Omnidirectional DSO:
Direct Sparse Odometry with Fisheye Cameras

Hidenobu Matsuki[1,2], Lukas von Stumberg[2,3], Vladyslav Usenko[2], Jörg Stückler[2] and Daniel Cremers[2]

Abstract— We propose a novel real-time direct monocular visual odometry for omnidirectional cameras. Our method extends direct sparse odometry (DSO) by using the unified omnidirectional model as a projection function, which can be applied to fisheye cameras with a field-of-view (FoV) well above 180 degrees. This formulation allows for using the full area of the input image even with strong distortion, while most existing visual odometry methods can only use a rectified and cropped part of it. Model parameters within an active keyframe window are jointly optimized, including the intrinsic/extrinsic camera parameters, 3D position of points, and affine brightness parameters. Thanks to the wide FoV, image overlap between frames becomes bigger and points are more spatially distributed. Our results demonstrate that our method provides increased accuracy and robustness over state-of-the-art visual odometry algorithms.

I. INTRODUCTION

Visual odometry (VO) with monocular cameras has widespread applications, for instance, in autonomous driving, mobile robot navigation or virtual/augmented reality. The benefit of using only a monocular vision system is that only a single low-cost camera is needed which is simple to maintain and often available in commodity hardware. Hence, research in the field of VO with monocular cameras is actively pursued in recent years [1], [2], [3]. Since VO algorithms estimate 3D structure and 6-DoF camera motion from visual information, sufficient texture needs to be present in the images so that correspondences can be observed between different frames. A major limiting factor for correspondence estimation is the field-of-view (FoV) of the camera. This becomes especially apparent in environments with sparse features such as indoor environments with textureless walls, or dynamic environments where robust tracking requires that the static part of the environment is sufficiently visible in the frames. Thus, wide FoV cameras are beneficial for VO.

It is, however, not straightforward to make full use of the wide FoV images in standard VO pipelines. Typically, these approaches are designed for the pinhole camera model. It projects measurements of 3D points onto an image plane and causes strong distortions of the image for a FoV of more than approx. 150 degrees. To avoid the processing of distorted image regions, the images are typically cropped to a smaller part in the inner region, resulting in an effectively lower FoV.
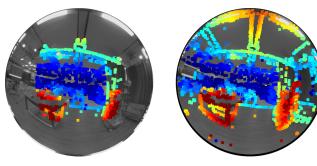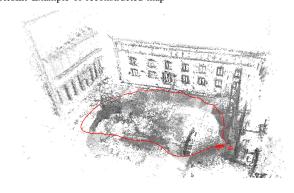

Fig. 1. **Top:** Active pixels in DSO for different camera models (left: pinhole, right: omnidirectional). With the pinhole model, DSO can only use a small part of the original fisheye image. In contrast, with the omnidirectional model, DSO can use pixels across the whole fisheye image. **Bottom:** Example of reconstructed map

There are two main approaches to increase the FoV of a VO system: Firstly, optimization can be performed in a window of frames in which frames share varying mutual image overlap. Examples are fixed-lag smoothing approaches such as the multi-state constrained Kalman Filter (MSCKF [4]) or Direct Sparse Odometry (DSO [5]). For some approaches, the use of a camera projection model such as the unified omnidirectional model can be a viable option to avoid image distortions. In this paper, we propose a state-of-the-art direct visual odometry method that incorporates the unified omnidirectional model as used in [6] into a fixed-lag smoothing approach to VO.

We extend DSO to make seamless, full use of fisheye images (Fig. 1) and to jointly optimize for the model parameters including camera intrinsics and extrinsics, sparse point depths and affine brightness parameters.

In experiments, we evaluate our approach on a benchmark of image sequences captured with a wide FoV fish-eye lens. We compare our approach to other state-of-the-art VO approaches such as DSO or LSD-SLAM [7] and validate that our method outperforms the previous state-of-the-art on benchmark datasets.

[1]The author is with the University of Tokyo, Japan matsuki.hidenobu@gmail.com

[2]The authors are with Technical University of Munich, Germany {matsuki, stumberg, usenko, stueckle, cremers}@in.tum.de

[3]The author is with Artisense GmbH, Germany

The paper is organized as follows: We first review the state-of-the-art in Sec. II. In Secs. III and IV, we introduce notation, the pinhole and the unified omnidirectional camera model. In Sec. V, we describe the pipeline of our omnidirectional DSO method. Our method is based on Direct Sparse Odometry [5] and integrates the unified omnidirectional camera model similar to [6]. We give a brief review of DSO and continue by detailing distance estimation along the epipolar curve with the unified omnidirectional camera model. In Sec. VI, we evaluate the performance of our method on publicly available datasets and compare it to the state-of-the-art.

II. RELATED WORK

Indirect visual-odometry methods: Early works on visual odometry and visual simultaneous localization and mapping (SLAM) have been proposed around the year 2000 [1], [8], [9] and relied on matching interest points between images to estimate the motion of the camera. While visual odometry methods focus on incremental real-time tracking of the camera pose with local consistency, SLAM approaches jointly estimate a globally consistent trajectory and map. Many of these approaches were based on probabilistic filtering (e.g. [10], [11]). For example, MonoSLAM proposed by Davison *et al.* [10] is a real-time capable approach based on the Extended Kalman Filter. However, since the landmark estimates are part of the filtered state space, the method is only capable to map small work-spaces due to computational limitations. A technical breakthrough occurred in 2007 when Klein *et al.* proposed PTAM [12], a keyframe-based approach that performs tracking and mapping in separate threads. Similarly, many current VO/SLAM algorithms also use keyframes and apply multithreading to perform locally consistent tracking and mapping in real-time while optimizing for global consistency in a slower SLAM optimization layer. ORB-SLAM [13] is the current state-of-the-art indirect and keyframe-based visual SLAM algorithm, which performs full bundle adjustment in a separate optimization layer.

Direct visual-odometry methods: More recently, direct methods have gained popularity for VO and SLAM. Direct methods avoid the extraction of geometric features such as keypoints but directly estimate odometry and 3D reconstruction from pixel intensities. Since they do not compress the image content to a small set of typically hand-crafted features, direct methods can use much more information in an image such as edges or shaded surfaces. This enables more dense 3D reconstructions while indirect methods only produce sparse point reconstructions. Direct visual odometry methods have also been proposed for RGB-D cameras, e.g. [14]. The method extracts RGB-D keyframes and tracks the camera motion towards the recent keyframe using direct image alignment based on the measured depth. LSD-SLAM [7] has been the first direct visual SLAM approach for monocular cameras that is capable of mapping large scale environments in realtime. It tracks the camera motion, produces a semi-dense map and performs pose graph optimization to obtain a consistent global map. The semi-

dense maps can be adapted to a variety of uses such as surface estimation in AR, 3D object recognition and semantic labeling [15], [16], [17].

In pose graph optimization, the individual direct image alignment measurements are aggregated in a relative pose measurement between keyframes. This neglects the fine-grained correlations of the direct measurements and requires linearization and Gaussian approximations to condense the measurement. Recently, Direct Sparse Odometry (DSO) has been proposed by Engel *et al.* [5]. In contrast to LSD-SLAM, DSO jointly optimizes multiple model parameters such as camera intrinsics/extrinsics, affine brightness parameters and depth in realtime within a window of keyframes by using a sparse set of points in each keyframe. This approach currently defines the state-of-the-art performance among visual odometry methods in terms of trajectory accuracy.

Limitation of monocular visual-odometry: Since monocular visual odometry estimates camera motion and scene reconstruction with a single camera, scale is invariably ambiguous and prone to drift. To recover metric scale, VO/SLAM methods are typically extended with additional sensors such as stereo camera, depth sensors or IMUs [18], [14], [19]. More recently, CNN-based depth predictions are combined with monocular visual SLAM [17]. In DSO and in our methods, due to the windowed optimization and marginalization, scale drift is comparably smaller than in tracking-based VO such as the VO front-end in LSD-SLAM.

Visual-odometry methods with omnidirectional camera models: To benefit from a larger FoV, VO and SLAM methods have also been extended for wide-FoV cameras [20], [21], [22], [23]. In particular, Omnidirectional LSD-SLAM [6] has been the first direct visual SLAM approach for fisheye cameras which runs in real-time. By incorporating the unified omnidirectional camera model, it works even for cameras with a FoV of more than 180 degree. In our approach, we also use the unified omnidirectional camera model, but optimize for a multitude of parameters such as camera intrinsics/extrinsics, affine brightness parameters and depth within a window of frames in DSO. We demonstrate how the combination of an optimization window with the extended FoV improves performance over direct baseline methods such as DSO and LSD-SLAM. Zhang *et al.* [24] developed Semi Direct Visual Odometry (SVO) [2] for fisheye cameras and compared the performance with different FoV under the same image resolution. According to their paper, the optimal FoV also depends on the environment so that a wider FoV does not always improve results. In indoor environments, however, they found that the wider FoV tends to increase performance.

Contribution: In this paper we present an omnidirectional extension of Direct Sparse Odometry. This is the first fisheye-based direct visual odometry which runs in real time and jointly optimizes multiple model parameters - camera pose, depth of points, camera intrinsics and affine brightness parameters.
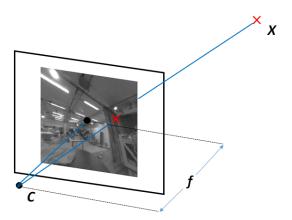
Fig. 2. Pinhole camera model. The image coordinate of a 3D point is found through direct projection onto the image plane through the camera center C.
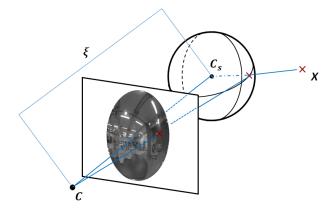

Fig. 3. Unified Omnidirectional Camera Model. The image coorinate of a 3D point is found by first projecting it on the unit sphere, and then projecting it to image plane. The offset between camera center C and unit sphere center C_s is parameterized as ξ.

III. NOTATION

We basically follow the notation in [6]: We denote scalars u with light lower-case letters, while light upper-case letters represent functions I. For matrices and vectors, we use bold capital letters $\mathbf{R}$ and bold lower-case letters $\mathbf{x}$, respectively. With $\mathbf{u} = [u,v]^T \in \Omega \subset \mathbb{R}^2$ we will generally denote pixel coordinates, where Ω denotes the image domain. Point coordinates in 3D are denoted as $\mathbf{x} = [x,y,z]^T \subset \mathbb{R}^3$. The operator $[\ ]_i$ extracts the i-th row of a matrix or vector. We represent camera poses by matrices of the special Euclidean group $\mathbf{T}_i \subset SE(3)$. They transform a 3D coordinate from the camera coordinate system to the world coordinate system. In general, a camera projection function is a mapping $\pi : \mathbb{R}^3 \to \Omega$. Its inverse $\pi^{-1} : \Omega \times \mathbb{R}^+ \to \mathbb{R}^3$ unprojects image coordinates using their inverse distance $d \in \mathbb{R}^+$. Camera frames are centered at C and the optical axis is along the z-axis pointing forward in positive direction. For optimization, we represent a 3D point by its image coordinate $\mathbf{p}$ and inverse distance d_p in its host keyframe in which it is estimated.

IV. CAMERA MODELS

In the following, we describe the two camera models used in this paper; the pinhole model and the unified omnidirectional model.

A. Pinhole Model

The pinhole camera model is the most popular camera model in literature. Each 3D point is projected onto a normalized image plane located at $z = 1$ and then linearly transformed into pixel coordinates. This is mathematically formulated as

$$\pi_u(\mathbf{x}) = \begin{bmatrix} f_x & 0 \\ 0 & f_y \end{bmatrix} \begin{bmatrix} x/z \\ y/z \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \qquad (1)$$

where f_x, f_y are the focal lengths, and c_x, c_y is the principal point. The projection model is illustrated in Fig. 2.

This is the most simple model because the projection function is linear in homogeneous coordinates. However, this does not consider the nonlinear image projection of fisheye images and is not suitable for wide FoV cameras. Radial and tangential distortion functions can be applied to remove

small non-linear distortions, however, the pinhole projection assumes that the measured 3D points are beyond the image plane, i.e. their depth is larger than the focal length. This limits the field-of-view below 180 degree.

B. Unified Omnidirectional Model

We use the unified omnidirectional camera model which has been originally proposed in [25] for a wide FoV fish-eye camera. The major advantages of this model are; (1) it can accurately model the geometric image formation for a wide range of imaging devices and lenses, (2) the unprojection function π^{-1} can be expressed in closed-form. A 3D point in Euclidean camera coordinates is first projected onto a camera-centered unit sphere (see Fig. 3). Then the point is projected to an image plane as in the pinhole model through a center with an offset $-\xi$ along the z axis. The model has five parameters, focal length f_x, f_y, camera centers c_x, c_y and the distance between camera center and unit sphere center ξ.

The projection of a point is computed as

$$\pi_u(\mathbf{x}) = \begin{bmatrix} f_x \frac{x}{z+\|\mathbf{x}\|\xi} \\ f_y \frac{y}{z+\|\mathbf{x}\|\xi} \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix} \qquad (2)$$

where $\|\mathbf{x}\|$ denotes the Euclidean norm of $\mathbf{x}$. The unprojection function for this model is

$$\pi_u^{-1}(\mathbf{u},d)$$
$$= \frac{1}{d} \left(\frac{\xi + \sqrt{1 + (1-\xi^2)(\tilde{u}^2 + \tilde{v}^2)}}{\tilde{u}^2 + \tilde{v}^2 + 1} \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ \xi \end{bmatrix} \right) \qquad (3)$$

where

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \end{bmatrix} = \begin{bmatrix} (u-c_x)/f_x \\ (v-c_y)/f_y \end{bmatrix}. \qquad (4)$$

Note that for $\xi = 0$ the model reduces to the pinhole model. We combine the unified omnidirectional model with a small radial-tangential distortion model to correct for lens imperfections. The model is used to undistort the raw images before applying the unified omnidirectional model.
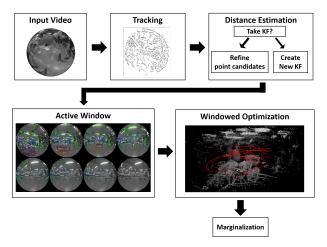
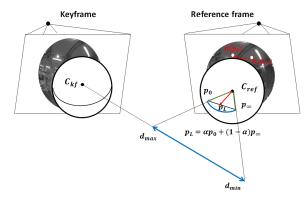Fig. 4. System overview of omnidirectional DSO (OmniDSO).

Fig. 5. Stereo Matching with the unified omnidirectional camera model: Correspondence search for stereo matching is performed by mapping the distance search interval along a ray to a directional interval on the projection unit sphere of the other camera.

V. System Overview

A. Model Formulation

DSO jointly optimizes camera poses, point depths and affine brightness parameters in a window of recent frames. As a direct method it optimizes for photometric consistency. DSO also takes the photometric calibration of the image formation process into account.

The energy function which represents the photometric error between two frames is formulated as

$$E_{\mathbf{p}j} := \sum_{\mathbf{p} \in N_p} w_p \| (I_j[\mathbf{p}'] - b_j) - \frac{t_j e^{a_j}}{t_i e^{a_i}} (I_i[\mathbf{p}] - b_i) \|_{\gamma} \qquad (5)$$

where we measure the photometric error of a point $\mathbf{p} \subset \Omega_i$ in reference frame I_i with respect to a target frame I_j through the weighted sum of squared differences (SSD) over a small pixel neighborhood N_p. w_p is a gradient dependent weighting. t_i, t_j are the exposure times of the images I_i, I_j; a and b are affine brightness correction factors; and $\| \cdot \|_{\gamma}$ denotes the Huber norm. $\mathbf{p}'$ is the reprojected point position of $\mathbf{p}$ with inverse distance d_p. $\mathbf{p}'$ is given by

$$\mathbf{p}' = \pi(\mathbf{R}\pi^{-1}(\mathbf{p}, d_p) + \mathbf{t}) \qquad (6)$$

with

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} := \mathbf{T_j}\, \mathbf{T_i}^{-1}. \qquad (7)$$

The photometric error terms of the active window of frames are

$$E_{photo} := \sum_{i \in F} \sum_{\mathbf{p} \in P_i} \sum_{j \in obs(\mathbf{p})} E_{\mathbf{p}j} \qquad (8)$$

where F is the set of frames in the active window, P_i are the points in frame i, and $obs(\mathbf{p})$ is the set of frames which observe the point $\mathbf{p}$. For tracking, this error function is minimized with respect to the relative camera pose $\mathbf{T_{ij}}$ between $\mathbf{T_i}$ and $\mathbf{T_j}$. For window optimization, the function is optimized for all variables $(\mathbf{T_i}, \mathbf{T_j}, d, \mathbf{c}, a_i, a_j, b_i, b_j)$, where $\mathbf{c}$ are camera intrinsic parameters. Different to [5], we parametrize points with the inverse distance $d = |\mathbf{x}|^{-1}$ instead of inverse depth. This allows us to model points behind the camera as well (s. Fig. 4 for an overview).

B. Distance Estimation along the Epipolar Curve

Once a frame is successfully tracked, we perform stereo matching to refine the inverse distance of candidate points. When a candidate point gets included into the photometric bundle adjustment, this estimated distance serves as an initialization. DSO searches for corresponding points along the epipolar line similar to [3]. However, when performing stereo matching on fisheye images using the unified omnidirectional model, rays through camera center and pixels project no longer to epipolar lines but curves (more precisely they are conics [25]).

We now describe the mathematical formulation of the epipolar curve. Similar as in [6], we define two points $\mathbf{p_0}$, $\mathbf{p_\infty} \in \mathbb{R}^3$ which lie on the unit sphere around a projective center C_{ref} and correspond to the maximum and minimum inverse distance d_{max}, d_{min} of the search interval,

$$\mathbf{p_0} := \pi_s(\mathbf{R}\pi_u^{-1}(\mathbf{p}, d_{min}) + \mathbf{t}) \qquad (9)$$
$$\mathbf{p_\infty} := \pi_s(\mathbf{R}\pi_u^{-1}(\mathbf{p}, d_{max}) + \mathbf{t}). \qquad (10)$$

Here, the function π_s projects the 3D points onto the unit sphere. π_u^{-1} is the unprojection function of the unified model, and $\mathbf{p}$ is the pixel in the keyframe we are trying to match. We then express the linear interpolation of these points with $\alpha \in [0, 1]$ as

$$\mathbf{p}_L(\alpha) := \alpha\mathbf{p_0} + (1 - \alpha)\mathbf{p_\infty} \qquad (11)$$

We find the epipolar curve by projecting this line to the target image,

$$\mathbf{u}_L(\alpha) := \pi_u(\mathbf{p}_L(\alpha)) \qquad (12)$$

We then search correspondences along the epipolar curve by starting at $\mathbf{u}_L(0)$ and incrementing α. The increment in α for 1 pixel in the image is determined by first-order Taylor approximation of $\mathbf{u}_L$ as

$$\delta\alpha := \| \mathbf{J}_{\mathbf{u}_L}|_\alpha \|^{-1} \qquad (13)$$

This value needs to be re-calculated for each increment while for the pinhole camera model a constant step size can be used

for epipolar line search. However, in DSO and LSD-SLAM, a distance prior is available from previous frame estimates or through initialization. Hence, the search interval is typically small and real-time computation is facilitated.

C. Frame Management

DSO maintains a constant number of N_f active keyframes in the optimization window (e.g. $N_f = 7$). It keeps track of the camera motion in every new frame by tracking it towards the latest keyframe and its sparse distance map (step 1). If the changes in the observed scene are too large towards the latest keyframe, a keyframe is created from the new frame (step 2). Afterwards, we marginalize one or more frames to keep the number of keyframes constrained (step 3).

1) Initial Frame Tracking: For tracking, conventional direct image alignment is performed in a 5 level image pyramid. The scene and brightness change is continuously estimated and if the change is bigger than a certain threshold value, the frame is selected as a new keyframe.

2) Keyframe Creation: When a keyframe is created, candidate points are selected considering space distribution and image gradient. We initialize the inverse distance estimate of these candidate points with a large variance that corresponds to a range from zero to infinity. After each subsequent new frame has been tracked towards this new key frame, the inverse distance estimates are refined using observations in the new frame which we obtain through the epipolar search (Sec. V-B).

3) Keyframe Marginalization: When the number of active keyframes grows above N_f, old points and frames are removed from the active window considering the number of visible points and frame distribution. Let us denote the active keyframes with $I_1, \ldots, I_n$, while I_1 is the newest and I_n the oldest keyframe. Our marginalization strategy follows [5]: a. We never marginalize the latest two keyframes (I_1, I_2). b. We marginalize a frames if the percentage of its points that are visible in I_1 drops below 5%. c. If the number of active keyframes grows above N_f, we marginalize one keyframe based on a heuristic distance score [5]. The score favors active keyframes that are spatially distributed close to the latest keyframe. Finally, candidate points are activated and added to the optimization.

D. Windowed Optimization

Our windowed optimization and marginalization policy follows [5]. As we formulated in (8), joint optimization is done for all activated points over all active keyframes. Nonlinear optimization for the photometric error function is performed using the Gauss-Newton algorithm [5]. All the variables including camera pose, inverse distance of active points, camera intrinsic parameters, and affine brightness parameters are jointly optimized. After the minimization of the photometric error, we marginalize old keyframes and points using the Schur complement ([5], [19]) if the number of active keyframes in the optimization window grows beyond N_f. We keep the optimization problem sparse by first marginalizing those points that are unobserved in

the two latest keyframes. We also marginalize the points which are hosted in the keyframe which will be marginalized. Afterwards, the keyframe is marginalized and removed from the optimization window.

VI. EVALUATION

We provide an experimental evaluation of our method both in terms of accuracy and robustness. We perform a quantitative comparison against the state of the art visual odometry methods on public benchmark datasets. We also qualitatively assess the benefit of a wider FoV. We used two public datasets for evaluation: TUM SLAM for Omnidirectional Cameras dataset first employed in [6] as a small-scale indoor benchmark and Oxford RobotCar dataset [26] as a large-scale outdoor benchmark.

A. TUM SLAM for Omnidirectional Cameras Dataset

The TUM omnidirectional dataset provides wide FoV fisheye images of indoor scenes. It also includes ground truth trajectory data recorded with a motion capture system and calibrated unified omnidirectional model camera parameters. The dataset consists of 5 indoor sequences with rapid and handheld motion. The camera is global shutter with a 185° FoV fisheye lens and recorded images of resolution 1280×1024. We cropped and scaled the images to a 480×480 resolution centered around the principal point. With this dataset, we compared 5 direct visual odometry algorithms; normal DSO [5], omnidirectional DSO (our method), normal LSD-SLAM [7] (without loop-closing), omnidirectional LSD-SLAM [6] (without loop-closing), and semi-direct visual odometry (SVO [2]). Note that we turned off loop-closing of LSD-SLAM to evaluate the performance of its underlying visual odometry in terms of the overall drift per sequence.

1) Accuracy Comparison: Following the evaluation methodology in [6] we measured the translational root mean square error (RMSE) between the estimated and the ground-truth camera translation for each evaluated sequence. The estimated camera position is calculated for all keyframes, and Sim(3) alignment with the ground-truth trajectory data is performed. Since the multi-threaded execution of DSO introduces non-deterministic behavior, we ran the algorithm 5 times for each sequence per method, then took the median RMSE. The results are shown in Table I. Some representative visual results are shown in Fig. 6 and 7. Table II shows the length of the trajectory estimated by OmniDSO.

We make two observations from Table I: First, DSO is more robust and accurate than SVO and LSD-SLAM without loop closure. Since the dataset scene contains a lot of small loops, this contributes to SLAM performance. However, as a pure visual odometry, DSO shows much better performance. This means sparse joint bundle adjustment and windowed optimization increases the performance of direct visual odometry. Second, the use of the unified omnidirectional camera model further improves the performance both for DSO and LSD-SLAM. Although for some sequences a clear performance improvement cannot be observed, considering the average result over all sequences, fisheye visual odometry
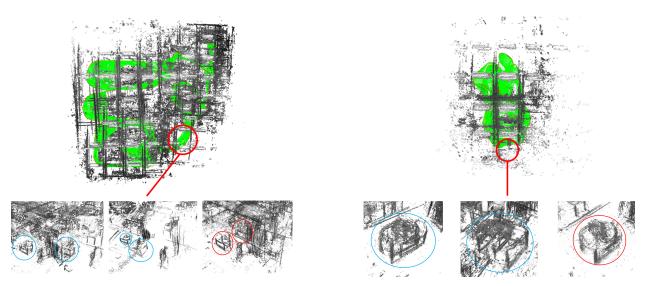
Fig. 6. Accumulated drift (left: normal DSO, middle: OmniLSD (VO), right: OmniDSO). Compared to the other 2 methods, OmniDSO demonstrates better performance both in terms of translation and scale drift.
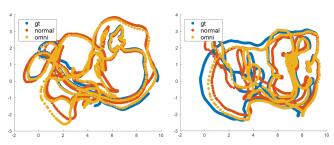

Fig. 7. **Top:** Horizontal trajectory estimates for T1 and T3.

demonstrates a clear advantage over using the pinhole camera model (approx. $0.157\,\mathrm{m}$ improvement of OmniDSO over DSO). Our OmniDSO incorporates both of these benefits and outperforms other existing visual odometry methods. From Tables I and II, we can clearly see the performance difference in T5, which has the longest trajectory among sequences.

TABLE I

ABSOLUTE RMSE IN METERS. DSO OUTPERFORMS OTHER METHODS AND ESPECIALLY OUR OMNIDSO SHOWS THE SMALLEST RMSE IN AVERAGE.

	DSO	**Ours**	LSD	OmniLSD	SVO	OmniLSD-SLAM
T1	0.243	**0.144**	0.751	0.873	1.22	(0.053)
T2	**0.450**	0.497	1.43	1.22	0.980	(0.051)
T3	0.499	**0.258**	1.43	0.551	1.28	(0.046)
T4	**0.240**	0.254	0.731	0.752	0.734	(0.045)
T5	1.47	**0.960**	1.91	1.69	3.06	(0.036)
Avg.	0.580	**0.423**	1.25	1.02	1.46	(0.046)

TABLE II

GENERATED TRAJECTORY LENGTH IN METERS.

T1	T2	T3	T4	T5
101	83.5	87.5	93.5	138

2) Benefit of Large Field of View: One of the major advantages of using wide FoV is that the image is more likely to contain strong gradient pixels which are beneficial

for visual odometry. Fig. 8 shows active points in a keyframe for the textureless scene in the T5 sequence for DSO with the pinhole and the unified omnidirectional camera model. Note that DSO uses points with strong image gradient. When comparing the two camera models, it is apparent that the omnidirectional camera model can track on a larger spatial distribution of points with a larger variety of gradient directions, while the pinhole camera model only observes a smaller inner part of the images.

Another benefit of using the omnidirectional model is the bigger image overlap between frames. Fig. 9 and 10 show the estimated trajectory and constraints between active keyframes in the current optimization window in the same scene for DSO and OmniDSO. As we described in V-C, a keyframe is marginalized from the window when the number of visible points from the current frame falls below 5%. Due to the increased image overlap, the omnidirectional camera model allows for maintaining a keyframe longer in the optimization window, thus increasing the spatial distribution of keyframes and the effective part of the scene that is observed within the window compared to DSO with the pinhole model. To evaluate this overlap effect numerically, we tested different maximum numbers of keyframes for the windowed optimization and compared the results. The tested number of keyframes (N_f) were $N_f = 7$ (default), $N_f = 5$ and $N_f = 3$. Table III shows the result of reducing keyframes and Table IV shows the performance difference. With Kf5-Kf7 we denote the difference in RMSE between $N_f = 5$ and $N_f = 7$. The smaller the number the more robust the approach is to keyframe number reduction. Table IV shows that the performance declines as the keyframe number decreases for both methods. However, as shown in Table IV, the decrease of our method (Omni DSO) is smaller than that of normal DSO. This demonstrates that the bigger visual overlap due to the use of the omnidirectional model contributes to maintaining performance even if less keyframes are used.
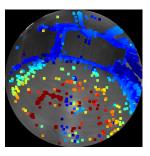
Fig. 8. Active points for optimization in a textureless scene (left: DSO, right: OmniDSO). With a wider field of view, it is less likely that the observed scene exhibits a degeneracy in any spatial direction. The left image exhibits an intensity constancy along the diagonal and hence motion estimation in that direction is not constrained.
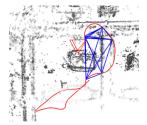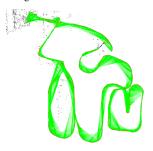

Fig. 9. Active keyframes in optimization window after camera rotation (left: DSO, right: OmniDSO). In our method, keyframes are not marginalized even after big camera rotation.
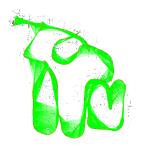

Fig. 10. Constraints between keyframes (green lines) during optimization (left: DSO, right: OmniDSO). OmniDSO maintains more spatially distributed keyframes within the optimization window.

3) Timing measurement: Table V shows the measured average time over the dataset (5 runs per sequence) in milliseconds taken for tracking and mapping (windowed optimization) steps. To measure these times, images have been processed at resolution 480×480. We used a computer with Intel Core i7-7820HK CPU at $2.90\,\mathrm{GHz}$ with 4 cores. These results demonstrate the realtime capability of our method since each frame can be tracked at at least $100\,\mathrm{Hz}$ and mapped at more than $15\,\mathrm{Hz}$. These results also show that the mapping process is sped up by reducing the number of keyframes. Even with less number of keyframes ($N_f = 5$), Table III displays that our method still outperforms normal DSO in average with $N_f = 7$ keyframes.

B. Oxford Robotcar Dataset

We also evaluated our algorithm on the Oxford Robotcar dataset in a large-scale outdoor scenario. The dataset contains more than 100 repetitions of a consistent route at different times under different weather, illumination and traffic conditions. The images are collected from 6 cameras mounted on the vehicle, along with LIDAR, GPS and INS ground truth.

TABLE III
ABSOLUTE RMSE FOR DIFFERENT NUMBER OF KEYFRAMES IN METERS. FOR ALL KEYFRAME NUMBER SETTINGS, OMNIDSO SHOWS SMALLER AVERAGE RMSE.

	Normal DSO			Omni DSO		
N_f	7	5	3	7	5	3
T1	0.243	0.288	0.35	0.144	0.145	0.218
T2	0.450	0.451	0.658	0.497	0.569	0.779
T3	0.499	0.642	0.624	0.258	0.304	0.382
T4	0.240	0.223	0.672	0.254	0.246	0.416
T5	1.47	1.71	1.82	0.96	1.07	1.14
Avg.	0.580	0.663	0.825	**0.423**	**0.467**	**0.587**

TABLE IV
PERFORMANCE DIFFERENCE BETWEEN USED NUMBER OF KEYFRAMES IN METERS. THE AVERAGE PERFORMANCE DECLINE OF OMNIDSO WITH RESPECT TO KEYFRAME NUMBER CHANGE IS SMALLER

	Kf5-Kf7		Kf3-Kf7	
	Normal	Omni	Normal	Omni
T1	0.045	0.001	0.107	0.074
T2	0.001	0.072	0.208	0.282
T3	0.143	0.046	0.125	0.124
T4	-0.017	-0.008	0.432	0.162
T5	0.240	0.110	0.35	0.18
Avg.	0.082	**0.044**	0.244	**0.164**

As an evaluation benchmark, we used videos taken by a rear-mounted global shutter camera with a $180°$ FoV fisheye lens. The raw image resolution is 1024×1024 and we cropped and scaled it to a 600×600 resolution. We obtained camera intrinsic parameters using the Kalibr calibration toolbox [27] with the original checkerboard sequence. The dataset has 3 types of routes (Full, Alternate and Partial). Full route covers the whole original route and consists of 5 sequences. Alternate route covers a different area and Partial route is a part of Full route. From this dataset, we selected sequences with overcast weather and less moving objects such as vehicles and pedestrians. Full 1 is chosen from 2015/02/10, Full 2,3,5 are from 2014/12/09 and Full 4 is from 2015/03/17. Alternate is from 2014/06/26 and Partial is from 2014/12/05. In the same way as for the indoor dataset, we measured the translational RMSE between the generated trajectory and the ground-truth data. We compared OmniDSO with DSO and monocular ORB-SLAM, the latter two using the pinhole camera model. We used the ORB-SLAM2 implementation (https://github.com/raulmur/ORB_SLAM2). Because the selected sequences do not contain loops, we can fairly compare VO and SLAM without turning off the loop closure of SLAM. We similarly ran the algorithm 5 times for each sequence per method and took the median. The results and the trajectory length are shown in Table VI. From the table, we observe that our method outperforms the other methods for all sequences. The performance difference tends to be more clear as the trajectory becomes longer. Visual results are shown in Fig. 11.

VII. CONCLUSIONS

In this paper, we introduced real-time Direct Sparse Odometry for omnidirectional cameras. We first incorporate the unified omnidirectional camera model into DSO. Depth estimation is performed by efficiently searching along the epipolar curve incrementally. Camera pose, point depth and

TABLE V

MEAN TIMING RESULTS (MS). AS THE KEYFRAME NUMBER REDUCES, THE COMPUTATIONAL COST BECOMES SMALLER.

	Normal DSO			Omni DSO		
N_f	7	5	3	7	5	3
Tracking	3.7	4.1	3.6	9.9	9.8	8.5
Mapping	53	43	30	63	54	36

TABLE VI

ABSOLUTE RMSE AND GENERATED TRAJECTORY LENGTH IN METERS. OUR OMNIDSO SHOWS THE SMALLEST RMSE.

	Full1	Full2	Full3	Full4	Full5	Alternate	Partial
ORB-SLAM	12.1	34.2	43.1	67.1	1.83	46.0	89.5
DSO	10.0	26.4	27.4	58.2	0.987	22.9	60.1
Ours	**9.30**	**24.5**	**26.9**	**45.7**	**0.822**	**21.6**	**50.3**
Length	736	1459	1554	1719	204	1003	2433

affine brightness parameters are jointly optimized to minimize photometric error within an active keyframe window. Then, we quantitatively evaluated the performance on 2 public benchmark datasets and demonstrated that our omnidirectional DSO yields better performance than other methods on the benchmark. We also qualitatively discussed the benefits of using a large field of view and quantitatively assessed the increase in robustness over using a pinhole camera model when reducing the number of keyframes in the DSO optimization window. Our omnidirectional DSO can make use of wide FoV fisheye camera images. Our combination of using a unified omnidirectional camera model and sparse windowed bundle-adjustment can outperform existing visual odometry methods. In future work, our method could be improved by adding global optimization and loop closure.

ACKNOWLEDGMENT

REFERENCES

[1] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *CVPR*. IEEE, 2004, pp. I–I.

[2] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 15–22.

[3] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Proc. of IEEE Int. Conf. on Computer Vision (ICCV)*, 2013.

[4] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. of IEEE ICRA*, 2007, pp. 10–14.

[5] 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.

[6] D. Caruso, J. Engel, and D. Cremers, "Large-scale direct SLAM for omnidirectional cameras," in *Proc. of IEEE/RSJ IROS*, 2015.

[7] J. Engel, T. Schoeps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. of ECCV*, 2014.

[8] J. Civera, A. J. Davison, and J. M. Montiel, "Inverse depth parametrization for monocular SLAM," *IEEE transactions on robotics*, vol. 24, no. 5, pp. 932–945, 2008.

[9] H. Jin, P. Favaro, and S. Soatto, "Real-Time 3-D Motion and Structure of Point-Features: A Front-End for Vision-Based Control and Interaction," in *Proc. of IEEE CVPR*, 2000.

[10] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.
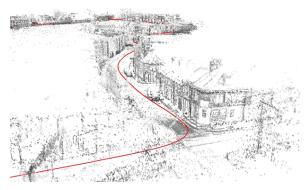

Fig. 11. Example qualitative result of OmniDSO on the Oxford Robotcar dataset. The dataset contains a large-scale outdoor trajectory through Oxford.

[11] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, "Structure from motion causally integrated over time," *PAMI*, vol. 24, no. 4, pp. 523–535, 2002.

[12] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *ISMAR*. IEEE, 2007, pp. 225–234.

[13] M. J. M. M. Mur-Artal, Raúl and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[14] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for RGB-D cameras," in *Proc. of IEEE ICRA*, 2013.

[15] T. Schöps, J. Engel, and D. Cremers, "Semi-dense visual odometry for ar on a smartphone," in *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*. IEEE, 2014, pp. 145–150.

[16] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 4628–4635.

[17] K. Tateno, F. Tombari, I. Laina, and N. Navab, "CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction," in *Proc. of IEEE CVPR*, 2017.

[18] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: an OpenSource SLAM System for Monocular, Stereo and RGB-D cameras," *CoRR*, vol. abs/1610.06475, 2016.

[19] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visualinertial odometry using nonlinear optimization," *IJRR*, vol. 34, no. 3, pp. 314–334, 2015.

[20] D. Scaramuzza and R. Siegwart, "Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles," vol. 24, no. 5, pp. 1015–1026, 2008.

[21] D. Gutierrez, A. Rituerto, J. Montiel, and J. J. Guerrero, "Adapting a real-time monocular visual slam from conventional to omnidirectional cameras," in *Proc. of the 11th OMNIVIS with Int. Conf. on Computer Vision (ICCV)*, 2011.

[22] S. Urban and S. Hinz, "MultiCol-SLAM - a modular real-time multi-camera slam system," *arXiv preprint arXiv:1610.07336*, 2016.

[23] C. Silpa-Anan, R. Hartley *et al.*, "Visual localization and loop-back detection with a high resolution omnidirectional camera," in *The 6th Workshop on Omnidirectional Vision*. Citeseer, 2005.

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

[25] C. Geyer and K. Daniilidis, "A unifying theory for central panoramic systems and practical implications," in *Proc. of ECCV*, 2000.

[26] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 1, pp. 3–15, 2017.

[27] P. Furgale, J. Rehder, and R. Siegwart, "Unified Temporal and Spatial Calibration for Multi-Sensor Systems ," in *Proc. of IEEE/RSJ IROS*, 2013, pp. 1280–1286.