out-Noc%	MAE	σ	density%	Out-Noc%	MAE	σ	density%	Out-Noc%	MAE	σ	density%
rectified [2]	7.23	1.05	7.30	*44.44*	0.58	0.18	0.45	*84.03*	-	-	-	-
planesweep [12]	2.19	0.77	1.01	*95.92*	0.51	0.31	0.54	*90.88*	11.86	1.75	1.41	*45.03*
VFS	0.89	0.32	0.55	*100.00*	0.20	0.18	0.29	*100.00*	10.13	1.30	**1.31**	*100.00*
VFS-eq	**0.88**	**0.28**	**0.54**	*100.00*	**0.17**	**0.17**	**0.28**	*100.00*	**8.24**	**1.11**	1.33	*100.00*

For comparison, we use both the synthetic and real dataset and summarize the results in Table II. For the synthetic dataset, **VFS-eq** is expectedly more accurate than **VFS**. For the real dataset, **VFS-eq** also has lower mean absolute error (MAE) compared to **VFS** but with slightly higher standard deviation ($0.02\,\text{px}$) which is acceptable since the MAE is lower by $0.19\,\text{px}$.

C. Comparison with Rectified Method

We first compare our proposed approach with an existing rectified stereo method. To achieve a fair comparison, we use the same energy function and parameters in our implementation, except that we apply them in a rectified image. This rectified stereo approach is similar to the method presented in [2], except that we use intensity values instead of the census transform. We also explicitly applied a time-step preconditioning step and a relaxation after every iteration.

We compare our method with varying FOV for [2] on the synthetic dataset. We use the same erroneous pixel measure from the previous section and summarize the result in Table II using an FOV of $165°$ for [2]. We also compare the disparity error [23] as well as the improvement additional accurate pixels (see Figure 7) using the full $180°$ for our method and a FOV of $90°$ and $165°$ for [2]. We select these FOV's to highlight the compression problem when using rectification method and how it affects the estimation using the variational methods.

To better visualize the comparison, we transform the rectified error back to the original fisheye form. From the results, extreme compression around the center with ultra-wide angle ($165°$ and higher) rectification results in higher error especially for distant objects. With larger image area coverage, our approach do not suffer from this compression problem and maintains uniform accuracy throughout the image. Moreover, with the lower compression around the center ($90°$), the rectified method have increased error around the edges for closer objects (ground) due to increased displacement. This observation is arguably scene dependent. However, we believe that this type of scenario is ubiquitous to outdoor navigation especially in autonomous driving where objects close to the center of the camera are far away and objects close to the edge are nearby. Moreover, this highlights the importance of directly using the fisheye instead of performing rectification and undistortion.

Additionally, we found no significant difference in processing time because the warping techniques are both run in a single GPU kernel call and consumes the same texture memory access latency.

D. Comparison with Non-Rectified Method

In this section, we compare our method with planesweep implemented on a fisheye camera system [12] on real and synthetic scenes. The images were captured from two arbitrary camera location with non-zero translation. We show the sample results in Figure 8 and Table II.

One of the advantages of variational methods is the inherent density of the estimated depth when compared to discrete matching methods. In our experiments, we found that while

our method is denser and significantly smoother than [12], it is more prone to miss very thin objects such as poles. Moreover, because our method is built upon a pyramid scheme, very large displacements are difficult to estimate which is visible in the results when the object is very close to the camera (nearest ground area).

Nevertheless, we show in Table II that our method is overall more accurate compared to [12] even after we removed the ambigous pixels due to occlusion and left-right inconsistency. (In Table II, we use only the valid pixels in [12] for comparison).

E. Real-World Test

We tested our method on a laptop computer with NVIDIA GTX1060 GPU and an Intel RealSense T265 stereo camera, which has a $163 \pm 5^\circ$ FOV, global-shutter 848x800 grayscale image and a 30fps throughput. We show the sample results in Figure 1. We were able to achieve a 10fps with 5 warping iterations on a full image, and 30fps with 20 warping iterations on a half-size image. This system can be easily mounted on medium sized rover for SLAM applications.

VI. CONCLUSION AND FUTURE WORK

In this paper, we presented real-time variational stereo method for fisheye cameras without explicit image rectification and undistortion by generating a trajectory field induced by camera motion. From our results, we showed that our approach is denser, smoother and more accurate compared to non-rectified discrete methods and handles larger FOV compared to rectified methods while improving accuracy and without increasing processing time.

Because of the wider FOV of fisheye cameras, the disadvantage of most variational methods, which is handling large displacement (wide baseline or near objects), is highlighted. However, this can be overcome by using large displacement techniques or initialization with discreet methods (such as planesweep). Moreover, since the trajectory field can be generated by projecting an arbitrary surface to the camera images using a projection function, we can say that any projection model (spherical, equirectangular, and even perspective) should, at least theoretically, work with our method.

REFERENCES

[1] J. Stühmer, S. Gumhold, and D. Cremers, "Real-time dense geometry from a handheld camera," *Pattern Recognition. DAGM 2010. LNCS.*, vol. 6376, 2010.

[2] R. Ranftl, S. Gehrig, T. Pock, and H. Bischof, "Pushing the limits of stereo using variational stereo estimation," in *Proc. IEEE Inteligent Vehicles Symp.*, June 2012.

[3] J. Schneider, C. Stachniss, and W. Forstner, "On the accuracy of dense fisheye stereo," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, 2016.

[4] Z. Arican and P. Frossard, "Dense disparity estimation from omnidirectional images," in *IEEE Conf. Adv. Vid. Sig. Based Surv.*, September 2007.

[5] M. Schonbein and A. Geiger, "Omnidirectional 3d reconstruction in augmented manhattan worlds," in *Proc. IEEE Int. Work. Robots Sys.*, 2014.

[6] L. Bagnato, P. Frossard, and P. Vandergheynst, "A variational framework for structure from motion in omnidirectional image sequences," *J. Math Imaging Vis.*, vol. 41, pp. 182–193, 2011.

[7] W. Gao and S. Shen, "Dual-fisheye omnidirectional stereo," in *Proc. IEEE Int. Work. Robots Sys.*, 2017.

[8] D. Caruso, J. Engel, and D. Cremers, "Large-scale direct slam for omnidirectional cameras," in *Proc. IEEE Int. Work. Robots Sys.*, 2015.

[9] C. Geyer and K. Daniilidis, "A unifying theory for central panoramic systems and practical applications," in *Proc. IEEE Europ. Conf. Comput. Vis.*, July 2000, pp. 445–461.

[10] R. Bunschoten and B. Krose, "Robust scene reconstruction from an omnidirectional vision system," *IEEE Trans. Robot. Automat.*, 2003.

[11] B. Khomutenko, G. Garcia, and P. Martinet, "Direct fisheye stereo correspondence using enhanced unified camera model and semi-global matching algorithm," in *Int. Conf. on Control, Automation, Robotics and Vision*, 2016.

[12] C. Hane, L. Heng, G. H. Lee, A. Sizov, and M. Pollefeys, "Real-time direct dense matching on fisheye images using plane-sweeping stereo," in *Proc. Int. Conf. 3D Vis.*, December 2014.

[13] S. Im, H. Ha, F. Rameau, H.-G. Jeon, G. Choe, and I. S. Kweon, "All-around depth from small motion with a spherical panoramic camera," in *Proc. IEEE Europ. Conf. Comput. Vis.*, 2016, pp. 156–172.

[14] B. Micusik and T. Pajdla, "Structure from motion with wide circular field-of-view cameras," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1135–1149, July 2006.

[15] R. Zabih and J. Li, "Non-parametric local transforms for computing visual correspondence," in *Proc. IEEE Europ. Conf. Comput. Vis.*, 1994, pp. 151–158.

[16] N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert, "Highly accurate optical flow computation with theoretically justified warping," *Int. J. Comput. Vis.*, vol. 67, pp. 141–158, 2006.

[17] T. Svoboda, T. Pajdla, and V. Hlavac, "Epipolar geometry for panoramic cameras," in *Proc. IEEE Europ. Conf. Comput. Vis.*, 1998, pp. 218–231.

[18] B. Khomutenko, G. Garcia, and P. Martinet, "An enhanced unified camera model," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 137–144, January 2016.

[19] J. Kannala and S. S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, fisheye lenses," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 28, no. 8, pp. 1335–1340, September 2006.

[20] W. Milled and J. Pesquet, "Disparity map estimation using a total variation bound," in *3rd Canadian Conference on Computer and Robot Vision*, 2006, pp. 48–55.

[21] T. Pock and A. Chambolle, "Diagonal pre-conditioning for first order primal-dual algorithms in convex optimization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011.

[22] A. Chambolle and T. Pock, "A first-orer primal-dual algorithm for convex problems with applications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, May 2011.

[23] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Robot. Res.*, 2013.

[24] Z. Zhang, H. Rebecq, C. Forster, and D. Scaramuzza, "Benefit of large field-of-view cameras for visual odometry," in *Proc. IEEE Int. Conf. Robot Automat.*, 2016.

[25] [Online]. Available: http://www.blender.org

[26] [Online]. Available: http://www.faro.com

[27] X. Ying and Z. Hu, "Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model," in *Proc. IEEE Europ. Conf. Comput. Vis.*, 2004, pp. 442–455.