GeoRefine: Self-Supervised Online Depth Refinement for Accurate Dense Mapping

Pan Ji*, Qingan Yan*, Yuxin Ma, and Yi Xu

OPPO US Research Center, InnoPeak Technology, Inc

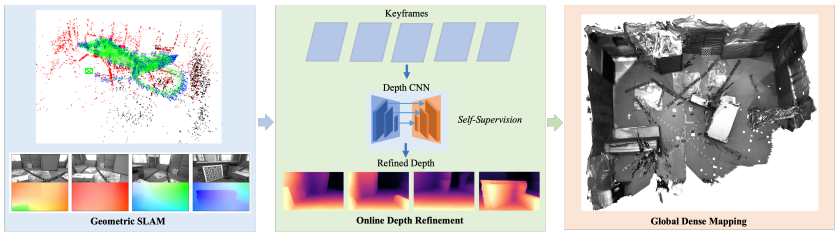

Fig. 1: We present an online depth refinement system for geometrically-consistent dense mapping from monocular data. Our system starts with geometric SLAM that is made robust by incorporating learning-based priors. Together with map points and camera poses from SLAM, a depth CNN is continuously updated using self-supervised losses. A globally consistent map is finally reconstructed from refined depth maps via an off-the-shell TSDF fusion method.

Abstract. We present a robust and accurate depth refinement system, named GeoRefine, for geometrically-consistent dense mapping from monocular sequences. GeoRefine consists of three modules: a hybrid SLAM module using learning-based priors, an online depth refinement module leveraging self-supervision, and a global mapping module via TSDF fusion. The proposed system is online by design and achieves great robustness and accuracy via: (i) a robustified hybrid SLAM that incorporates learning-based optical flow and/or depth; (ii) self-supervised losses that leverage SLAM outputs and enforce long-term geometric consistency; (iii) careful system design that avoids degenerate cases in online depth refinement. We extensively evaluate GeoRefine on multiple public datasets and reach as low as 5% absolute relative depth errors.

1 Introduction

Over the years, monocular geometric methods have been continuously improved and become very accurate in recovering 3D map points. Representative open-source systems along this line include COLMAP [43] – an offline SfM system, and ORB-SLAM [35,36,5] – an online SLAM system.

Recently, deep-learning-based methods [9,14,16] have achieved impressive results in predicting a *dense* depth map from a single image. Those models are

* Joint first authorship. P. Ji is the corresponding author (peterji530@gmail.com).

either trained in a supervised manner [9,40,39] using ground-truth depths, or through a self-supervised framework [14,16] leveraging photometric consistency between stereo and/or monocular image pairs. During inference, with the prior knowledge learned from data, the depth models can generate *dense* depth images even in textureless regions. However, the errors in the predicted depths are still relatively high.

A few methods [54,33] aim to get the best out of geometric systems and deep methods. Tiwari *et al.* [54] let monocular SLAM and learning-based depth prediction form a self-improving loop to improve the performance of each module. Luo *et al.* [33] adopt a test-time fine tuning strategy to enforce geometric consistency using outputs from COLMAP. Nonetheless, both methods pre-compute and store sparse map points and camera poses from SfM or SLAM in an offline manner, which is not applicable to many applications where data pre-processing is not possible. For example, after we deploy an agent to an environment, we want it to automatically improve its 3D perception capability as it moves around. In such a scenario, an online learning method is more desirable.

In this paper, we propose to combine geometric SLAM and a single-image depth model within an **online** learning scheme (see Fig. 1). The depth model can be any model that has been pretrained either with a self-supervised method [16] or a supervised one [40,39]. Our goal is then to incrementally refine this depth model on the test sequences in an online manner to achieve geometrically consistent depth predictions over the entire image sequence. Note that SLAM in itself is an online system that perfectly fits our online learning framework, but on the other hand, front-end tracking of SLAM often fails under challenging conditions (*e.g.*, with fast motion and large rotation). We propose to enhance the robustness of geometric SLAM with learning-based priors, *e.g.*, RAFT-flow [52], which has been shown to be both robust and accurate in a wide range of *unseen* scenes [25,64,53]. We then design a parallel depth refinement module that optimizes the neural weights of depth CNN with self-supervised losses. We perform a careful analysis of failure cases of self-supervised refinement and propose a simple yet effective keyframe mechanism to make sure that no refinement step worsens depth results. We further propose a novel occlusion-aware depth consistency loss to promote long-term consistency over temporally distant keyframes. We perform detailed ablation study to verify the effectiveness of each new component of our proposed *GeoRefine*, and conduct extensive experiments on several public datasets [3,49,7,15], demonstrating state-of-the-art performance in terms of dense mapping from monocular images.

2 Related Work

Geometric Visual SLAM. SLAM is an online geometric system that reconstructs a 3D map consisting of 3D points and simultaneously localizes camera poses w.r.t. the map [4]. According to the methods used in front-end tracking, SLAM systems can be roughly classified into two categories: (i) direct SLAM [11,10,44], which directly minimizes photometric error between adjacent

frames and optimizes the geometry using semi-dense measurements; (ii) feature-based (indirect) SLAM [47,35,36,5], which extracts and tracks a set of sparse feature points and then computes the geometry in the back-end using these sparse measurements. Geometric SLAM systems have become accurate and robust due to a number of techniques developed over the years, including robust motion estimation [12], keyframe mechanism [23], bundle adjustment [55], and pose-graph optimization [26]. In our work, we build our system upon one of the state-of-the-art feature-based systems, *i.e.*, ORB-SLAM [5]. We use ORB-SLAM because it is open-source, delivers accurate 3D reconstructions, and supports multiple sensor modes.

Learning-Based SLAM. CNN-SLAM [50] is a hybrid SLAM system that uses CNN depth to bootstrap back-end optimization of sparse geometric SLAM and helps recover metric scale 3D reconstruction. In contrast, DROID-SLAM [53] builds SLAM from scratch with a deep learning framework and achieves unprecedented accuracy in camera poses, but does not have the functionality of dense mapping. TANDEM [24] presents a monocular tracking and dense mapping framework that relies on photometric bundle adjustment and a supervised multi-view stereo CNN model. CodeSLAM [2] is a real-time learning-based SLAM system that optimizes a compact depth code over a conditional variational auto-encoder (VAE) and simultaneously performs dense mapping. DeepFactor [6] extends CodeSLAM by using fully-differentiable factor-graph optimization. CodeMapping [34] further improves over CodeSLAM via introducing a separate dense mapping thread to ORB-SLAM3 [5] and additionally conditioning VAE on sparse map points and reprojection errors. Our system bears the most similarity with CodeMapping in terms of overall functionalities, but is significantly different in system design and far more accurate in dense mapping.

Supervised depth estimation. Supervised depth estimation methods dominate the early trials [9,31,56,66,51,38,32] of learning-based depth estimation. Eigen *et al.* [9] propose the first deep learning based method to predict depth maps via a convolutional neural network and introduce a set of depth evaluation metrics that are still widely used today. Liu *et al.* [31] formulate depth estimation as a continuous conditional random field (CRF) learning problem. Fu *et al.* [13] leverage a deep ordinal regression loss to train the depth network. A few other methods combine depth estimation with additional tasks, *e.g.*, pose estimation [56,66,51] and surface normal regression [38].

Self-supervised depth estimation. Self-supervised depth estimation has recently become popular [14,67,16,20,27,45,60,42,28]. Garg *et al.* [14] are the first to apply the photometric loss between left-right stereo image pairs to train a monocular depth model in an unsupervised/self-supervised way. Zhou *et al.* [67] further introduce a pose network to facilitate using a photometric loss across neighboring temporal images. Later self-supervised methods are proposed to improve the photometric self-supervision. Some methods [63,69,41] leverage an extra flow network to enforce cross-task consistency, while a few others [17,57,1] employ new loss terms during training. A notable recent method Monodepth2 by Godard *et al.* [16] achieves great improvements via a few thoughtful designs, in-

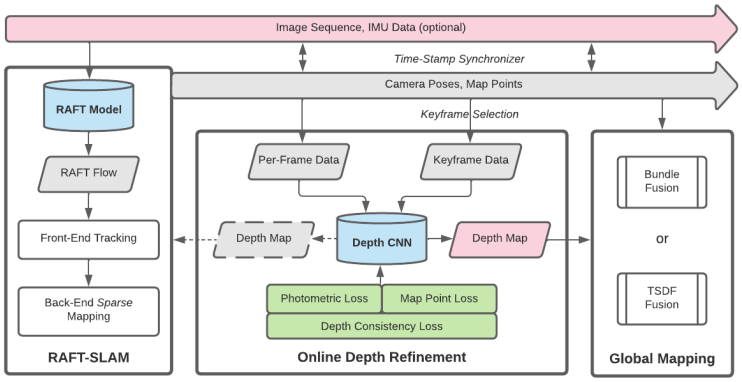

Fig. 2: The system workflow of our **GeoRefine**. Our system consists of three main modules, *i.e.*, a RAFT-SLAM module, an Online Depth Refinement module, and a Global Mapping module. Note that keyframe selection in *Online Dense Refinement* uses a **different** strategy than in SLAM.

cluding a per-pixel minimum photometric loss, an auto-masking strategy, and a multi-scale framework. New network architectures are also introduced to boost the performance. Along this line, Wang *et al.* [58] and Zou *et al.* [68] exploit recurrent networks for the pose and/or depth networks. Ji *et al.* [21] propose a depth factorization module and an iterative residual pose module to improve depth prediction for indoor environments. Our system is theoretically compatible with all those methods in the pretraining stage.

Instead of ground-truth depths, some methods [30,29,40,65,54,33] obtain the training depth data from the off-the-shell SfM or SLAM. Li and Snavely [30] perform 3D reconstruction of Internet photos via geometric SfM [43] and then use the reconstructed depths to train a depth network. Li *et al.* [29] learn the depths of moving people by watching and reconstructing static people. Ranftl *et al.* [40,39] improve generalization performance of the depth model by training the depth network with various sources, including ground-truth depths and geometrically reconstructed ones. Zhang *et al.* [64] extend the work of [33] to handling moving objects by unrolling scene flow prediction. Kopf *et al.* [25] further bypass the need of running COLMAP via the use of deformation splines to estimate camera poses. Most of those methods require a pre-processing step to compute and store 3D reconstructions. In contrast, our system runs in an online manner without the need of performing offline 3D reconstruction.

3 Method – GeoRefine

In this section, we present *GeoRefine*, a self-supervised depth refinement system for geometrically consistent dense mapping from monocular sequences. As shown in Fig. 2, our system consists of three parallel modules, *i.e.*, a RAFT-SLAM

module, an Online Depth Refinement module, and a Global Mapping module. We detail the first two modules in the following sub-sections.

3.1 RAFT-SLAM

It is well-known that monocular visual SLAM has several drawbacks: (i) its front-end often fails to track features under adverse environments, *e.g.*, with low-texture, fast motion, and large rotation; (ii) it can only reconstruct the scene up to an *unknown* global scale. To improve the performance of SLAM, a few methods [50,62,61] have been proposed to improve back-end optimization of *direct* LSD-SLAM [11]. In this work, we instead seek to improve the front-end of *feature-based* SLAM based on the observation that front-end tracking lose is one of the most common causes for failures and accuracy decrease. We thus present RAFT-SLAM, a hybrid SLAM system that runs a learning-based flow front-end and a traditional back-end optimizer.

3.1.1 RAFT-Flow Tracking RAFT [52] is one of the state-of-the-art optical flow methods that has shown strong cross-dataset generalization performance. It constructs a correlation volume for all pairs of pixels and uses a gated recurrent unit (GRU) to iteratively update the flow. In our system, we replace the front-end feature matching in ORB-SLAM [5] with RAFT-flow, but still sample sparse points for robust estimation in the back-end. This simple strategy allows us to have the advantages of both learning-based flow and traditional robust estimator in one system.

More specifically, for each feature from last frame $\mathbf{I}_{i-1}$, once it is associated with a map point, we locate its correspondence in incoming frame $\mathbf{I}_i$ by adding the flow $\mathbf{F}_{(i-1)\to i}$. If there are multiple candidates within a predefined radius around a target pixel in $\mathbf{I}_i$, we choose the one with the smallest descriptor residual; or if there is none, we create a new feature instead, with the descriptor being copied from $\mathbf{I}_{i-1}$. In all our experiments, we set the radius to 1 pixel. For the sake of robustness, we only keep $N_f = 0.1 \cdot N_t$ matched correspondences for initial pose calculation, where N_t is the total ORB features within the current frame. We note that, compared to leveraging the entire flow, sampling a subset of pixels is more beneficial to the accuracy. We do a forward-backward consistency check on predicted flows to obtain a valid flow mask by using a stringent threshold of 1 pixel. Similar to [5], we then perform a local map point tracking step to densify potential associations from other views and further optimize the camera pose. The reason for combining ORB features with flow is that traditional features can help us keep the structural information, mitigating drifting caused by flow mapping in long sequential tracking.

3.1.2 Multiple Sensor Modes Our RAFT-SLAM inherits the good properties of ORB-SLAM3 [5] in supporting multiple sensor modes. In our system, we consider a minimum sensor setup, *i.e.*, using a monocular camera with (or

without) IMU sensor. Thus, two SLAM modes are supported, *i.e.*, the monocular and Visual-Inertial (VI) modes. As we have a CNN depth model to infer the depth map for every image, we additionally form a pseudo-RGBD (pRGBD) mode as in [54].

Monocular Mode. Under the monocular mode, RAFT-SLAM reconstructs camera poses and 3D map points in an arbitrary scale. Since we have a pretrained depth model available in our system, we then leverage the CNN predicted depth maps to adapt the scale of map points and camera poses for SLAM. This scale alignment step is necessary in our system because SLAM outputs will be used in the downstream task of refining the depth model. If the scales between these two modules differ too much, depth refinement will be sub-optimal or even totally fail. After initial map points are constructed in our system, we continuously align the scale for a few steps by solving the following least-squares problem:

$$\min_{s} \sum_{\mathbf{x}} \left(d(\mathbf{x}) - s \cdot \hat{d}(\mathbf{x}) \right)^2 , \tag{1}$$

where s is the scale alignment factor to be estimated, and $d(\mathbf{x}), \hat{d}(\mathbf{x})$ are the depth values from a pretrained depth model and SLAM map points respectively. However, if the scales of two modules are already in the same order, *e.g.*, when SLAM runs in the VI or pRGBD mode, such an alignment step is not necessary.

VI Mode. VI SLAM is usually more robust than monocular SLAM under challenging environments with low-texture, motion blur, and occlusions [5]. Since the inertial sensors provide scale information, camera poses and 3D map points from VI RAFT-SLAM are recovered in metric scale. In this mode, given a scale-aware depth model (*i.e.*, a model that predicts depth in metric scale), we can run the online depth refinement module without taking special care of the scale discrepancy between the two modules.

pRGBD Mode. The pRGBD mode provides a convenient way to incorporate deep depth priors into geometric SLAM. However, we observe that it results in sub-optimal SLAM performance if we naïvely treat depth predictions as the groundtruth to run the RGBD mode (as done in [54]) due to noisy predictions. In the RGBD mode of ORB-SLAM3 [5], the depth is mainly used in two stages, *i.e.*, system initialization and bundle adjustment. By using the input depth, the system can initialize instantly from the first frame, without the need of waiting for enough temporal baselines. For each detected feature point, employing the depth and camera parameters, the system creates a *virtual right correspondence*, which leads to an extra reprojection error term in bundle adjustment [5]. To mitigate the negative impact of the noise in depth predictions, we make two simple yet effective changes in the pRGBD mode as compared to the original RGBD mode: i) we take as input the refined depth maps from the online refinement module (as described in the next subsection) to ensure that the input depth maps are more accurate and temporally consistent; ii) we remove the reprojection error term for the virtual right points in bundle adjustment. Note that the input CNN depth is still used in the map point initialization and new keypoint insertion, benefiting the robustness of the SLAM system.

3.2 Online Depth Refinement

The depth refinement module receives map points and camera poses from RAFT-SLAM. The depth model is then incrementally refined with self-supervised losses, including a photometric loss, an edge-aware depth smoothness loss, a map-point loss, and a depth consistency loss.

Similar to [67], the photometric loss is defined as the difference between a target frame $\mathbf{I}_i$ and a synthesized frame $\mathbf{I}_{j\to i}$ warped from a source frame $\mathbf{I}_j$ using the depth image $\mathbf{D}_i$ and the relative pose $\mathbf{T}_{j\to i}$, *i.e.*,

$$L_p = \sum_j pe(\mathbf{I}_i, \mathbf{I}_{j\to i}) \ , \tag{2}$$

where $pe()$ is the photometric loss function computed with the ℓ_1 norm and the SSIM [59]. Instead of only using 3 neighboring frames to construct the photo-consistency as in [67,16], we employ a wider baseline photometric loss, *e.g.*, by using a 5-keyframe snippet with $j \in \mathcal{A}_i = \{i-9, i-6, i-3, i+1\}$. Another important difference is that the relative pose $\mathbf{T}_{j\to i}$ comes from our RAFT-SLAM, which is more accurate than the one predicted by a pose network.

Following [16], we use an edge-aware normalized smoothness loss, *i.e.*,

$$L_s = |\partial_x d_i^*| e^{-|\partial_x I_i|} + |\partial_y d_i^*| e^{-|\partial_y I_i|} \ , \tag{3}$$

where $d_i^* = d_i / \bar{d}_i$ is the mean-normalized inverse depth to prevent depth scale diminishing [57].

The map points from RAFT-SLAM have undergone extensive optimization through bundle adjustment [55], so the depths of these map points are usually more accurate than the pretrained CNN depths. As in [62,54], we also leverage the map-point depths to build a map-point loss as a supervision signal to the depth model. The map-point loss is simply the difference between SLAM map points and the corresponding CNN depths as follows,

$$L_m = \frac{1}{N_i} \sum_{n=1}^{N_i} \left| \mathbf{D}_{i,n} - D_{i,n}^{slam} \right| \ , \tag{4}$$

where we have N_i 3D map points from RAFT-SLAM after filtering with a stringent criterion (see Sec. 4.1) to ensure that only accurate map points are used as supervision. In addition to the above loss terms, we propose an occlusion-aware depth consistency loss and a keyframe strategy to build our online depth refinement pipeline.

3.2.1 Occlusion-Aware Depth Consistency Given the depth images of two adjacent images, *i.e.*, $\mathbf{D}_i$ and $\mathbf{D}_j$, and their relative pose $\mathbf{T} = [\mathbf{R}|\mathbf{t}]$, we aim to build a robust consistency loss between $\mathbf{D}_i$ and $\mathbf{D}_j$ to make the depth predictions consistent with each other. Note that the depth values are not necessarily equal at corresponding positions of frame i and j as the camera can move over

time. With camera pose $\mathbf{T}$, the depth map $\mathbf{D}_j$ can be warped and then transformed to a depth map $\tilde{\mathbf{D}}_i$ of frame i, via image warping and coordinate system transformation [1,21]. We then define our initial depth consistency loss as,

$$L_c(\mathbf{D}_i, \mathbf{D}_j) = \left| 1 - \tilde{\mathbf{D}}_i / \mathbf{D}_i \right| . \tag{5}$$

However, the loss in Eq. (5) will inevitably include pixels in occluded regions, which hamper model refinement. To effectively handle occlusions, following the per-pixel photometric loss in [16], we devise a per-pixel depth consistency loss by taking the minimum instead of the average over a set of neighboring frames:

$$L_c = \min_{j \in \mathcal{A}_i} L_c(\mathbf{D}_i, \mathbf{D}_j) . \tag{6}$$

3.2.2 Degenerate Cases and Keyframe Selection Self-supervised photometric losses are not without degenerate cases. If they are not carefully considered, self-supervised training or finetuning will deteriorate, leading to worse depth predictions. A first degenerate case happens when the camera stays *static*. This degeneracy has been well considered in the literature. For example, Zhou *et al.* [67] remove static frames in an image sequence by computing and thresholding the average optical flow of consecutive frames. Godard *et al.* [16] propose an auto-masking strategy to automatically mask out static pixels when calculating the photometric loss.

A second degenerate case is when the camera undergoes *purely rotational* motion. This degeneracy is well-known in traditional computer vision geometry [19], but has not been considered in self-supervised depth estimation. Under pure rotation, motion recovery using the fundamental matrix suffers from ambiguity, so homography-based methods are preferred [19]. In the context of the photometric loss, if the camera motion is pure rotation, *i.e.*, the translation $\mathbf{t} = 0$, the view synthesis (or reprojection) step does not depend on depth anymore (*i.e.*, depth cancels out after applying the projection function). This is no surprise as their 2D correspondences are directly related by a homography matrix. So in this case, as long as the camera motion is accurately given, any arbitrary depth can minimize the photometric loss, which is undesirable when we train or finetune the depth network (as depth will be arbitrarily wrong).

To circumvent the degenerate cases described above, we propose a simple yet effective keyframe mechanism to facilitate online depth refinement without deterioration. After we receive camera poses from RAFT-SLAM, we can simply select keyframes for depth refinement according to the magnitude of camera *translations*. Only if the norm of the camera translation is over a certain threshold (see Sec. 4.1), we set its corresponding frame as a keyframe, *i.e.*, the candidate for applying self-supervised losses. This ensures that we have enough baselines for the photometric loss to be effective.

3.2.3 Overall Refinement Strategy

Algorithm 1 *GeoRefine*: self-supervised online depth refinement for geometrically consistent dense mapping.

Pretrain the depth model. ▷ supervised or self-supervised
Run RAFT-SLAM. ▷ on separate threads
Data preparation: buffer time-synchronized keyframe data into a fixed-sized queue $\mathcal{Q}^*$; (optionally) form another data queue $\mathcal{Q}$ for per-frame data.
while True **do**
  Check stop condition. ▷ stop-signal from SLAM
  Check SLAM failure signal. ▷ clear data queue if received
  for $k \leftarrow 1$ to K^* **do** ▷ *keyframe refinement*
    Load data in $\mathcal{Q}^*$ to GPU, ▷ batch size as 1
    Compute losses as in Eq. (7),
    Update depth model via one gradient descent step. ▷ ADAM optimizer
  end for
  Run inference and save refined depth for current keyframe.
  for $k \leftarrow 1$ to K **do** ▷ *Per-frame refinement*
    Check camera translation from last frame, ▷ skip if too small
    Load data in $\mathcal{Q}^*$ and $\mathcal{Q}$ to GPU, ▷ batch size as 1
    Compute losses as in Eq. (7),
    Update depth model via one gradient descent step. ▷ ADAM optimizer
  end for
  Run inference and save refined depth for current frame.
end while
Run global mapping. ▷ TSDF or bundle fusion
Output: refined depth maps and global TSDF meshes.

Our overall refinement loss writes as

$$L = L_p + \lambda_s L_s + \lambda_m L_m + \lambda_c L_c \ , \tag{7}$$

where $\lambda_s, \lambda_m, \lambda_c$ are the weights balancing the contribution of each loss term.

GeoRefine aims to refine any pretrained depth models to achieve geometrically-consistent depth prediction for each frame of an image sequence. As RAFT-SLAM runs on separate threads, we buffer the keyframe data, including images, map points, and camera poses, into a time-synchronized data queue of a fixed size. If depth refinement is demanded for every frame, we additionally maintain a small data queue for per-frame data and construct the 5-frame snippet by taking 3 recent keyframes and 2 current consecutive frames. We conduct online refinement for the current keyframe (or frame) by minimizing the loss term in Eq. (7) and performing gradient descent for K^* (or K) steps. After depth refinement steps, we run depth inference using the refined depth model and save the depth map for the current keyframe (or frame). Global maps can be finally reconstructed by performing TSDF or bundle fusion [37,8]. The whole *GeoRefine* algorithm is summarized in Alg. 1.

4 Experiments

We mainly conduct experiments on three public datasets: EuRoC [3], TUM-RGBD [49], and ScanNet [7]. For quantitative depth evaluation, we employ the standard error and accuracy metrics, including the Mean Absolute Error (MAE), Absolute Relative (*Abs Rel*) error, *RMSE*, $\delta < 1.25$ (namely δ_1), $\delta < 1.25^2$ (namely δ_2), and $\delta < 1.25^3$ (namely δ_3) as defined in [9]. All of our experiments are conducted on an Intel i7 CPU machine with 16-GB memory and one 11-GB NVIDIA GTX 1080.

4.1 Implementation Details

Our GeoRefine includes a RAFT-SLAM module and an online depth refinement module. RAFT-SLAM is implemented based on ORB-SLAM3 [5] (other SLAM systems are also applicable) which support monocular, visual-inertial, and RGBD modes. In our experiments, we test the three modes and show that GeoRefine achieves consistent improvements over pretrained models. The pose data queue is maintained and updated in the SLAM side, where a frame pose is stored relative to its reference keyframe which is continuously optimized by BA and pose graph. The online learning module refines a pretrained depth model with customized data loader and training losses. In our experiments, we choose a supervised model, *i.e.*, DPT [39], to showcase the effectiveness of our system. The initial DPT model is trained on a variety of public datasets and then finetuned on NYUv2 [46]. We utilize Robot Operating System (ROS) [48] to exchange data between modules for cross-language compatibility. We use ADAM [22] as the optimizer and set the learning rate to $1.0e^{-5}$. The weighting parameters λ_s, λ_m, and λ_c are set to $1.0e^{-4}$, $5.0e^{-2}$, and $1.0e^{-1}$ respectively. We freeze its decoder layers of DPT for the sake of speed and stability. We filter map points with stringent criterion to ensure good supervision signal for online depth refinement. To this end, we discard map points observed in fewer than 5 keyframes or with reprojection errors greater than 1 pixel. We maintain a keyframe data queue of length 11 and a per-frame data queue of length 2. The translation threshold for keyframe (or per-frame) refinement is set to 0.05 m (or 0.01 m). The number of refinement steps for keyframes (or per-frame) is set to 3 (or 1). All system hyper-parameters are tuned on a validation sequence (EuRoC V2_01).

4.2 EuRoC Indoor MAV Dataset

The EuRoC MAV dataset [3] is an indoor dataset which provides stereo image sequences, IMU data, and camera parameters. An MAV mounted with global shutter stereo cameras is used to capture the data in a large machine hall and a VICON room. Five sequences are recorded in the machine hall and six are in the VICON room. The ground-truth camera poses and depths are obtained with a VICON device and a Leica MS50 laser scanner, so we use all Vicon sequences as the test set. We rectify the images with the provided intrinsics to remove image distortion. To generate ground-truth depths, we project the laser point cloud

Table 1: Quantitative depth evaluation on EuRoC under different SLAM modes.

Method	Monocular				Visual-Inertial				pRGBD			
	MAE ↓	AbsRel ↓	RMSE ↓	δ_1 ↑	MAE ↓	AbsRel ↓	RMSE ↓	δ_1 ↑	MAE ↓	AbsRel ↓	RMSE ↓	δ_1 ↑
V1_01												
DPT [39]	0.387	0.140	0.484	0.832	0.501	0.174	0.598	0.709	0.387	0.140	0.484	0.832
CodeMapping [34]	-	-	-	-	0.192	-	0.381	-	-	-	-	-
Ours-DPT	**0.153**	**0.050**	**0.241**	**0.980**	**0.147**	**0.048**	**0.241**	**0.980**	**0.151**	**0.049**	**0.239**	**0.982**
V1_02												
DPT [39]	0.320	0.119	0.412	0.882	0.496	0.182	0.586	0.712	0.320	0.119	0.412	0.882
CodeMapping [34]	-	-	-	-	0.259	-	0.369	-	-	-	-	-
Ours-DPT	**0.171**	**0.058**	**0.255**	**0.967**	**0.166**	**0.058**	**0.251**	**0.972**	**0.160**	**0.056**	**0.240**	**0.973**
V1_03												
Monodepth2 [16]	0.305	0.111	0.413	0.886	0.360	0.132	0.464	0.815	0.305	0.111	0.413	0.886
DPT [39]	0.305	0.112	0.396	0.890	0.499	0.185	0.581	0.700	0.305	0.112	0.396	0.890
CodeMapping [34]	-	-	-	-	0.283	-	0.407	-	-	-	-	-
Ours-DPT	**0.202**	**0.074**	**0.297**	**0.949**	**0.188**	**0.067**	**0.278**	**0.956**	**0.190**	**0.068**	**0.286**	**0.949**
V2_01												
Monodepth2 [16]	0.423	0.153	0.581	0.800	0.490	0.181	0.648	0.730	0.423	0.153	0.581	0.800
DPT [39]	0.325	0.128	0.436	0.854	0.482	0.205	0.571	0.703	0.325	0.128	0.436	0.854
CodeMapping [34]	-	-	-	-	0.290	-	0.428	-	-	-	-	-
MonoIndoor [21]	-	0.125	0.466	0.840	-	-	-	-	-	-	-	-
Ours-DPT	**0.170**	**0.054**	**0.258**	**0.973**	**0.162**	**0.052**	**0.258**	**0.970**	**0.181**	**0.057**	**0.0269**	**0.970**
V2_02												
Monodepth2 [16]	0.597	0.191	0.803	0.723	0.769	0.233	0.963	0.562	0.597	0.191	0.803	0.723
DPT [39]	0.404	0.134	0.540	0.838	0.601	0.191	0.727	0.699	0.404	0.134	0.540	0.838
CodeMapping [34]	-	-	-	-	0.415	-	0.655	-	-	-	-	-
Ours-DPT	**0.177**	**0.053**	**0.208**	**0.976**	**0.193**	**0.063**	**0.312**	**0.966**	**0.167**	**0.053**	**0.267**	**0.976**
V2_03												
Monodepth2 [16]	0.601	0.211	0.784	0.673	0.764	0.258	0.912	0.498	0.601	0.211	0.784	0.673
DPT [39]	0.283	0.099	0.366	0.905	0.480	0.154	0.564	0.746	0.283	0.099	0.366	0.905
CodeMapping [34]	-	-	-	-	0.686	-	0.952	-	-	-	-	-
Ours-DPT	**0.163**	**0.053**	**0.231**	**0.970**	**0.159**	**0.055**	**0.220**	**0.973**	**0.152**	**0.051**	**0.214**	**0.975**

onto the image plane of the left camera using the code by [18]. The original images have a size of 480×754 and are resized to 384×384 for DPT.

Quantitative Depth Results in the Monocular Mode. We conduct quantitative evaluation by running GeoRefine under *monocular* RAFT-SLAM on the EuRoC VICON sequences, and present the depth evaluation results in the left columns of Tab. 1. Following [16], we perform per-frame scale alignment between the depth prediction and the groundtruth. From Tab. 1, we can observe consistent and significant improvements by our method over the baseline model on all test sequences. In particular, on V1_01, "Ours-DPT" reduces *Abs Rel* from 14.0% (by DPT) to 5.0%, achieving over two-times reduction in depth errors.

Quantitative Depth Results in the Visual-Inertial Mode. When IMU data are available, we can also run GeoRefine under *visual-inertial* (VI) RAFT-SLAM to get camera poses and map points directly in metric scale. Note that, in the visual-inertial mode, no scale alignment is needed. We present the quantitative depth results in Tab. 1, from which we can see that our system under the VI mode performs on par with the monocular mode even without scale alignment. Compared to a similar dense mapping, *i.e.*, CodeMapping [34], our GeoRefine is significantly more accurate with similar runtime (*i.e.*, around 1 sec. per keyframe; see the supplementary), demonstrating the superiority of our system design.

Quantitative Depth Results in the pRGBD Mode. We present the quantitative depth evaluation under the *pRGBD* mode in the right columns of Tab. 1. We can see that the pRGBD mode performs slightly better than the other two modes in terms of depth results. This may be attributed to the fact that under

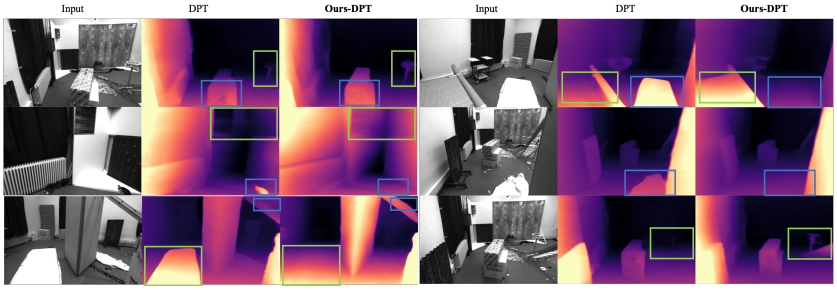

Fig. 3: Visual comparison of depth maps by the pretrained DPT and our system. Regions with salient improvements are highlighted with green/blue boxes.

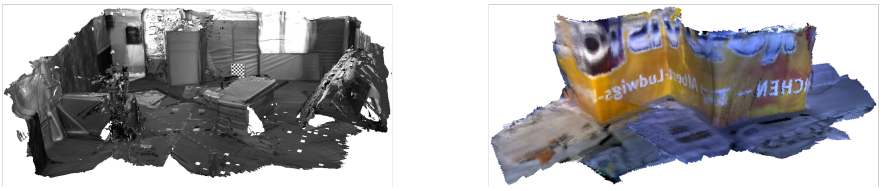

Fig. 4: Global reconstruction on EuRoC (left) and TUM-RGBD (right) using the refined depth maps by GeoRefine.

Table 2: Monocular SLAM results on EuRoC (RMSE ATE in meters).

Method	MH_01	MH_02	MH_03	MH_04	MH_05	V1_01	V1_02	V1_03	V2_01	V2_02	V2_03	Mean
DeepFactor [6]	1.587	1.479	3.139	5.331	4.002	1.520	0.679	0.900	0.876	1.905	1.021	2.040
DeepV2D [51]	0.739	1.144	0.752	1.492	1.567	0.981	0.801	1.570	0.290	2.202	2.743	1.298
D3VO [61]	-	-	0.080	-	0.090	-	-	0.110	-	0.050	0.019	-
DROID-SLAM [53]	0.013	**0.014**	**0.022**	**0.043**	0.043	0.037	0.012	**0.020**	**0.017**	0.013	**0.014**	**0.022**
ORB-SLAM [35]	0.071	0.067	0.071	0.082	0.060	**0.015**	0.020	x	0.021	0.018	x	-
DSO [10]	0.046	0.046	0.172	3.810	0.110	0.089	0.107	0.903	0.044	0.132	1.152	0.601
ORB-SLAM3 [5]	0.016	0.027	0.028	0.138	0.072	0.033	0.015	0.033	0.023	0.029	x	-
RAFT-SLAM (Ours)	**0.012**	0.018	0.023	0.045	**0.041**	0.032	**0.010**	0.022	0.019	**0.011**	0.025	0.023

this mode, the SLAM and depth refinement modules form a loosely-coupled loop so that each module benefits from the other.

Qualitative Depth Results. We show some visual comparisons in Fig. 3, from which we can clearly observe the qualitative improvements brought by our online depth refinement method. In particular, our system can correct the inaccurate geometry that is commonly present in the pretrained model. For example, in the first row of Fig. 3, a piece of thin paper lying on the floor is predicted to have much higher depth values than its neighboring floor pixels by the pretrained models (DPT); our GeoRefine is able to rectify its depth to be

Table 3: Ablation study on EuRoC Sequence V2_03. Each component in our method improves the depth results.

Method	Monocular					
	MAE ↓	Abs Rel ↓	RMSE ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑
DPT [39]	0.283	0.099	0.366	0.905	0.979	0.994
Our BaseSystem	0.269	0.090	0.347	0.905	0.983	0.997
+ RAFT-flow	0.248	0.083	0.331	0.915	0.985	0.997
+ Scale Alignment	0.199	0.064	0.274	0.952	0.991	0.998
+ Depth Consistency	**0.163**	**0.053**	**0.231**	**0.970**	**0.995**	**0.999**

Method	pRGBD					
	MAE ↓	Abs Rel ↓	RMSE ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑
DPT [39]	0.283	0.099	0.366	0.905	0.979	0.994
Our BaseSystem	0.216	0.076	0.288	0.933	0.989	0.998
+ Refined Depth	0.199	0.065	0.268	0.958	0.995	**0.999**
+ RAFT-flow	0.171	0.056	0.237	0.972	0.995	0.998
+ Remove BA Term	**0.152**	**0.051**	**0.214**	**0.975**	**0.997**	**0.999**

consistent with the floor. A global map of the EuRoC VICON room is shown in Fig. 1 and 4, where we can reach geometrically consistent reconstruction.

Odometry Results. Tab. 2 shows the odometry comparisons of our proposed RAFT-SLAM with current state-of-the-art methods on the EuRoC dataset in the monocular mode. For fairness, we adopt the same parameter settings with ORB-SLAM3 [5] in all our experiments. Note that, although our system is not elaborately designed for SLAM, it significantly outperforms other monocular baselines both in terms of accuracy and robustness, and achieves comparable results against DROID-SLAM [53] (with 19 steps of global bundle adjustment).

Ablation Study. Without loss of generality, we perform an ablation study on Seq. V2_03 to gauge the contribution of each component to our method under both monocular and pRGBD modes. Specifically, we first construct a base system by running a vanilla online refinement algorithm with the photometric loss as in Eq. (2), the depth smoothness loss as in Eq. (3), and the map-point loss as in Eq. (4). Note that the photometric loss uses camera poses from RAFT-SLAM instead of a pose network. Under the monocular mode, we denote this base model as "Our BaseSystem". We then gradually add new components to this base model, including the RAFT-flow in SLAM front-end ("+RAFT-flow"), the scale alignment strategy in RAFT-SLAM ("+Scale Alignment"), and the occlusion-aware depth consistency loss ("+Depth Consistency"). Under the pRGBD mode, "Our BaseSystem" takes the pretrained depth as input without using our proposed changes, and this base system uses the depth consistency loss. We then gradually add new components to the base system, *i.e.*, using refined depth from the online depth refinement module ("+Refined Depth"), using the RAFT-flow in SLAM front-end ("+RAFT-flow"), and removing the reprojection error term in bundle adjustment ("+Remove BA Term").

We show a complete set of ablation results in Tab. 3. Under the monocular mode, "Our BaseSystem" reduces the absolute relative depth error from 9.9% (by the pretrained DPT model) to 9.0%, which verifies the effectiveness of the basic self-supervised refinement method. However, the improvement brought by our base model is not significant and the SLAM module fails. Using RAFT-flow in SLAM front-end makes SLAM more robust, generating more accurate pose estimation, which in turn improves the depth refinement module. Adding our scale self-alignment in RAFT-SLAM ("+Scale Alignment") improves the depth quality significantly in all metrics, *e.g.*, *Abs Rel* decreases from 8.3% to 6.4% and δ_1 increases from 91.5% to 95.2%. Our occlusion-aware depth consistency loss ("Depth Consistency") further achieves an improvement of 1.1% in terms of *Abs*

Table 4: Quantitative depth evaluation on TUM-RGBD.

Method	Monocular						pRGBD					
	MAE ↓	AbsRel ↓	RMSE ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑	MAE ↓	AbsRel ↓	RMSE ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑
freiburg3_structure_texture_near												
DPT [39]	0.280	0.140	0.529	0.794	0.924	0.968	0.280	0.140	0.529	0.794	0.924	0.968
Ours-DPT	**0.138**	**0.057**	**0.314**	**0.943**	**0.977**	**0.990**	**0.140**	**0.056**	**0.317**	**0.941**	**0.974**	**0.992**
freiburg3_structure_texture_far												
DPT [39]	0.372	0.134	0.694	0.810	0.939	0.968	0.372	0.134	0.694	0.810	0.939	0.968
Ours-DPT	**0.108**	**0.035**	**0.317**	**0.974**	**0.985**	**0.997**	**0.105**	**0.036**	**0.290**	**0.975**	**0.985**	**0.996**

Rel and 1.8% in terms of δ_1. From this ablation study, it is evident that each component of our method makes non-trivial contributions in improving depth results. We can draw a similar conclusion under the pRGBD mode.

4.3 TUM-RGBD Dataset

TUM-RGBD is a well-known dataset mainly for benchmarking performance of RGB-D SLAM or odometry [49]. This dataset was created using a Microsoft Kinect sensor and eight high-speed tracking cameras to capture monocular images, their corresponding depth images, and camera poses. This dataset is particularly difficult for monocular systems as it contains a large amount of motion blur and rolling-shutter distortion caused by fast camera motion. We take two monocular sequences from this dataset, *i.e.*, "freiburg3_structure_texture_near" and "freiburg3_structure_texture_far", to test our system, as they satisfy our system's requirement of sufficient camera translations. The quantitative depth results are presented in Tab. 4. As before, under both SLAM modes, our GeoRefine improves upon the pretrained DPT model by a significant margin, achieving 2-4 times' reduction in terms of *Abs Rel*. A global reconstruction is visualized in Fig. 4, where the scene geometry is faithfully recovered.

5 Conclusions

In this paper, we have introduced *GeoRefine*, an online depth refinement system that combines geometry and deep learning. The core contribution of this work lies in the system design itself, where we show that accurate dense mapping from monocular sequences is possible via a robust hybrid SLAM, an online learning paradigm, and a careful consideration of degenerate cases. The self-supervised nature of the proposed system also suggests that it can be deployed in any unseen environments by virtue of its self-adaptation capability. We have demonstrated the state-of-the-art performance on several challenging public datasets.

Limitations. Our system does not have a robust mechanism for handling moving objects which are outliers both for SLAM and self-supervised losses. Hence, datasets with plenty of foreground moving objects such as KITTI [15] would not be the best test-bed for GeoRefine. Another limitation is that GeoRefine cannot deal with scenarios where camera translations are small over the entire sequence. This constraint is intrinsic to our system design, but it is worth exploring how to relax it while maintaining robustness.

References

1. Bian, J.W., Li, Z., Wang, N., Zhan, H., Shen, C., Cheng, M.M., Reid, I.: Unsupervised scale-consistent depth and ego-motion learning from monocular video. In: NeurIPS (2019)
2. Bloesch, M., Czarnowski, J., Clark, R., Leutenegger, S., Davison, A.J.: Codeslam—learning a compact, optimisable representation for dense visual slam. In: CVPR. pp. 2560–2568 (2018)
3. Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M.W., Siegwart, R.: The euroc micro aerial vehicle datasets. The International Journal of Robotics Research (2016)
4. Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J.: Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on robotics **32**(6), 1309–1332 (2016)
5. Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M., Tardós, J.D.: Orb-slam3: An accurate open-source library for visual, visual-inertial and multi-map slam. arXiv preprint arXiv:2007.11898 (2020)
6. Czarnowski, J., Laidlow, T., Clark, R., Davison, A.J.: Deepfactors: Real-time probabilistic dense monocular slam. IEEE Robotics and Automation Letters **5**(2), 721–728 (2020)
7. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: CVPR. pp. 5828–5839 (2017)
8. Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. ToG **36**(4), 1 (2017)
9. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. arXiv preprint arXiv:1406.2283 (2014)
10. Engel, J., Koltun, V., Cremers, D.: Direct sparse odometry. TPAMI **40**(3), 611–625 (2017)
11. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular slam. In: ECCV (2014)
12. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)
13. Fu, H., Gong, M., Wang, C., Batmanghelich, K., Tao, D.: Deep ordinal regression network for monocular depth estimation. In: CVPR (2018)
14. Garg, R., BG, V.K., Carneiro, G., Reid, I.: Unsupervised cnn for single view depth estimation: Geometry to the rescue. In: ECCV (2016)
15. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. International Journal of Robotics Research (2013)
16. Godard, C., Aodha, O.M., Firman, M., Brostow, G.J.: Digging into self-supervised monocular depth estimation. In: ICCV (2019)
17. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: CVPR. pp. 270–279 (2017)
18. Gordon, A., Li, H., Jonschkowski, R., Angelova, A.: Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In: CVPR (2019)
19. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press (2003)

20. Hermann, M., Ruf, B., Weinmann, M., Hinz, S.: Self-supervised learning for monocular depth estimation from aerial imagery. arXiv preprint arXiv:2008.07246 (2020)
21. Ji, P., Li, R., Bhanu, B., Xu, Y.: Monoindoor: Towards good practice of self-supervised monocular depth estimation for indoor environments. In: ICCV. pp. 12787–12796 (2021)
22. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic gradient descent. In: ICLR. pp. 1–15 (2015)
23. Klein, G., Murray, D.: Parallel tracking and mapping for small ar workspaces. In: ISMAR (2007)
24. Koestler, L., Yang, N., Zeller, N., Cremers, D.: Tandem: Tracking and dense mapping in real-time using deep multi-view stereo. In: CoLR. pp. 34–45 (2022)
25. Kopf, J., Rong, X., Huang, J.B.: Robust consistent video depth estimation. In: CVPR. pp. 1611–1621 (2021)
26. Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g 2 o: A general framework for graph optimization. In: ICRA (2011)
27. Li, Q., Zhu, J., Liu, J., Cao, R., Li, Q., Jia, S., Qiu, G.: Deep learning based monocular depth prediction: Datasets, methods and applications. arXiv preprint arXiv:2011.04123 (2020)
28. Li, S., Wu, X., Cao, Y., Zha, H.: Generalizing to the open world: Deep visual odometry with online adaptation. In: CVPR. pp. 13184–13193 (2021)
29. Li, Z., Dekel, T., Cole, F., Tucker, R., Snavely, N., Liu, C., Freeman, W.T.: Learning the depths of moving people by watching frozen people. In: CVPR. pp. 4521–4530 (2019)
30. Li, Z., Snavely, N.: Megadepth: Learning single-view depth prediction from internet photos. In: CVPR. pp. 2041–2050 (2018)
31. Liu, F., Shen, C., Lin, G., Reid, I.: Learning depth from single monocular images using deep convolutional neural fields. TPAMI **38**(10), 2024–2039 (2015)
32. Liu, J., Ji, P., Bansal, N., Cai, C., Yan, Q., Huang, X., Xu, Y.: Planemvs: 3d plane reconstruction from multi-view stereo. CVPR (2022)
33. Luo, X., Huang, J.B., Szeliski, R., Matzen, K., Kopf, J.: Consistent video depth estimation. TOG **39**(4), 71–1 (2020)
34. Matsuki, H., Scona, R., Czarnowski, J., Davison, A.J.: Codemapping: Real-time dense mapping for sparse slam using compact scene representations. IEEE Robotics and Automation Letters **6**(4), 7105–7112 (2021)
35. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: a versatile and accurate monocular slam system. IEEE Transactions on Robotics **31**(5), 1147–1163 (2015)
36. Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE Transactions on Robotics **33**(5), 1255–1262 (2017)
37. Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M.: Real-time 3d reconstruction at scale using voxel hashing. ToG **32**(6), 1–11 (2013)
38. Qi, X., Liao, R., Liu, Z., Urtasun, R., Jia, J.: Geonet: Geometric neural network for joint depth and surface normal estimation. In: CVPR. pp. 283–291 (2018)
39. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: ICCV. pp. 12179–12188 (2021)
40. Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. arXiv preprint arXiv:1907.01341 (2019)

41. Ranjan, A., Jampani, V., Balles, L., Kim, K., Sun, D., Wulff, J., Black, M.J.: Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In: CVPR. pp. 12240–12249 (2019)
42. Ruhkamp, P., Gao, D., Chen, H., Navab, N., Busam, B.: Attention meets geometry: Geometry guided spatial-temporal attention for consistent self-supervised monocular depth estimation. In: 3DV. pp. 837–847 (2021)
43. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR. pp. 4104–4113 (2016)
44. Schubert, D., Demmel, N., Usenko, V., Stückler, J., Cremers, D.: Direct sparse odometry with rolling shutter. In: ECCV (2018)
45. Shu, C., Yu, K., Duan, Z., Yang, K.: Feature-metric loss for self-supervised learning of depth and egomotion. In: ECCV. pp. 572–588 (2020)
46. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: ECCV (2012)
47. Song, S., Chandraker, M., Guest, C.C.: Parallel, real-time monocular visual odometry. In: ICRA (2013)
48. Stanford Artificial Intelligence Laboratory et al.: Robotic operating system, `https://www.ros.org`
49. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: IROS. pp. 573–580 (2012)
50. Tateno, K., Tombari, F., Laina, I., Navab, N.: CNN-SLAM: Real-time dense monocular slam with learned depth prediction. In: CVPR (2017)
51. Teed, Z., Deng, J.: DeepV2D: Video to depth with differentiable structure from motion. arXiv preprint arXiv:1812.04605 (2018)
52. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: ECCV. pp. 402–419 (2020)
53. Teed, Z., Deng, J.: Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. arXiv preprint arXiv:2108.10869 (2021)
54. Tiwari, L., Ji, P., Tran, Q.H., Zhuang, B., Anand, S., Chandraker, M.: Pseudo rgb-d for self-improving monocular slam and depth prediction. In: ECCV. pp. 437–455 (2020)
55. Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A.: Bundle adjustment—a modern synthesis. Vision Algorithms: Theory and Practice pp. 153–177 (2000)
56. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: DEMON: Depth and motion network for learning monocular stereo. In: CVPR (2017)
57. Wang, C., Buenaposada, J.M., Zhu, R., Lucey, S.: Learning depth from monocular videos using direct methods. In: CVPR. pp. 2022–2030 (2018)
58. Wang, R., Pizer, S.M., Frahm, J.M.: Recurrent neural network for (un-) supervised learning of monocular video visual odometry and depth. In: CVPR (2019)
59. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. TIP **13**(4), 600–612 (2004)
60. Xiong, M., Zhang, Z., Zhong, W., Ji, J., Liu, J., Xiong, H.: Self-supervised monocular depth and visual odometry learning with scale-consistent geometric constraints. In: IJCAI. pp. 963–969 (2021)
61. Yang, N., Stumberg, L.v., Wang, R., Cremers, D.: D3VO: Deep depth, deep pose and deep uncertainty for monocular visual odometry. In: CVPR (2020)
62. Yang, N., Wang, R., Stückler, J., Cremers, D.: Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In: ECCV (2018)

63. Yin, Z., Shi, J.: GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In: CVPR (2018)
64. Zhang, Z., Cole, F., Tucker, R., Freeman, W.T., Dekel, T.: Consistent depth of moving objects in video. ACM TOG **40**(4), 1–12 (2021)
65. Zhao, W., Liu, S., Shu, Y., Liu, Y.J.: Towards better generalization: Joint depth-pose learning without posenet. In: CVPR. pp. 9151–9161 (2020)
66. Zhou, H., Ummenhofer, B., Brox, T.: Deeptam: Deep tracking and mapping. In: ECCV. pp. 822–838 (2018)
67. Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: CVPR. pp. 1851–1858 (2017)
68. Zou, Y., Ji, P., Tran, Q.H., Huang, J.B., Chandraker, M.: Learning monocular visual odometry via self-supervised long-term modeling. In: ECCV (2020)
69. Zou, Y., Luo, Z., Huang, J.B.: DF-Net: Unsupervised joint learning of depth and flow using cross-task consistency. In: ECCV (2018)