A System for Dense Monocular Mapping with a Fisheye Camera

Louis Gallagher[1], Ganesh Sistu[2], Jonathan Horgan[2] and John B. McDonald[1*†]

Abstract

We introduce a novel dense mapping system that uses a single monocular fisheye camera as the sole input sensor and incrementally builds a dense surfel representations of the scene's 3D geometry. We extend an existing hybrid sparse-dense monocular SLAM system, reformulating the mapping pipeline in terms of the Kannala-Brandt fisheye camera model. Each frame is processed in its original undistorted fisheye form, with no attempt to remove distortion. To estimate depth, we introduce a new version of the PackNet depth estimation neural network adapted for fisheye inputs. We reformulate PackNet's multi-view stereo self-supervised loss in terms of the Kannala-Brandt fisheye camera model. To encourage the network to learn metric depth during training, the pose network is weakly supervised with the camera's ground-truth inter-frame velocity. To improve overall performance, we additionally provide sparse depth supervision from dataset LiDAR and SICK laser scanners. We demonstrate our system's performance on the real-world KITTI-360 benchmark dataset. Our experimental results show that our system is capable of accurate, metric camera tracking and dense surface reconstruction within local windows. Our system operates within real-time processing rates and in challenging conditions. We direct the reader to the following video where the system can be seen in operation: https://youtu.be/Y-9q_wfqocs.

1. Introduction & Background

A robot's ability to compute a map of its surroundings, and simultaneously localise itself within that map, is foundational to its ability to act autonomously. Path planning, obstacle avoidance, environment interaction or any task that requires an autonomous robot to reason about its location in the world or the structure of its surroundings ultimately depends on this ability to perform simultaneous localisation and mapping (SLAM). As a robot moves through an environment, the robot's SLAM system integrates local sensor measurements into a geometric map and an estimate of the global trajectory of the robot. Cameras have long been considered as a particularly useful primary, and, often sole sensor for robot SLAM. A single monocular camera with a wide viewing angle offers many potential benefits. Such sensors are typically small, inexpensive, energy efficient, can be deployed in a diverse range of environments, such as in underwater [1] and space applications [2], and produce rich measurements at high frame rates. Current research on monocular depth estimation and object detection indicates the potential for cameras to either match or exceed the performance of LiDAR on these tasks [3,4]. Visual SLAM systems continue to show improved performance in scenarios where a LiDAR would typically be used. In particular, the hybrid sparse-dense monocular SLAM system of [5] has shown that it is possible to perform metric, dense visual SLAM in outdoor automotive applications with a single monocular camera as the only input sensor.

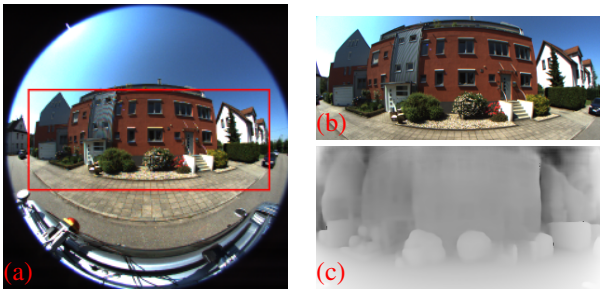


Figure 1. **Fisheye inputs to our system sequence.** Images from [6]. (a) shows the original uncropped fisheye image (1400×1400). (b) shows the cropped region used by our system. We crop the image to remove the car, camera housing and the sky (c) shows the live depth estimate of (b).

In this paper, we present a fisheye monocular dense visual mapping system by extending the framework from [5] to use large FOV fisheye imagery. Fisheye cameras have become increasingly popular given their wide field of view allowing a much larger portion of the scene to remain in view of the camera at any one time. Surround view sensing is essential in applications where sensory blind spots are a safety hazard [7]. However, increasing the FOV of an imaging device while holding all other characteristics fixed

*[1]Louis Gallagher and John B. McDonald are with the Department of Computer Science, Maynooth University, Co Kildare, Ireland firstname.lastname@mu.ie

†[2]Ganesh Sistu and Jonathan Horgan are with Valeo Vision Systems, Tuam, Ireland firstname.lastname@valeo.com

decreases the angular resolution of a pixel and increases the level of distortion in the image. When it comes to localisation and mapping, the loss of angular resolution and degree of distortion can have an effect on performance [8]. However, when taking a holistic view of a robot platform, such as an autonomous car, the benefits of the increased FOV to downstream safety critical modules may outweigh the negative impact on SLAM performance. Thus there is a strong need for a robust dense SLAM system that can cater for fisheye camera geometries. In this respect, the original hybrid system of [5] is fundamentally limited by its use of the pinhole camera model, which cannot accurately model fisheye imagery. In order to induce a large receptive field, and project all the light within it onto a flat sensor plane, fisheye lenses introduce distortions which are not well captured by this model. A plethora of camera models that follow different projections and more accurately describe fisheye cameras have been proposed [9]. In this work, we incorporate the Kannala-Brandt camera model [10] into the system to allow it to take as input frames from a single fisheye camera and to incrementally build a dense, metric $3D$ surface reconstruction.

Several SLAM systems have been presented that leverage alternate camera models to allow the use of fisheye and omnidirectional camera imagery. In [11] the direct whole image alignment camera tracker and depth map filtering of LSD-SLAM [12] are reformulated in terms of the unified omnidirectional camera model of [13]. However, the system only estimates a semi-dense map. Similar to [11], [14] implement an omnidirectional extension to DSO [15], reformulating the underlying optimisation in terms of the unified camera. [14] is a visual odometry system, not a SLAM system, and so it does not perform loop closure detection and correction. [16] uses motion stereo to predict depth frames which are then integrated into a truncated signed distance function. However, the system does not include a loop closure mechanism which is vital for large-scale long-term mapping and tracking. [17] uses a multiple-view sweeping planes-based algorithm to estimate a dense depth map for a reference camera in a multi-camera rig. Dynamic objects are detected and removed to prevent trailing artefacts from being integrated into the map. Only a local volume of space around the car is held in the map at any one point in time and no loop closure mechanism is included. [18] presents an omnidirectional dense SLAM system where a sensor platfrom consisting of four 220° FOV cameras is used. The system uses a light weight omnidirectional MVS depth estimation neural network, based on [19], to estimate a 360° depth map for each frame. Each frame is then tracked using ROVO [20]. Platform poses and depth maps are then fused into a dense TSDF map. Loop closures are identified using a feature-based vocabulary tree where a geometric consistency check verifies proposed matches. Once identified, loops are closed via pose-graph optimisation. In ORB-SLAM3 [21], the authors present a sparse visual SLAM framework that supports operation with fisheye cameras. The feature-based approach of ORB-SLAM3 lends itself to robust camera tracking during fast camera motion and in challenging conditions, such as variable lighting. We use ORB-SLAM3's feature-based camera tracking capabilities to estimate the frame-to-frame motion of the camera in our system. We also use ORB-SLAM3's loop closure mechanism to bootstrap a deformation-graph-based correction of the dense surfel map during loop closure. However, as we are interested in building dense maps in real-time we go beyond ORB-SLAM3 and its low resolution reconstruction of the scene's geometry.

In contrast to these works, we focus on monocular dense mapping with loop closures. Our system uses a single fisheye camera where the incoming camera frames are integrated into a dense global surfel map. We avoid undistorting the incoming imagery and instead operate directly

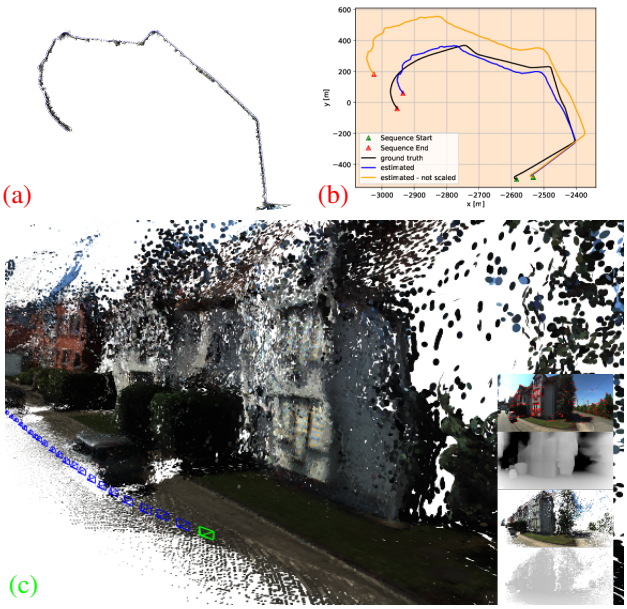


Figure 2. **Trajectory and dense reconstruction of a sequence from [6].** (a) shows the final surfel model produced by our system. The model contains approximately $7.5m$ surfels. (b) shows the estimated trajectory (projected onto the xy plane) alongside the ground truth trajectory. (c) shows an up-close view of the reconstruction as the vehicle passes through an intersection contained in (a). Inset in (c), from top to bottom: the current live image, the corresponding estimated depth map from the network, the model predicted RGB image and the model predicted depth map. Note that these results were captured using an accurate high resolution version of our network that allows our SLAM system to operate at approximately $2hz$. We trained a slim version of the network that allows the system to operate at $5hz$.

on the distorted fisheye images. One of the main benefits of fisheye images is their increased FOV which inevitably gets diminished during undistortion procedures. For every frame, our system detects and, if necessary, closes, loops in the vehicle's trajectory. To summarise, our contributions include:

- **Fisheye PackNet**: Reformulated PackNet training procedure for Fisheye images. Including velocity supervision, sparse LiDAR supervision, and Kannala-Brandt fisheye model for multi-view stereo loss function.
- **Generic Camera Model**: Redesigned ElasticFusion's mapping pipeline to be generic (in the software engineering sense of the word) with respect to camera model. Within this genericized framework, we implemented the Kannala-Brandt camera model, allowing fusion of fisheye depth maps.
- **A Hybrid Sparse-Dense Fisheye SLAM system**: Combining ORB-SLAM3 feature-based tracking with ElasticFusion dense mapping with fisheye cameras.
- **Kitti-360 Analysis**: We provide quantitative analysis of our system's depth estimation and local surface reconstruction accuracy, using the KITTI-360 benchmark dataset. Additionally, we provide qualitative results that demonstrate the overall performance of our proposed fisheye SLAM system.

An overview of our system's inputs and outputs can be seen in Figure 1 and Figure 2, respectively. In this work we emphasise local map accuracy over global consistency. Though they are not mutually exclusive, we argue that local surface reconstruction accuracy is, in many robotics applications, just as important as achieving global consistency, one of the main objectives of SLAM. Dense monocular mapping with a fisheye camera is a challenging problem, however, as our experimental results show locally accurate reconstructions are possible.

2. Dense Monocular Fisheye SLAM

The hybrid system of [5] consists of 6 main elements: (i) Neural network-based metric depth prediction; (ii) Sparse feature-based tracking with ORBSLAM-3 providing estimates of the initial camera pose; (iii) Refinement of the initial sparse pose estimate through direct whole image alignment against the dense map ; (iv) Loop closure detections from ORB-SLAM's pipeline to bootstrap a deformation graph which is applied to reflect the loop closure in the dense map; (v) Local, model-to-model loop closures in the dense map reactivate inactive surfels used for data-association and camera tracking; (vi) Fusion of the current live camera frame and depth prediction into the dense model. At the foundation of this SLAM system is the pinhole camera model limited to narrow FOV cameras. In this paper, we go beyond this initial system, developing a novel system that takes advantage of large FOV fisheye imagery. The architecture of the proposed system is summarised as follows:

1. PackNet [22] is trained to predict metrically-scaled depth maps for fisheye cameras. Metric scaling is achieved by training the network with a weak velocity supervision on the camera pose. At inference time the velocity is not required. The network is used to infer metric scale depth maps using only the current live RGB image as input.
2. ORB-SLAM3 is used to provide the camera pose in each frame where we use the live RGB for monocular camera tracking. Internally ORB-SLAM uses the Kannala-Brandt fisheye model to process fisheye images without undistorting them.
3. The ORB-SLAM3 estimated camera pose is used to fuse the current RGB and depth information into a dense surfel-based map, similar to the map representation of [23]. Our fusion operation incorporates the Kannala-Brandt fisheye camera model.
4. Global and local loop closure correction allow the system to correct for accumulated drift and reuse old parts of the map.

Our full pipeline is visualised in Figure 3

2.1. Kannala-Brandt Camera Model

We use the 8-parameter Kannala-Brandt camera similar to OpenCV's implementation and the implementation used in ORB-SLAM3. The first 4 parameters of the model are the focal lengths in the x and y direction f_x and f_y and the principal point (c_x, c_y). The final 4 parameters are the coefficients of a 5^{th} degree polynomial for correcting the point of projection as a function of angle of incidence $(\kappa_0, ..., \kappa_3)$. In particular, the projection function is given by,

$$p = \pi\left(P_w\right) = \begin{bmatrix} f_x \cdot r \cdot cos(\psi) + c_x \\ f_y \cdot r \cdot sin(\psi) + c_y \end{bmatrix} \tag{1}$$

where,

$$\begin{aligned} \theta &= atan2\left(\sqrt{X^2 + Y^2}, Z\right) \\ \psi &= atan2(Y, X) \\ r(\theta) &= \theta + \kappa_0\theta^3 + \kappa_1\theta^5 + \kappa_2\theta^7 + \kappa_3\theta^9 \end{aligned}$$

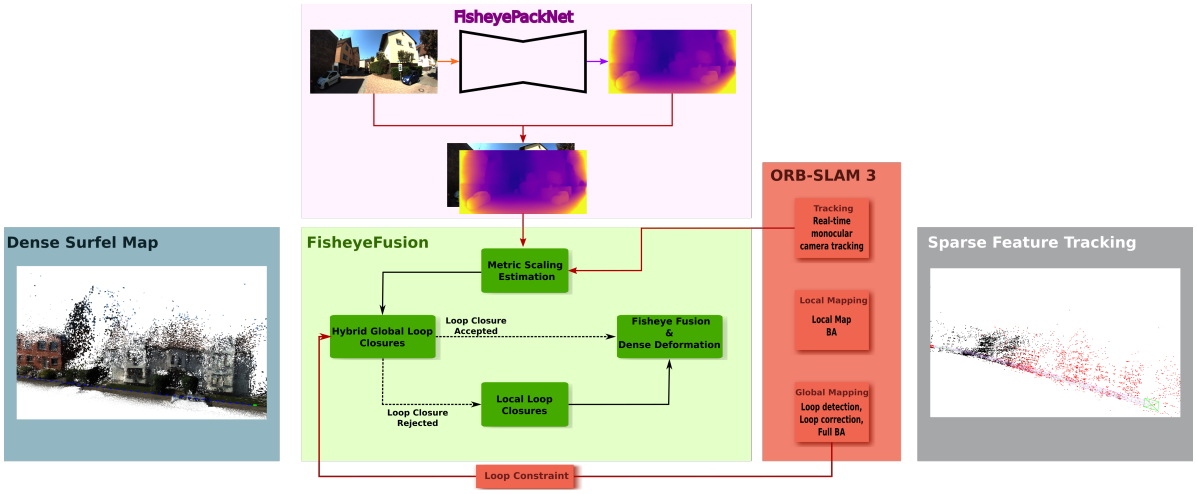


Figure 3. **Overview of system architecture.** PackNet is used to estimate a depth map for the current live camera image. ORB-SLAM then tracks the motion of the camera. ORB-SLAM continues to extract keyframes and pass them to its local and global mapping threads. The sparse feature map on the right is used internally by ORB-SLAM. Assuming no local or global loop closures have occurred in this timestep, the live RGB image and predicted depth are fused into the surfel model. If ORB-SLAM has detected a global loop closure, then a global deformation is attempted. Surface-surface constraints for optimization of the deformation graph are generated from the pre-corrected and corrected (post loop closure) pose of the camera P_{kf}^t and $\hat{P}_{kf}^t$ respectively. If deformation graph optimization fails or no global loop is detected by ORB-SLAM a local loop closure is attempted as per the original EF algorithm.

Here, $P_w = (X, Y, Z) \in \mathbb{R}^3$ denotes a $3D$ point in camera coordinates, with the corresponding image point given by $p \in \mathbb{R}^2 \subset \Omega$ where Ω is the image domain. To re-project an image point to world coordinates, the function π can be inverted. The inverse of the mapping from Cartesian coordinates to pixel coordinates (i.e the application of f_x, f_y, c_x, c_y in Equation (1) above) and from ray view direction to Cartesian coordinates ($(cos(\psi), sin(\psi))$ in Equation (1) above) is straightforward to compute. The radial distortion component can be inverted by solving for the roots of $r(\theta)$ using Newton's method. See [10] for more details.

2.2. Scale-Aware Fisheye Depth Estimation

For depth estimation, we propose a variant of PackNet [22]. The network is composed of two sub networks; a depth estimation network $\mathbf{F}_d : I \rightarrow \frac{1}{d(I)}$ (that is, the network outputs inverse depth) and pose estimation network $\mathbf{F}_P : (I_t, I_{\mathcal{N}}) \rightarrow P_{t \rightarrow n}$ that predicts the pose of each frame I_n in the frame of reference of I_t. We use a lightweight version of the depth network more suited to real-time operation. Both networks are trained simultaneously in a self-supervised manner by establishing a structure-from-motion loss function over a small temporal window around the current training frame. To do so, the pixels of each context frame $I_n \in \mathcal{N}(I_t)$ are warped into the target frame I_t, producing a set of warped frames $\hat{I}_n$. The basis of the self-supervised loss function is the photometric error induced between the target frame and each of the warped frames. We reformulate the multi-view warping function of PackNet to employ the Kannala-Brandt camera model, allowing the network to estimate dense depth maps for fisheye camera imagery. The function to warp an image point x_n in a context frame to its corresponding location x_t in the target frame is given by

$$x_t = \pi \left(\mathbf{P}_n^t \cdot \left[\frac{1}{d_n^{-1}} \cdot \pi^{-1} \left(x_n \right) \right] \right) \tag{2}$$

where, π is the Kannala-Brandt projection model from Section 2.1, $\mathbf{P}_n^t$ is the rigid body pose from context frame I_n to target frame I_t and d_n^{-1} is the inverse depth estimate for image point n. Note that for simplicity we have left out homogenisation and de-homogenisation operations. We take additional measures to enforce metric scale-aware depth estimates and to improve overall training and inference performance. Firstly, we apply weak velocity supervision loss on the pose network. Secondly, we perform sparse LiDAR supervision. For example, when using the KITTI-360 dataset, we render the LiDAR and SICK laser scanner point clouds into each fisheye frame using the ground-truth camera poses. This produces a set of sparse ground-truth depth maps which we use to supervise the training. The full loss function is

$$\mathcal{L} = \mathcal{L}_p + \lambda_1 \mathcal{L}_{reg} + \lambda_2 \mathcal{L}_{vel} + \lambda_3 \mathcal{L}_{depth} \tag{3}$$

where, $\mathcal{L}_p$, $\mathcal{L}_{reg}$ and $\mathcal{L}_{vel}$ are, respectively, the photometric loss, depth smoothness loss and velocity supervision loss as per the original PackNet paper [22]. $\mathcal{L}_{depth}$ is an L_1 loss over groundtruth and predicted depth estimates (see [24] for more details).

2.3. Hybrid Sparse-Dense Mapping With Kannala-Brandt Camera Model

To compute a dense model of the environment, we extend the hybrid sparse-dense SLAM system of [5]. ORB-SLAM3 is used for camera tracking and global loop closure detection [21]. To achieve dense mapping, [5] combines ORB-SLAM3 with ElasticFusion [23]. As in ElasticFusion, the dense map is represented as an unordered list of surfels $\mathcal{M}$ where each surfel consists of a position $p \in \mathbb{R}^3$, a normal $n \in \mathbb{R}^3$, a radius $r \in \mathbb{R}$, a confidence $c \in \mathbb{R}$ and a color. At each point in time, the map is divided into two non-overlapping subsets of surfels; an active map $\mathcal{M}_a$ representing surfels that have been fused with live data within the last δ_t iterations and; an inactive map $\mathcal{M}_i$ representing surfels that have not been fused with live camera data within the last δ_t iterations. For each live frame I_t our system runs a number of steps detailed below.

Depth estimation The system computes a depth map $D_t = F_d(I_t)$ using the depth estimation network described in Section 2.2.

Sparse camera tracking I_t is passed to ORBSLAM-3 which performs feature-based camera tracking as per the original ORB-SLAM algorithm [21]. Note that we run the full ORB-SLAM pipeline in the background including camera tracking, local mapping and loop closure detection and global optimisation. The output of this phase is the pose of the camera for the current timestep $\mathbf{P}_t \in SE(3)$.

Global loop closures We close global loops in a two-stage hybrid process similar to [5]. ORB-SLAM3 extracts novel and informative frames as keyframes, using a background thread to search for loop closures between keyframes. Once a loop closure candidate is found, it is geometrically verified and a constraint $\mathbb{T}_{K_i}^{K_j} \in SE(3)$ between the two triggering keyframes is calculated. ORB-SLAM corrects its sparse map with a pose-graph optimisation followed by a bundle adjustment. To correct the dense map, our system renders the dense model into two virtual frames located at the same poses as the two triggering keyframes. The pose of the newly added keyframe K_i is used to render the active portion of the model $\mathcal{M}_a$, while the pose of the matched keyframe K_j, transformed by the loop closing constraint $\mathbb{T}_{K_i}^{K_j}$, is used to render the inactive portion of the model $\mathcal{M}_i$. This ensures that it is the newer, active portion of the model that is brought into registration with the old part of the model during loop correction. From these two frames, a set of $3D \rightarrow 3D$ surface-surface correspondences are calculated. The surface correspondences are used to optimise a deformation graph which, when applied to the dense model, corrects its geometry to reflect the loop closure.

Local loop closures Assuming no global loop closure has occurred during the current iteration, the system looks for a local loop closure within the dense model. A local loop closure seeks to align the active and inactive portions of the model in view of the current camera pose. Local loop closures help keep the camera aligned with the dense map during locally loopy camera motion. To do so, we render two views of the scene, one each for the active and inactive map regions, using P_t. The loop closure constraint is calculated via a robust ICP non-linear optimisation, as per the original EF algorithm. The result is a 6-DOF rigid body transform T_a^i aligning the active frame with the inactive frame. Since local loops are purely model-to-model, instead of using π to project $\mathcal{M}_a$ and $\mathcal{M}_i$ into P_t, we render to virtual pinhole images. This allows us to take advantage of the EF's RGB-D alignment. To correct the dense geometry, T_a^i is used to compute a set of $3D \rightarrow 3D$ surface-surface point correspondences, in the same way as during a global loop closure. The loop is then corrected with a deformation graph in the same manner as a global loop closure.

Fusion Once the camera has been tracked and all loop closures have been identified, the current live camera frame I_t and estimated depth map D_t are fused into the active region of the map $\mathcal{M}_a$. To do so, data-associations between $\mathcal{M}_a$ and I_t are found by rendering a high resolution index map of $\mathcal{M}_a$ using $\mathbf{P}_t$ and π. Once the current frame has been associated, fusion proceeds in the same way as in [23].

2.4. Accounting for Scale

The overall scale of a scene is unobservable by a single monocular camera with no prior information. Therefore, a monocular mapping system can only estimate the motion of the camera and the geometry of the scene up to a scale factor. What's more, the scale of the reconstruction tends to drift over time. For long and loopy mapping sessions, scale consistency can be enforced by observing and correcting scale drift during loop closures [25]. In our system, before we can use $\mathbf{P}_t$ to fuse the current frame and depth map, we must recover the scale ratio $s \in \mathbb{R}$ between the sparse reconstruction and the dense reconstruction. We estimate the scale ratio between the metric dense depth estimate at time t D_t and the sparse ORB-SLAM map points $\mathcal{M}_{ORB}$ in view of the current pose $\mathbf{P}_t$. First, we render a sparse depth map D_{ORB} from $\mathcal{M}_{ORB}$ using $\mathbf{P}_t$. The scale difference at time step t is given by median ratio $s_t = \frac{med(D_t)}{med(D_{orb})}$. To ameliorate noise and to account for the fact that scale drifts over time, we keep a rolling buffer of scale estimates over the last n frames $\mathbf{S}$. The final scale ratio $s \in \mathbb{R}$ estimate is then taken as the average over $\{s_i \in \mathbf{S} | s_i \in [\mu_\mathbf{S} \pm 2\sigma_\mathbf{S}]\}$. Once the scale ratio s has been recovered we use it to scale the translational component of the current pose, lifting it into metric scale and allowing fusion of the current frame into the metric scale dense model.

Method	Abs_{rel}	Sq_{rel}	RMSE	RMSE_{log}
PackNet Fisheye	0.127	0.605	2.422	0.204

Table 1. **Evaluation of depth estimation on the proposed KITTI-360 test split.**

3. Experimental Results

To investigate the suitability of our proposed approach to dense monocular fisheye SLAM, we performed a qualitative and quantitative analysis on the KITTI-360 benchmark dataset [6]. KITTI-360 is a follow-on to the well known KITTI benchmark dataset [26]. The dataset consists of 9 public access sequences of a car driving through a suburban environment where the car is equipped with two lateral facing fisheye cameras pointing in opposite directions. The sequences also include 3D LiDAR and a 2D SICK laser scanner. Dataset annotations include groundtruth poses, per-frame LiDAR and SICK pointclouds, accumulated pointclouds and calibration parameters. Semantic labels, GPS/IMU data, and bounding boxes are also included, however we do not make use of these additional annotations in this work.

Note that KITTI-360 provides a different camera model for the fisheye cameras to the one used in our SLAM system. To utilise KITTI-360 with our system we calibrate the fisheye cameras using the chessboard detector of [27] and the fisheye calibration optimisation of OpenCV [28].

3.1. Depth Estimation Training Results

We trained Fisheye PackNet described in Section 2.2 on the fisheye images of the KITTI-360 dataset. Each of the 9 KITTI-360 sequences vary in length, ranging from $\sim 1k$ frames to $\sim 20k$ frames. We use both left and right images during training but treat them as separate samples. For training we end up with $\sim 45k$ training images, $\sim 45k$ testing images and $\sim 20k$ validation images. Sparse depth map training labels are produced in a pre-processing step by rendering the LiDAR point clouds and SICK point clouds into the ground truth camera frame. We train the network on a GPU server with 4 Nvidia RTX-3090 cards for 66 epochs. Each epoch takes approximately 1hrs40mins. In Table 1 we establish a first baseline of depth estimation results on KITTI-360 fisheye, reporting the accuracy of the trained depth estimation on the KITTI-360 test split with respect to the ground truth LiDAR scans.

3.2. Dense Fisheye Mapping Results

In this section, we present the results of experiments that investigate the effectiveness, at a local level, of our hybrid approach to dense monocular fisheye SLAM in estimating de-noised surface reconstructions. We also provide a number of qualitative examples of the system's overall performance, including side-by-side comparison of estimated and ground truth trajectories (e.g. Figures 5a and 6). All Mapping experiments are run on an i7 CPU using an NVIDIA 1080ti using the test splits outlined in Section 3.1. Within the testing split of each sequence, we use chunks of contiguous frames for benchmarking our system. In our experiments, we demonstrate that, while global consistency can be difficult to achieve, our system still consistently achieves accurate local mapping over windows up to $> 300m$. In Figure 5 we report surface reconstruction accuracy results. In particular, for a given test sequence, we retrieve nearby ground truth LiDAR scans to construct a local model. We then align this local model to the model estimated by our system. We take the mean distance between each point in our model and its corresponding closest point in the aligned local model as a measure of the local mapping accuracy of our system. Figure 5a shows how incrementally increasing the neighbourhood region from $10 - 300m$ within which LiDAR scans are gathered results in local mapping accuracy of $\sim 13cm - \sim 42cm$. In Figure 5b we show how estimating scale is challenging for our system given the use of monocular ORBSLAM and the scale ratio (see Section 2.4). Figure 5b shows the trajectory estimated by our system (lifted into metric scale during mapping) alongside the ground truth trajectory. We note that despite the exhibited scale drift, the overall morphology of the trajectory is captured in the output, and as shown in Figure 5a, the resulting map remains highly accurate over ranges of $150m$. In Figure 7 we report a breakdown of our system's runtime performance. The system runs at $\sim 5hz$. Note that as the number of surfels increases, the time taken to fuse new frames into the dense model also increases. Depth estimation is the main bottleneck in the system. Figure 6 qualitatively demonstrates loop closures in our system.

4. Conclusions

We presented a monocular fisheye dense mapping system that combines dense depth prediction with sparse feature tracking and dense surfel fusion techniques. Fisheye depth prediction is achieved by integrating the Kannala-Brandt camera model into PackNet. With a combination of weak velocity supervision and sparse LiDAR-based depth supervision, we trained PackNet to predict dense depth maps on the fisheye images of the KITTI-360 dataset. The proposed SLAM system permits live-dense reconstruction of outdoor scenes using a fisheye camera in automotive scenarios. Sparse tracking provides camera pose estimation capable of operating robustly at vehicle speeds and in outdoor environments with variable lighting. Global loop closures are identified by ORB-SLAM3's appearance-based place recognition module with the resultant constraints passed to the dense fusion algorithm where they are integrated with a

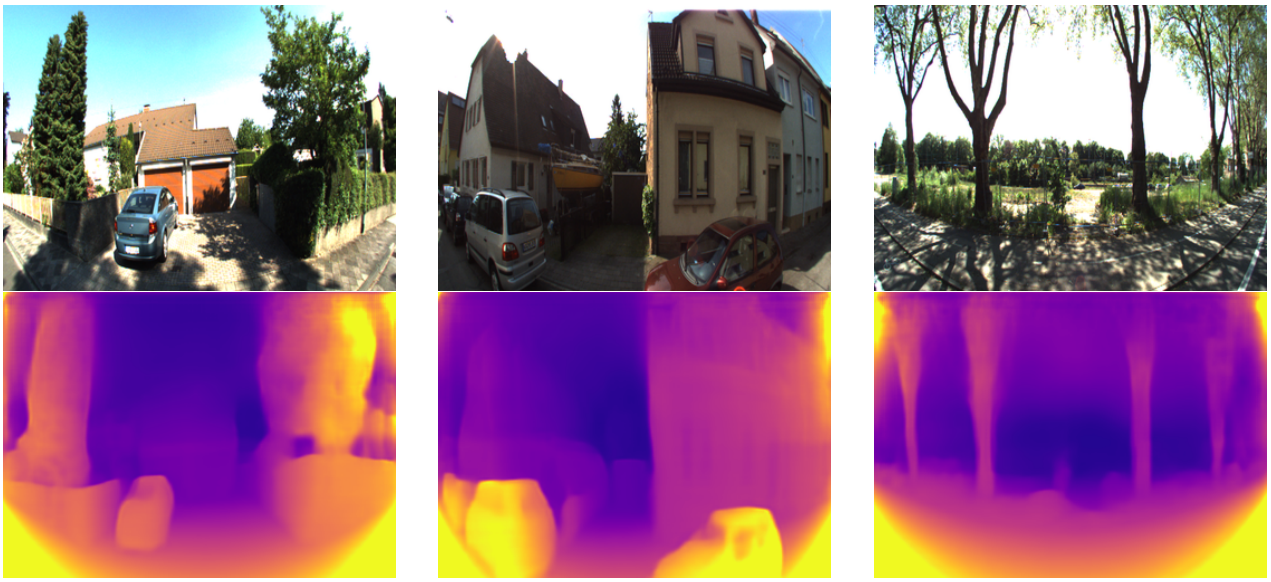
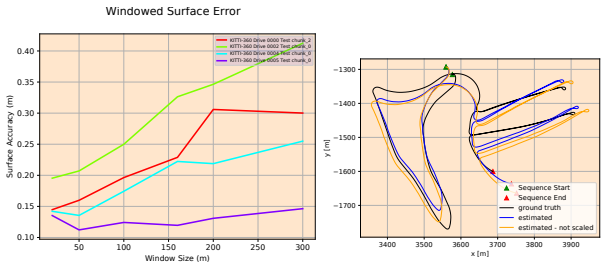


Figure 4. **Qualitative results of Fisheye PackNet on test split of KITTI-360.** The top row shows the input RGB fisheye image, the bottom row shows the estimated depth maps. Left column; note that while the gross geometry of the scene is well estimated, finer grained details, on the garage door and on the rear of the car, are smoothed out. Middle column; the prominent building corner highlights instabilities in depth estimation of sharp details in the scene. Right column; foliage represents a particularly challenging input for the network.



(a) (b)

Figure 5. **Windowed dense surface reconstruction accuracy from test splits from [6].** Figure 5a shows the mean surface-surface error as a function of local window size. For each of the sequences in Fig. 5a, we average the results across 5 uniformly selected locations from trajectory. Figure 5b shows our system's estimated trajectory alongside the ground truth trajectory for KITTI-360 sequence 4. The blue curve shows the estimated trajectory aligned and scaled to the ground truth. Note how, even though tracking exhibits significant scale drift, within relative large local areas ($\sim 300m$) mapping is accurate.

deformation graph-based map correction step.

Our results demonstrate accurate dense, metric visual mapping with a single fisheye camera over local windows of up to $300m$. The integration of loop closure constraints within a SLAM based approach permit leveraging of repeated passes of regions of the environment withing a globally consistent framework. Although our system exhibits accurate mapping over extended scales, globally accurate mapping remains a challenge. Within the context of KITTI-360, the lateral, sideways facing positioning of the fisheye cameras, combined with the fast motion of the vehicle, represents a difficult input for ORB-SLAM3. Furthermore, while we have incorporated measures to address scale ambiguity inherent in monocular SLAM, it remains a significant challenge over long sequences where the scale of the sparse map is bound to drift.

Supporting a wide range of camera models within a single mapping system is a challenging problem. Projection and re-projection functions can vary drastically between different camera models. Often the Jacobian of the camera model is required for integrating the model into non-linear camera tracking and mapping optimisations. Furthermore, each camera model requires calibration, which itself can involve a complicated procedure that may fail or produce a poor calibration. This limits a SLAM systems suitability as a research and consumer grade tool. In future work, we aim to investigate Neural Ray Surfaces [29] and their ability to recover a ray surface for a camera without known calibration parameters, as the basis for a unified approach to dense, CNN-based mapping.

References

[1] A. Kim and R. M. Eustice, "Real-time visual slam for autonomous underwater hull inspection using visual saliency," IEEE Transactions on Robotics, vol. 29, no. 3, pp. 719–733, 2013.

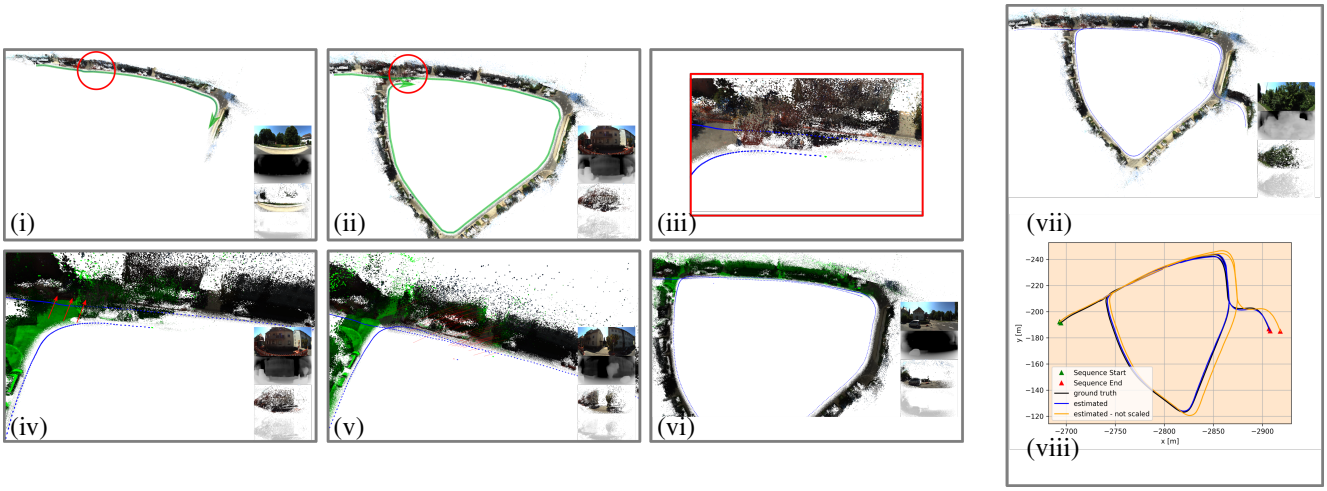


Figure 6. **A qualitative example of our system's performance**. (i) The system begins building a dense map as the vehicle explores (green arrow). (ii) The vehicle loops back on itself (red circle). (iii) Error accumulated during the loop results in drift. (iv) The active portion of the model (green) needs to be brought back into alignment with the old portion of the model (greyed out) from the vehicle's first pass. (v) A loop closure is detected and a deformation is applied to the dense surface, correcting for the effects of drift. Red lines denote surface-surface deformation constraints. (vi) Mapping continues; local loop closures reactivate the old part of the map from the first pass which can now be used for mapping. (vii) The final global model. $(viii)$ The estimated and ground truth trajectories. The sequence used in this example corresponds to a sub-sequence from KITTI-360 sequence 9 [6].

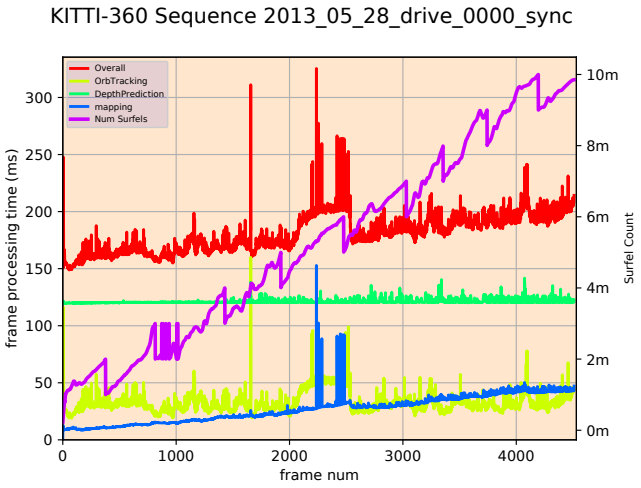


Figure 7. **A breakdown of our system's run-time performance.**

[2] L. Matthies, M. Maimone, A. Johnson, Y. Cheng, R. Willson, C. Villalpando, S. Goldberg, A. Huertas, A. Stein, and A. Angelova, "Computer vision on mars," International Journal of Computer Vision, vol. 75, no. 1, pp. 67 – 92, October 2007.

[3] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-LiDAR From Visual Depth Estimation: Bridging the Gap in 3D Object Detection for Autonomous Driving," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 8437–8445.

[4] Y. You, Y. Wang, W.-L. Chao, D. Garg, G. Pleiss, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving," in ICLR, 2020.

[5] L. Gallagher, V. Ravi Kumar, S. Yogamani, and J. B. McDonald, "A hybrid sparse-dense monocular slam system for autonomous driving," in European Conference on Mobile Robots, 2021.

[6] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," arXiv preprint arXiv:2109.13410, 2021.

[7] C. Hughes, M. Glavin, E. Jones, and P. Denny, "Wide-angle camera technology for automotive applications: a review," Iet Intelligent Transport Systems, vol. 3, pp. 19–31, 2009.

[8] Z. Zhang, H. Rebecq, C. Forster, and D. Scaramuzza, "Benefit of large field-of-view cameras for visual odometry," in 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 801–808.

[9] V. R. Kumar, C. Eising, C. Witt, and S. K. Yogamani, "Surround-view fisheye camera perception for automated driving: Overview, survey and challenges," CoRR, vol. abs/2205.13281, 2022.

[10] J. Kannala and S. Brandt, "A generic camera model and calibration method for conventional, wide-angle,

and fish-eye lenses," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006.

[11] D. Caruso, J. Engel, and D. Cremers, "Large-scale direct slam for omnidirectional cameras," in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 141–148.

[12] J. J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in ECCV, 2014.

[13] C. Geyer and K. Daniilidis, "A unifying theory for central panoramic systems and practical applications," in Proceedings of the 6th European Conference on Computer Vision-Part II, ser. ECCV '00. Berlin, Heidelberg: Springer-Verlag, 2000, p. 445–461.

[14] H. Matsuki, L. von Stumberg, V. Usenko, J. Stückler, and D. Cremers, "Omnidirectional dso: Direct sparse odometry with fisheye cameras," IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 3693–3700, 2018.

[15] 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.

[16] T. Schöps, T. Sattler, C. Häne, and M. Pollefeys, "Large-scale outdoor 3d reconstruction on a mobile device," Computer Vision and Image Understanding, 2017.

[17] Z. Cui, L. Heng, Y. C. Yeo, A. Geiger, M. Pollefeys, and T. Sattler, "Real-time dense mapping for self-driving vehicles using fisheye cameras," International Conference on Robotics and Automation, 2019.

[18] C. Won, H. Seok, Z. Cui, M. Pollefeys, and J. Lim, "Omnislam: Omnidirectional localization and dense mapping for wide-baseline multi-camera systems," IEEE International Conference on Robotics and Automation, 2020.

[19] C. Won, J. Ryu, and J. Lim, "Omnimvs: End-to-end learning for omnidirectional stereo matching," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 8986–8995.

[20] H. Seok and J. Lim, "Rovo: Robust omnidirectional visual odometry for wide-baseline wide-fov camera systems," 2019 International Conference on Robotics and Automation (ICRA), pp. 6344–6350, 2019.

[21] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," IEEE Transactions on Robotics, 2021.

[22] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon, "3d packing for self-supervised monocular depth estimation," in IEEE Conference on Computer Vision and Pattern Recognition, 2020.

[23] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison, "Elasticfusion: Dense slam without a pose graph," in Robotics: Science and Systems, 2015.

[24] V. C. Guizilini, J. Li, R. Ambrus, S. Pillai, and A. Gaidon, "Robust semi-supervised monocular depth estimation with reprojected distances," ArXiv, vol. abs/1910.01765, 2019.

[25] H. M. Strasdat, J. M. M. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular slam," in Robotics: Science and Systems, 2010.

[26] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," International Journal of Robotics Research (IJRR), 2013.

[27] A. Geiger, F. Moosmann, O. Car, and B. Schuster, "Automatic calibration of range and camera sensors using a single shot," in International Conference on Robotics and Automation (ICRA), 2012.

[28] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.

[29] I. Vasiljevic, V. Guizilini, R. Ambrus, S. Pillai, W. Burgard, G. Shakhnarovich, and A. Gaidon, "Neural ray surfaces for self-supervised learning of depth and ego-motion," in International Conference on 3D Vision, 2020.