Large-Scale Direct SLAM for Omnidirectional Cameras

David Caruso[1] and Jakob Engel[2] and Daniel Cremers[2]

***Abstract*— We propose a real-time, direct monocular SLAM method for omnidirectional or wide field-of-view fisheye cameras. Both tracking (direct image alignment) and mapping (pixel-wise distance filtering) are directly formulated for the unified omnidirectional model, which can model central imaging devices with a field of view well above 150°. This is in stark contrast to existing direct mono-SLAM approaches like DTAM or LSD-SLAM, which operate on rectified images, limiting the field of view to well below 180°. Not only does this allow to observe – and reconstruct – a larger portion of the surrounding environment, but it also makes the system more robust to degenerate (rotation-only) movement. The two main contribution are (1) the formulation of direct image alignment for the unified omnidirectional model, and (2) a fast yet accurate approach to incremental stereo directly on distorted images. We evaluated our framework on real-world sequences taken with a 185° fish-eye lens, and compare it to a rectified and a piecewise rectified approach.**

I. INTRODUCTION

Visual Odometry (VO) and Simultaneous Localization and Mapping (SLAM) are becoming increasingly important for robotics and mobile vision applications, as they only require optical cameras – which are cheap, light and versatile, and hence can easily be put into commodity hardware. A lot of research has been focussed around these topics throughout the last decade, with an particular focus on real-time systems – which for example can be used for autonomous control for example of UAVs [1], [2].

Most existing approaches are based on keypoints: Once keypoints are extracted, the images are abstracted to a collection of point-observations which are then used to compute geometrical information. This can be done in a filtering framework [3][4][5], or in a keyframe-based non-linear optimization framework [6], [7], [8]. This arguably has the advantage that a large part of the required work-load only is done once on keypoint extraction, such that remaining computational resources can be spent on enforcing large-scale geometric consistency (Bundle Adjustment), and outliers can be removed in a straight-forward way.

More recently, so-called direct approaches have gained in popularity: instead of abstracting the images to point-observations, they compute dense [9], or semi-dense [10] depth maps in an incremental fashion, and track the camera using direct image alignment. This has the advantage that much more information can be used, in particular information contained edges or densely textured surfaces. Further,

*The research leading to these results was supported by the BMBF within the Software Campus (NanoVis) No. 01IS12057

[1]David Caruso is with the Ecole polytechnique, Palaiseau, France david.caruso@polytechnique.edu

[2]Jakob Engel and Daniel Cremers are with the Technical University Munich engelj@in.tum.de, cremers@tum.de

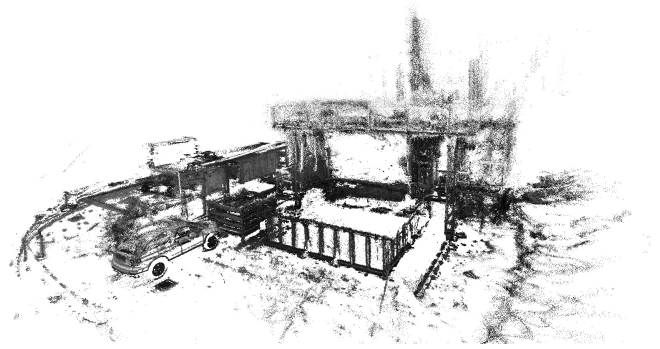

Fig. 1. **Top**: 3D reconstruction obtained in real-time with our approach, using a 185° fisheye lens. **Bottom**: Map of the trajectory and set of example keyframes, with associated color-coded inverse distance maps.

the generated map contains substantially more information about the environment, which can be used for obstacle-avoidance and path-planning.

What all these visual methods have in common, is that they rely on a sufficiently informative observed environment. In many practical cases however, this can be a very restrictive assumption: For example indoors where there are many untextured white walls, or in the presence of moving objects, large parts of the camera image can become uninformative for SLAM. This is especially true if the used camera only has a small field of view (FoV). On the other hand, the wider the field of view, the more likely that some part of the visible scene is well-suited for SLAM.

Nevertheless, most visual SLAM or VO systems are restrained to using a classical pinhole camera model. Often,

this is combined with a radial distortion model (such as the ATAN model used in PTAM). All these models can not directly be used for omnidirectional camera (with FoV of more than 180°). This is especially true for direct methods, which typically operate on rectified images – limiting the field of view to no more than 130°.

In this paper, we propose an extension of LSD-SLAM [10] to a generic omnidirectional camera model. The resulting method is capable of handling all types of central projection systems such as fish-eye and catadioptric cameras. We evaluate it on images captured with a fish-eye lens covering a FoV of 185°. We show that especially for trajectories which contain aggressive camera rotations, it outperforms in the previously presented algorithms, without losing its real-time capability.

A. Related Work

There is a range of related work regarding omnidirectional vision, in particular for robot and ground-vehicle localization. For instance [11] uses a catadioptric system to estimates the ego-motion of a vehicle, using direct photometric error minimization for rotation estimation – it is however restricted to planar motion. In [12], ransac point association for SIFT features is used for estimating translation and rotation, on a rig of 5 rectified cameras. Again, the system is restricted to planar motion. In [13] a multicamera rig is used to build a topological map based on appearance. In [14] an EKF-based SLAM system is adapted for omnidirectional cameras; In [15], the advantage of using omnidirectional cameras in this context is shown. The work of Meilland et. al. [16] is somewhat closer to ours, as it performs dense registration against multiple frames from a database of spherical images. They are augmented with distance information from an external sensor or stereo-vision. However the system is based on a priori learned database of georeferenced images and does not perform online SLAM.

B. Contribution and Outline

In this paper we explore the use of omnidirectional and fish-eye cameras for direct, large-scale visual SLAM. We propose two different camera model choices, which we integrate into the recently appeared LSD-SLAM [10] framework, and evaluate the resulting algorithm on real-world and simulated data. More precisely, the main contribution of this paper is two-fold: (1) We give a direct image alignment formulation operating on an omnidirectional camera model. (2) We derive an efficient and accurate approach to perform stereo directly on omnidirectional images, both for the piecewise rectification approach as well as natively on the Unified Omnidirectional Model. We intend to make the used datasets including ground-truth publicly available.

The paper is organized as follows: In Chapter II, we introduce a camera model as general projection function, and describe the three parametrized models considered in this paper: The *Pinhole Model* in Sec. II-A, an *Array of Pinhole Models* in Sec. II-B, and the *Unified Omnidirectional Model* in Sec. II-C. In Chapter III, we describe our omnidirectional direct SLAM method. We start by reviewing the LSD-SLAM pipe-line as introduced in [10]. We then detail how the two major steps that depend on the camera model – probabilistic, semi-dense depth estimation and direct image alignment – are adapted to operate in real-time on images from omnidirectional cameras. In Chapter IV, we evaluate the accuracy, robustness and runtime for the three different models on a both simulated and real-world data. Finally, in Chapter V, we summarize the results and line out future work.

Fig. 2. Camera Models : The same image, warped to fit the three projection models considered in this paper. While the Unified model and the piecewise rectified model can cover the full 185° field-of-view, the pinhole model shows significant distortion – when cropping the image to a field-of view of only 120° TODO diagonal or in x-direction?, as done for this figure.

II. CAMERA MODELS

In this chapter, we will lay out the three different parametric projection functions π considered in the paper: In Sec. II-A, we briefly review the well-known *Pinhole Model* and discuss its limitations. We then extend it to a more general *Array of Pinhole Models* allowing to cover the full viewing sphere in Sec. II-B. In Sec. II-C, we introduce the Unified Omnidirectional Model, which allows to model 360°-vision in closed-form.

Notation. We use bold, capital letters $\boldsymbol{R}$ to denote matrices, and bold, lower-case letters $\mathbf{x}$ for vectors. $\mathbf{u} = [u, v]^T \in \Omega \subset \mathbb{R}^2$ will generally denote pixel coordinates, where Ω denotes the image domain. $\mathbf{x} = [x, y, z]^T \in \mathbb{R}^3$ will be used for 3D point coordinates and $\tilde{\mathbf{x}} := [\mathbf{x}^T, 1]^T$ for the corresponding homogeneous point. $[\cdot]_i$ denotes the i'th row of a matrix / vector.

In the most general case, a camera model is a function $\pi \colon \mathbb{R}^3 \to \Omega$, which defines the mapping between 3D points $\mathbf{x}$ in the camera frame, and pixels $\mathbf{u}$ in the image. For lenses with negligible diameter, a common assumption is the *single viewpoint assumption*, i.e., that all light-rays pass through a single point in space – the origin of the camera frame $\mathcal{C}$. Hence the projected position of the point only depends on the direction of $\mathbf{x}$. We will use $\pi^{-1} \colon \Omega \times \mathbb{R}^+ \to \mathbb{R}^3$ for the function mapping pixels back to 3D, using their inverse distance d. Further, we define a directed orthonormal camera frame centred at $\mathcal{C}$ by fixing a privileged direction $\mathbf{z}$ (the *principal axis*) and two other orthogonal directions.

Note that the single viewpoint assumption allows transforming images from any camera model to any other, for the domain of visible points they have in common – this is generally referred to as *image rectification*, and is a frequently done pre-processing step, transforming the image

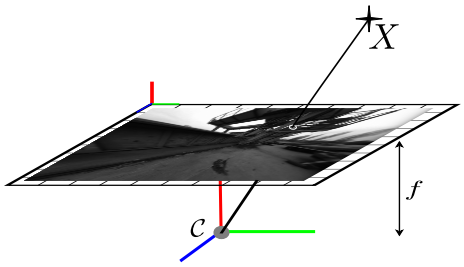

Fig. 3. Pinhole Model. A 3D point is directly projected onto the image plane through $\mathcal{C}$.

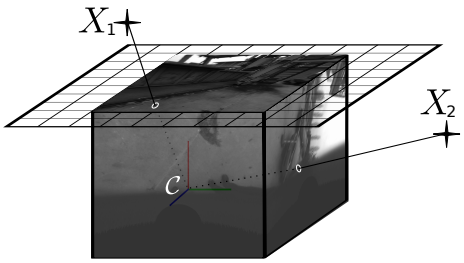

Fig. 4. Piecewise Pinhole Model. A 3D point is projected through the center of the camera on one of the image planes depending depending on the subspace it lies in, effectively forming a cube-shaped image plane. X_1 and X_2 are projected to different images Ω_1 and Ω_2.

to follow a more simple model e.g. by removing radial distortion. Given two projection functions π_1, π_2 and an image $I_1 \colon \Omega_1 \to \mathbb{R}$ taken with a camera π_1, we can compute the respective image $I_2 \colon \Omega_2 \to \mathbb{R}$ following projection π_2 as

$$I_2(u,v) = I_1(\pi_1(\pi_2^{-1}(u,v))) \tag{1}$$

This warping however introduces interpolation artefacts and can degrade the image quality, especially in areas where the angular resolution changes significantly.

A. Pinhole Model

The pinhole camera model is the most used camera model. The image is obtained by projecting each point onto a plane located at $z = 1$, followed by an affine mapping

$$\pi_p(\mathbf{x}) := \begin{bmatrix} f_x & 0 \\ 0 & f_y \end{bmatrix} \begin{bmatrix} x/z \\ y/z \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \tag{2}$$

where f_x, f_y are the focal lengths, and c_x, c_y is the principal point. It is schematically shown in Fig. 3.

This model is often used as the linearity of the projection function (in homogeneous coordinates) – and the fact that straight lines in 3D are projected to straight lines in the image – make it the most simple model choice to use. It however has the major drawback that it cannot model a wide field of view: The angular resolution decreases drastically towards the borders of the image, leading to a distorted image – an example is shown on the right in Fig. 8.

In order to make this model compatible to small radial distortions, a non-linear radial distortion function – often approximated polynomially – can be applied to the projected pixel coordinates. Still, the nature of a pinhole projection forbids points to lie behind the image plane, limiting the field of view to below 180°.

B. Array of Pinhole Camera

A straight-forward approach to extending the field of view is to use a camera model consisting of an array of several pinhole cameras, which have the same principal point but different orientations. The projection function $\pi_{mp}(\mathbf{x}) \colon \mathbb{R}^3 \to \cup_i \Omega_i$ is then given by piecewise rotation followed by pinhole projection, i.e.,

$$\pi_{mp}(\mathbf{x}) := \pi_{p_{i(\mathbf{x})}}(\mathbf{R}_{i(\mathbf{x})}\mathbf{x}) \tag{3}$$

where $i(\mathbf{x}) \colon \mathbb{R}^3 \to [1,k]$ segments the 3D space into k subspaces. While in general the segmentation and orientation of the associated cameras can be chosen arbitrarily, we choose to split $\mathbb{R}^3$ into six equally sized quadrants, forming a cube-shaped image plane. This has the advantage that $i(\mathbf{x})$ can be computed from binary comparisons on x, y and z, while the $\mathbf{R}_i$ correspond to orthogonal rotations.

While this model has a number of desirable properties – it is piecewise linear in homogeneous coordinates, simple to compute and offers reasonably homogeneous angular resolution – it does not fit natural lenses. In order to use it, incoming images have to be rectified in a preprocessing step. Further, the piecewise nature of the model causes discontinuities in the image space $\Omega = \cup_i \Omega_i$, complicating its use in practice.

C. Central Omnidirectionnal Camera: Unified Model

A number of different projection functions has been proposed in the literature for modelling and calibrating catadioptric and diotpric omnidirectionnal cameras. Desirable properties of such a function include (1) its capability to accurately describe a wide range of actual physical imaging devices, (2) the ease of parameter calibration and (3) the existence of a closed-form expression for the unprojection function π^{-1}. As this paper targets real-time direct SLAM, an additional criterion is the computational cost of projecting and unprojecting points, as well as the cost of evaluating the corresponding derivatives.

Accurate results were obtained by moving all non-linearities into a radially symmetric function, and identifying the first coefficients of its Taylor expansion [17]. While this approach can model every camera that fits the single viewpoint assumption, it lacks a closed-form unprojection function – and approximating it is computationally costly.

Instead, we use the model originally proposed in [18] for central catadioptric systems and extended in [19], [20] for a wider range of physical devices including fish-eye camera. The central idea behind this model is to concatenate two successive projections: The first one projects the point from the world onto a camera-centered unit sphere. The second one is an ordinary pinhole projection trough a center shifted along the z axis by $-\xi$. The model is described by a total of five parameters, f_x, f_y, c_x, c_y and ξ. The projection of a

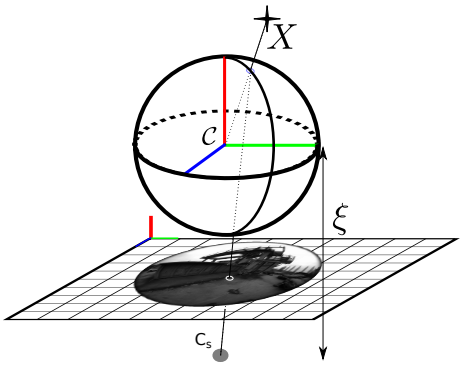

Fig. 5. Unified Model. A 3D point is first projected on the unit sphere, and then the image plane via a secondary, shifted camera center C_s.

point is computed as

$$\pi_u(\mathbf{x}) = \begin{bmatrix} f_x \dfrac{x}{z + \|\mathbf{x}\|\xi} \\ f_y \dfrac{y}{z + \|\mathbf{x}\|\xi} \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \tag{4}$$

where $\|\mathbf{x}\|$ is the euclidean norm of $\mathbf{x}$. The corresponding unprojection function can be computed in closed form, and is given by

$$\pi_u{}^{-1}(\mathbf{u}, d) = \frac{1}{d} \left(\frac{\xi + \sqrt{1 + (1-\xi^2)(\tilde{u}^2 + \tilde{v}^2)}}{\tilde{u}^2 + \tilde{v}^2 + 1} \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ \xi \end{bmatrix} \right), \tag{5}$$

where

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \end{bmatrix} = \begin{bmatrix} (u - c_x)/f_x \\ (v - c_y)/f_y \end{bmatrix}. \tag{6}$$

The major advantage of this model is the availability of an easy-to-compute projection and unprojection function. Note that for $\xi = 0$ it reduces to the pinhole model. In order to improve the generality of the model, we we combine it with a small radial-tangential distortion to correct lens imperfections – similar to the pinhole case, images are warped once in the beginning, to perfectly fit this model.

III. DIRECT OMNIDIRECTIONAL SLAM

In this Chapter, we describe our omnidirectional, large-scale direct SLAM system, which is based on LSD-SLAM [10]. First, in Sec. III-A we review the LSD-SLAM pipeline adapted to omnidirectional cameras. We then derive a direct image registration formulation for the unified camera model in Sec. III-B. In Sec. III-D, we show how – in this framework – stereo can be done efficiently on the unified (1) and piecewise rectified (2) model.

Notation. $D: \Omega_d \to \mathbb{R}^+$ will denote the inverse distance map of the current keyframe.With a slight abuse of notation, elements of $\mathfrak{se}(3)$ will directly be represented as 6-vector $\boldsymbol{\mu}$, and we use exp and log to associate an element of the lie algebra to the corresponding element of the lie group. We then define the composition operator $\circ$ as

$$\boldsymbol{\mu_1} \circ \boldsymbol{\mu_2} := \log\left(\exp(\boldsymbol{\mu}_1) \cdot \exp(\boldsymbol{\mu}_2)\right). \tag{7}$$

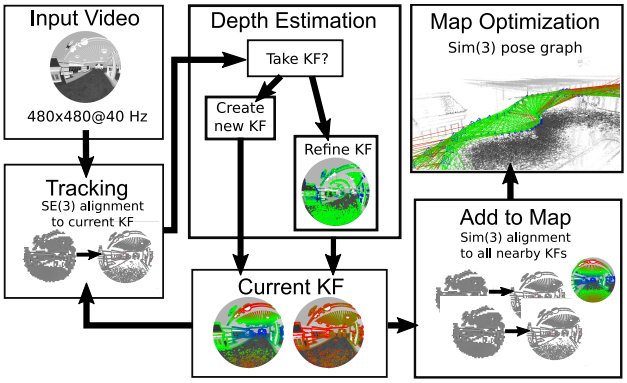

Fig. 6. Overview over the LSD-SLAM pipeline for omnidirectional Cameras. Tracking and depth estimation inherently depend on the camera model used, their omnidirectional versions and are detailed in Sec. III-B and III-D receptively.

As a shorthand, we use $\mathbf{R}_{\boldsymbol{\mu}}$ and $\mathbf{t}_{\boldsymbol{\mu}}$ to denote the corresponding rotation matrix and translation vector of a transformation, and $[\cdot]_i$ to extract the i'th row of a matrix/vector.

A. Method Overview

Our method continuously builds and maintains a pose-graph of keyframes. Each keyframe contains a probabilistic semi-dense inverse *distance* map, which maintains a Gaussian probability distribution over the inverse distance for all pixels which have sufficient intensity gradient. It is estimated over time by filtering over a large number of small-baseline stereo comparisons. In turn, new images – as well as loop-closure constraints – are computed using direct image alignment. Note that in contrast to [10], we use the inverse distance $d = \|\mathbf{x}\|^{-1}$ instead of depth, such that we can model points behind the camera. An overview is shown in Fig. 6.

1) SE(3) Tracking: When a new camera frame is captured, its rigid-body pose relative to the closest keyframe is tracked using direct image alignment, which will be described in III-B.

2) Probabilistic Distance Map Estimation: Keyframes are selected at regular intervals, based on the moved distance to the previous keyframe (relative to its mean inverse distance), as well as the relative overlap. For each keyframe, an inverse distance map is initialized by propagating the inverse distance map from its immediate predecessor. Subsequently, it is updated – and extended to new regions – by incorporating information obtained from many small-baseline stereo comparisons. This step will be described in more detail in Sec III-D.

3) Scale-drift aware Pose-Graph optimization on Sim(3): In the background, we continuously perform pose graph optimization between all keyframes, and attempt to find new constraints between keyframes which are likely to overlap. Constraints are expressed as similarity transforms to account for scale-drift – more details on this part can be found in [10].

4) Initialization of the SLAM system: The system is initialized with a random depth map with mean one and a large covariance – this generally converges to a good

estimate, as long as the camera motion within the first few seconds is not too degenerate.

B. Omnidirectional Direct Image Alignment on SE(3)

Every new frame I_{new} is tracked relative to the closest keyframe I_{Kf} with associated inverse distance map D_{Kf} by direct minimization of the photometric error, defined as

$$E^{\text{frame}}(\boldsymbol{\mu}) := \sum_{\mathbf{u}\in\Omega_\mathbf{d}} \rho\left(\left[\frac{r^I_\mathbf{u}(\boldsymbol{\mu})}{\sigma_{r^I_\mathbf{u}(\boldsymbol{\mu})}}\right]^2\right), \tag{8}$$

where ρ denotes the robust Huber norm, and with

$$r^I_\mathbf{u}(\boldsymbol{\mu}) = I_{\text{Kf}}(\mathbf{u}) - I_{\text{new}}\left(\pi(\omega(\boldsymbol{\mu},\mathbf{u}))\right) \tag{9}$$

$$\omega(\boldsymbol{\mu},\mathbf{u}) = \mathbf{R}_{\boldsymbol{\mu}}\pi^{-1}(\mathbf{u}, D_{\text{Kf}}(\mathbf{u})) + \mathbf{t}_{\boldsymbol{\mu}}. \tag{10}$$

The function ω unprojects a point, and and transforms it by $\boldsymbol{\mu}$. As in [10], the residuals are normalized with their propagated inverse distance variance.

This weighted least-squares problem is then minimized in a coarse-to-fine scheme using the iteratively re-weighted Levenberg-Marquad algorithm in a left-compositional formulation: In each iteration, we solve for a left-multiplied increment

$$\boldsymbol{\delta\mu}^{(k)} = \left(\mathbf{J}^T\mathbf{W}\mathbf{J} + \lambda\text{diag}(\mathbf{J}^T\mathbf{W}\mathbf{J})\right)^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r}, \tag{11}$$

where $\mathbf{r} = [\mathbf{r}^I_{\mathbf{u}_1}\dots\mathbf{r}^I_{\mathbf{u}_n}]^T$ is the stacked residual vector and $\mathbf{W}$ a diagonal matrix containing the weights. $\mathbf{J}$ is the $n\times 6$ Jacobian of the stacked residual vector evaluated at $\boldsymbol{\mu}^{(k)}$:

$$\mathbf{J} = \frac{\partial\mathbf{r}(\boldsymbol{\epsilon}\circ\boldsymbol{\mu}^{(k)})}{\partial\boldsymbol{\epsilon}} \tag{12}$$

which is then left-multiplied on the current estimate

$$\boldsymbol{\mu}^{(k+1)} = \boldsymbol{\delta\mu}^{(k)}\circ\boldsymbol{\mu}^{(k)}. \tag{13}$$

Using the chain rule, each 1×6 row $\mathbf{J}_\mathbf{u}$ of the Jacobian can be decomposed into three parts

$$\mathbf{J}^{\text{fwd}}_\mathbf{u} = -\mathbf{J}_{I_{\text{new}}}\big|_\pi\mathbf{J}_\pi\big|_\omega\mathbf{J}_\omega\big|_{\boldsymbol{\mu}}, \tag{14}$$

where

- $\mathbf{J}_\omega\big|_{\boldsymbol{\mu}^{(k)}}$ is a 3×6 Jacobian, denoting the left-compositional derivative of the transformed point, evaluated at $\boldsymbol{\mu} = \boldsymbol{\mu}^{(k)}$

$$\mathbf{J}_\omega\big|_{\boldsymbol{\mu}} = \frac{\partial\omega(\boldsymbol{\epsilon}\circ\boldsymbol{\mu},\mathbf{u})}{\partial\boldsymbol{\epsilon}}. \tag{15}$$

- $\mathbf{J}_\pi\big|_{\boldsymbol{\omega}}$ is the 2×3 Jacobian of the projection function π evaluated at $\omega = \omega(\boldsymbol{\mu}^{(k)},\mathbf{u})$.
- $\mathbf{J}_{I_{\text{new}}}\big|_\pi$ is the 1×2 intensity gradient of the new image, evaluated at point $\pi = \pi(\omega(\boldsymbol{\mu}^{(k)},\mathbf{u}))$.

Notice how the evaluation point of each of these Jacobians depends on $\boldsymbol{\mu}^{(k)}$, hence everything has to be re-evaluated in each iteration. In practice, the computational cost is the dominated by this evaluation – which is especially true in our case, as for the unified model the projection, and hence its derivative $\mathbf{J}_\pi\big|_{\boldsymbol{\omega}}$ is much more complex.

To avoid this, we use an *inverse compositional* formulation – a trick that is well known in the literature [21]: In each iteration, instead of applying the increment to the points in the reference frame, its inverse is applied to the points in the keyframe. That is, instead of linearising

$$I_{\text{Kf}}(\mathbf{u}) - I_{\text{new}}(\pi(\omega(\boldsymbol{\epsilon}\circ\boldsymbol{\mu}^{(k)},\mathbf{u}))), \tag{16}$$

with respect to $\boldsymbol{\epsilon}$, we linearize

$$I_{\text{Kf}}(\pi(\omega(\boldsymbol{\epsilon},\mathbf{u}))) - I_{\text{new}}(\pi(\omega(\boldsymbol{\mu}^{(k)},\mathbf{u}))). \tag{17}$$

The Jacobian now becomes

$$\mathbf{J}^{\text{bkwd}}_\mathbf{u} = \mathbf{J}_{I_{\text{Kf}}}\big|_\pi\mathbf{J}_\pi\big|_\omega\mathbf{J}_\omega\big|_\mathbf{0}, \tag{18}$$

with $\omega = \omega(\mathbf{0},\mathbf{u})$ and $\pi = \pi(\omega(\mathbf{0},\mathbf{u}))$. It is thus independent of $\boldsymbol{\mu}^{(k)}$. This allows us to pre-compute it once per pyramid level, saving much of the required computations. Note that we still have to re-evaluate the outer product $\mathbf{J}^T\mathbf{W}\mathbf{J}$ on each iteration, as the weight matrix changes. The inverse of the resulting update is then right-multiplied onto the current estimate, i.e.,

$$\boldsymbol{\mu}^{(k+1)} = \boldsymbol{\mu}^{(k)}\circ(-\boldsymbol{\delta\mu}^{(k)}). \tag{19}$$

C. Omnidirectional Direct Image Alignment on Sim(3)

In monocular SLAM, the absolute scale is not observable and drifts over time – which has to be taken into account when finding loop-closures. As in [10], we use Sim(3) image alignment between keyframes, to estimate not only their relative pose, but also the scale difference between their inverse distance maps. This is done by introducing an additional error term – the geometric error – which penalizes differences in inverse distance. The energy function for aligning $(I_{\text{K1}}, D_{\text{K1}})$ and $(I_{\text{K2}}, D_{\text{K2}})$ thus becomes

$$E^{\text{Kf}}(\boldsymbol{\mu}) := \sum_{\mathbf{u}\in\Omega_\mathbf{d}} \rho\left(\left[\frac{r^I_\mathbf{u}(\boldsymbol{\mu})}{\sigma_{r^I_\mathbf{u}(\boldsymbol{\mu})}}\right]^2 + \left[\frac{r^D_\mathbf{u}(\boldsymbol{\mu})}{\sigma_{r^D_\mathbf{u}(\boldsymbol{\mu})}}\right]^2\right), \tag{20}$$

where $\boldsymbol{\mu}\in\mathfrak{sim}(3)$, and

$$r^I_\mathbf{u}(\boldsymbol{\mu}) = I_{\text{K1}}(\mathbf{u}) - I_{\text{K2}}\left(\pi(\omega_s(\boldsymbol{\mu},\mathbf{u}))\right) \tag{21}$$

$$r^D_\mathbf{u}(\boldsymbol{\mu}) = \|\omega_s(\boldsymbol{\mu},\mathbf{u})\|^{-1} - D_{\text{K2}}\left(\pi(\omega_s(\boldsymbol{\mu}.\mathbf{u}))\right). \tag{22}$$

Note that we now optimize over relative scale as well, and hence have to apply a similarity warp, defined as

$$\omega_s(\boldsymbol{\mu},\mathbf{u}) = s_{\boldsymbol{\mu}}\mathbf{R}_{\boldsymbol{\mu}}\pi^{-1}(\mathbf{u}, D_{\text{K2}}(\mathbf{u})) + \mathbf{t}_{\boldsymbol{\mu}}, \tag{23}$$

where $s_{\boldsymbol{\mu}}$ is the scaling factor of $\boldsymbol{\mu}$. Note that in contrast to [10], this residual now penalizes differences in inverse distance. Again, we apply statistical normalization based on the propagated variances as in [10]. For tracking Sim(3)-constraints, we use a forward-compositional formulation.

We further note that as in [10], the approximated Hessian $(\mathbf{J}^T\mathbf{W}\mathbf{J})^{-1}$ of the last iteration can be interpreted as covariance on a left-multiplied increment on $\boldsymbol{\mu}$, and is used in the subsequent pose-graph optimization.

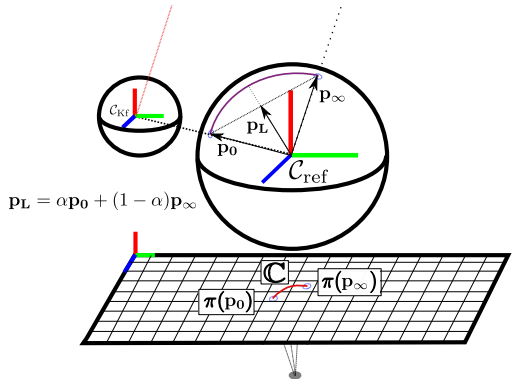

Fig. 7. Non-Rectified Stereo Matching: We efficiently browse the epipolar curve in the image $\mathbf{u}_L$ using a parametric equation. It is obtained by projecting the line connecting $\mathbf{p}_0$ and $\mathbf{p}_\infty$ on the unit sphere around the camera center.

D. Semi-Dense Depth Map Estimation

Once a frame is registered to a keyframe, stereo matching is performed to refine the keyframe distance map D_{Kf}. As matching cost we use the *sum of squared differences* (SSD) over five equidistant pixels along the epipolar line. If a prior exists, the epipolar search is constrained to the interval $[d - 2\sigma_d, d + 2\sigma_d]$. This greatly improves efficiency and minimizes the probability of finding an incorrect match, as in practice only very short line segments have to be searched. Subsequently we refine the found match to sub-pixel precision.

Similar to [22], each new measurement is fused into the existing depth map. Measurement variances σ_m^2 are obtained using the geometric and photometric error, as derived in [22]. Finally, we smoothe the inverse distance map and remove outliers.

1) Non-Rectified Stereo: When performing stereo on the unified model, epipolar lines are not in fact lines but curves. More precisely, Geyer et al. showed that these epipolar curves are conics [18], as they are the pinhole-projection of a geodesic on the unit sphere, as visualized in Fig. 7. We here present a general method to incrementally and efficiently compute points along the epipolar curve, at a constant step-size of 1 px: While this is trivial for straight lines, it is not straight-forward for the general case of epipolar curves.

We first define the two points $\mathbf{p}_0, \mathbf{p}_\infty \in \mathbb{R}^3$ on the unit sphere around the projective center $\mathcal{C}_{\text{ref}}$, which correspond to the maximum and minimum inverse distance of the search interval d_{max},d_{min}:

$$\mathbf{p}_0 := \pi_s(\mathbf{R}\pi_u^{-1}(\mathbf{u}, d_{\text{max}}) + \mathbf{t}) \tag{24}$$

$$\mathbf{p}_\infty := \pi_s(\mathbf{R}\pi_u^{-1}(\mathbf{u}, d_{\text{min}}) + \mathbf{t}). \tag{25}$$

Here, π_s projects a point onto the unit sphere, π_u^{-1} is the unprojection function of the unified model (5), and $\mathbf{u}$ is the pixel in I_{Kf} we are trying to match. We then express the straight line between these points as

$$\mathbf{p}_L(\alpha) = \alpha\mathbf{p}_0 + (1-\alpha)\mathbf{p}_\infty, \tag{26}$$

for $\alpha \in [0, 1]$. This also gives a parametric expression for the epipolar curve in I_{ref} as

$$\mathbf{u}_L(\alpha) := \pi_u(\mathbf{p}_L(\alpha)). \tag{27}$$

Note that we apply the full unified projection function, which first projects a point on $\mathbf{p}_L$ onto the geodesic, and then into the image. This is visualized in Fig. 7.

Starting at $\mathbf{u}_L(0)$, we then browse the epipolar curve by incrementing α. A step-size of 1 pixel is enforced by using a first-order Taylor expansion of $\mathbf{u}_L$, and choosing the increment in α as

$$\delta\alpha = \left\| \mathbf{J}_{\mathbf{u}_L}\big|_\alpha \right\|^{-1}, \tag{28}$$

which we re-evaluate for each increment. Note that this method is independent of the shape of the epipolar curve, and hence can be used for any central camera model. Nevertheless, it is much more expensive than browsing a straight line, as each point is projected individually. In LSD-SLAM however, the search interval is always small, as either a good prior is available, or the pixel has just been initialized and hence the baseline is small.

2) Pre-Rectified Stereo: For a large disparity search range, the above method can become very costly since it requires re-evaluation of the projection function for each point. Thus, the valid question arises whether piecewise rectification of the input image as described in Sec.II-B, followed by straight-forward line-browsing would be faster. For this we determined suitable values for the focal lengths f_x and f_y of each pinhole camera individually, minimizing the change in angular resolution at each point in the image. An example is shown in Fig. 8: Still, some distortion is clearly visible, note for example how the checker-board shape is altered. Further, we extend the rectified images by extending their visible field by 20 pixel, which is not displayed in the figure. We then perform line-stereo the same way as is done in [22]. In Sec. IV we will compare these two approaches regarding accuracy and efficiency.

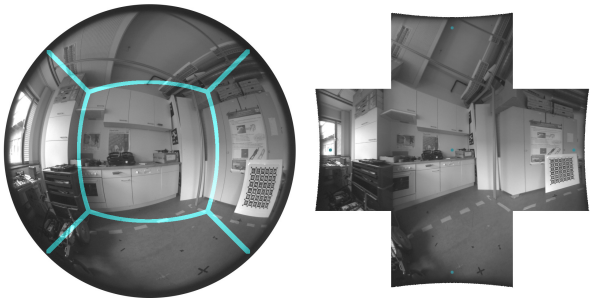

Fig. 8. Piecewise rectification: Example of fish-eye camera rectification. The borders are still distorted, as it is clearly visible on the checker-board, which leads to interpolation artefacts or blur.

IV. RESULTS

In this Chapter, we evaluate our algorithm regarding accuracy and computational requirements on both synthetic and real data. We first describe the experimental setup in. IV-A and IV-B. We then evaluate the accuracy and the computational requirements in Sec. IV-C and IV-D respectively.

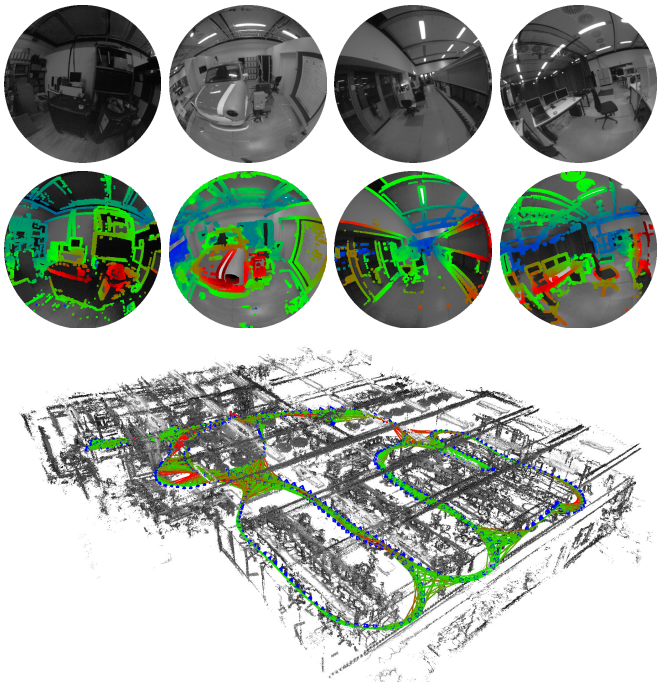

Fig. 9. Reconstruction of *T5* sequence. **Top**: Color-coded inverse distance maps. Note how we can obtain geometry for the full 185° field of view. **Bottom**: Final point-cloud. This corresponds to the right plot in Fig. 10

A. Hardware Setup

For real data experiment, we use a global shutter usb3 camera equipped with a 185 FoV fish-eye lens. The ξ parameter for this system has been estimated to 2.06 by off-line calibration, using the *Kalibr* toolbox [23]. Images are cropped and scaled to a 480×480 region centered around the principal point. We recorded a number of trajectories with rapid, handheld motion, including quick rotation - an overview over one of the sequences can be seen Fig. 9. We also show two of the sequences (T2 and T5) in the attached video. For ground truth acquisition, we use a motion capture system which covers an area of approximately $7 \times 12\,\text{m}$ – as some of the trajectories leave this area, we only compute errors on the part for which ground truth data is available. The synthetic data was generated using the ROS gazebo simulator, modified to have as extra output the synchronised pose, 185° images, and distance ground truth. The movement is slower on this dataset and mimics that of a quadrotor. For comparison with a pinhole model, we also synthesize a sequence of rectified images, artificially cropping the field of view to 100° horizontally and vertically. We intend to make the dataset including ground truth publicly available.

B. Evaluated parameters

We evaluate the effect of three different parameters:

- *The camera model:* We use either the unified omni-directional model (*Uni*, Sec. III-D.1), or a piecewise rectified model (*Multi*, Sec. III-D.2). As baseline, we use the cropped & rectified video with a pinhole model (*Pin*).
- *The input resolution:* We use either an input resolution of 480×480 (*Full*) or 240×240 (*Half*).

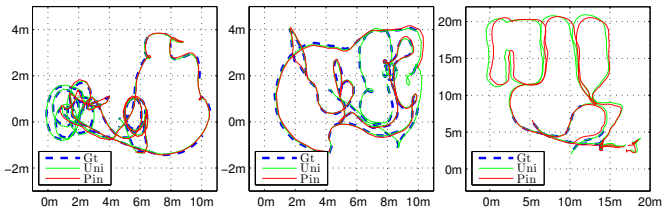

Fig. 10. Horizontal position for *T2*, *T3* and *T5*. The red line shows the result of Uni-Full-0, the green line that of Pin-Full-0, and the blue dotted line the ground truth where available. For *T2* and *T3*, the pinhole version is lost for a large portion of the trajectory, as they include fast rotations which cannot be tracked well with the cropped field of view. For *T5* the trajectory is correctly reconstructed with both camera models. But the accuracy is significantly better for the unified model (see Tab.II; the final pointcloud is shown in Fig. 9.

- *Resolution used for tracking:* We choose to stop the coarse to fine approach in tracking either at the input image (level 0 of the image pyramid) or at the first octave of it (level 1), allowing to speed-up tracking significantly, while maintaining most of the accuracy.

All the experiments were conducted on a Intel i7 CPU, on a commercially available laptop.

C. Accuracy comparison

In order to assess the accuracy of our method, we measure the translational root mean square error (RMSE) of the final position of all keyframes, after 7DoF alignment with the ground-truth. Because of the hard real-time constraint and the high frame-rate of the camera, frames may be dropped and different frames will be selected as keyframes – potentially having a significant effect on the result. We therefore average the RMSE of five runs. The results are shown in Tab. II and some representative visual result are shown in Fig. 10; also see the attached video.

Two things can be observed: First, results obtained with the omnidirectional camera clearly outperform the pinhole model. This shows that our algorithm can benefit from additional information in the image due to an increased field of view – the in some cases very significant difference is not surprising, as the recorded trajectories contain large amounts of rotation, which is very challenging for a normal camera.

The other observation is also expected: A higher resolution gives consistently better results than a lower resolution, although not by much. Interestingly, both half resolution omnidirectional methods outperform the full resolution pinhole model, showing that, at least in challenging scenes, a larger field of view is more important than high resolution. An example of 3D reconstruction with *half* and *full* resolution is given for the synthetic scene in Fig. 11.

D. Timing measurement

Table I shows the measured average time taken by tracking and mapping. These times are measured on the same dataset as used for the accuracy assessment, and are in millisecond. These results shows that our distorted stereo matching algorithm is slightly more efficient than the multi rectified version. Thisis due to the rectification required

TABLE I

MEAN TIMING RESULTS (MS)

	480×480			240×240			160×160		
	Mul	Uni	Pin	Mul	Uni	Pin	Mul	Uni	Pin
Mapping	31	28	20	11	8	7	-	-	-
Tracking	24	24	17	10	10	6	3	3	2.2

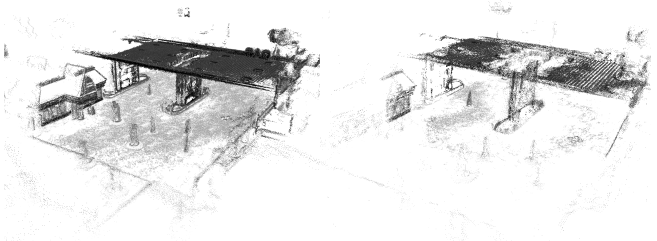

Fig. 11. Reconstruction evaluation. Final pointcloud obtained on S1 trajectory for two different resolutions (left 480×480, right 240×240): The resolution has a very sensitive impact on the completeness and accuracy of the 3D reconstruction.

beforehand, and the fact that almost always, the browsed epipolar segments do not exceed a couple of pixels in length. Real time is easily achieved since each frame can be tracked at least 40 Hz, and mapped at more than 30 Hz for a 480×480 image.

V. CONCLUSION

We proposed in this paper a direct, semi-dense monocular SLAM system for omnidirectional cameras. Based on two different omnidirectional camera models, our system allows to directly use a wide range of classical dioptric or catadioptric imaging systems. The contribution of this paper is two-fold: (1) we explicitly formulate a camera model independent registration algorithm and (2) derived a generic, accurate, and efficient way to perform stereo, based on a parametric equation of the epipolar curves. We integrated these ideas into the LSD-SLAM framework and ran the algorithm in real-time on a number of videos captured by a 185° fish-eye camera. We measure both an improvement of the accuracy of the localization and of its robustness to strong rotational movement compared to a standard camera. We also observe that even at relatively low resolutions (240×240), the localization accuracy surpasses the accuracy obtained when using a pinhole model, with a cropped field of view.

TABLE II

ABSOLUTE RMSE IN METERS

	T1	T2	T3	T4	T5	S1	S2
Mul-Full-0	**0.0493**	**0.0656**	**0.0456**	**0.0424**	**0.0535**	**0.0208**	**0.0484**
Mul-Full-1	0.0491	0.0699	0.0475	0.0479	0.0600	0.0387	0.0895
Mul-Half-0	0.0765	0.0966	0.0554	0.0546	0.0849	**0.0209**	**0.0433**
Mul-Half-1	0.0756	0.0977	0.0650	0.0952	0.1211	0.0345	0.1144
Uni-Full-0	0.0531	**0.0506**	**0.0463**	**0.0454**	**0.0358**	0.0340	0.0492
Uni-Full-1	**0.0508**	0.0634	0.0497	0.0514	0.0544	0.0429	0.0728
Uni-Half-0	0.0845	0.0731	0.0569	0.0588	0.0684	0.0382	0.0602
Uni-Half-1	0.1856	0.0837	0.0598	0.0730	0.1236	0.0428	0.0709
Pin-Full-0	0.5784	0.2282	0.0832	0.6049	0.5498	0.0474	0.7016
Pin-Full-1	0.6445	0.1526	0.0724	1.9756	0.9423	0.0861	0.6749
Pin-Half-0	1.1729	0.7301	0.6022	0.0863	1.1555	0.1297	1.5106
Pin-Half-1	1.5125	0.8351	0.6005	0.0941	2.3820	0.1685	1.1376

REFERENCES

[1] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter," in *IROS*, 2012.

[2] M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart, "Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments," in *ICRA*, 2011.

[3] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 29, no. 6, pp. 1052–1067, 2007.

[4] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

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

[6] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *ISMAR*, 2007.

[7] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, "Keyframe-based visual-inertial slam using nonlinear optimization," in *Proceedings of Robotics: Science and Systems*, 2013.

[8] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual SLAM," in *ICCV*, 2011.

[9] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *ICCV*, 2011.

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

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

[12] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," in *IROS*, 2008.

[13] C. Silpa-Anan and R. Hartley, "Visual localization and loop-back detection with a high resolution omnidirectional camera," in *Workshop on Omnidirectional Vision*, 2005.

[14] D. Gutierrez, A. Rituerto, J. Montiel, and J. J. Guerrero, "Adapting a real-time monocular visual slam from conventional to omnidirectional cameras," in *ICCV Workshops*, 2011.

[15] A. Rituerto, L. Puig, and J. Guerrero, "Comparison of omnidirectional and conventional monocular systems for visual slam," *10th OMNIVIS with RSS*, 2010.

[16] M. Meilland, A. I. Comport, and P. Rives, "A spherical robot-centered representation for urban navigation," in *IROS/RSJ*, 2010, pp. 5196–5201.

[17] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A flexible technique for accurate omnidirectional camera calibration and structure from motion," in *ICVS*, 2006.

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

[19] X. Ying and Z. Hu, "Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model," in *ECCV*, 2004.

[20] J. P. Barreto, "Unifying image plane liftings for central catadioptric and dioptric cameras," in *Imaging Beyond the Pinhole Camera*. Springer, 2006, pp. 21–38.

[21] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *IJCV*, vol. 56, no. 3, pp. 221–255, 2004.

[22] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *ICCV*, 2013.

[23] J. Maye, P. Furgale, and R. Siegwart, "Self-supervised calibration for robotic systems," in *Proc. of the IEEE Intelligent Vehicles Symposium (IVS)*, June 2013.