Robust Consistent Video Depth Estimation

Johannes Kopf
Facebook

Xuejian Rong
Facebook

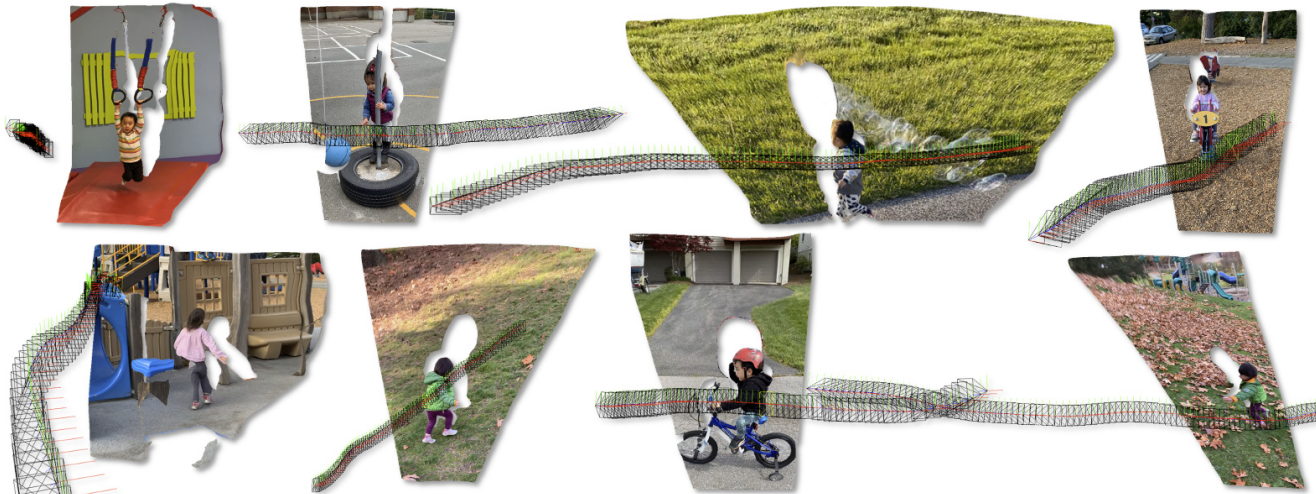Jia-Bin Huang
Virginia Tech

https://robust-cvd.github.io

Figure 1: **Robust consistent video depth estimation of dynamic scenes.** Our method estimates a smooth camera trajectory and detailed and stable dense depth map on challenging hand-held cellphone videos. Our method supports both still (static) and dynamic camera motion.

Abstract

We present an algorithm for estimating consistent dense depth maps and camera poses from a monocular video. We integrate a learning-based depth prior, in the form of a convolutional neural network trained for single-image depth estimation, with geometric optimization, to estimate a smooth camera trajectory as well as detailed and stable depth reconstruction. Our algorithm combines two complementary techniques: (1) flexible deformation-splines for low-frequency large-scale alignment and (2) geometry-aware depth filtering for high-frequency alignment of fine depth details. In contrast to prior approaches, our method does not require camera poses as input and achieves robust reconstruction for challenging hand-held cell phone captures containing a significant amount of noise, shake, motion blur, and rolling shutter deformations. Our method quantitatively outperforms state-of-the-arts on the Sintel benchmark for both depth and pose estimations and attains favorable qualitative results across diverse wild datasets.

1. Introduction

Dense per-frame depth is an important intermediate representation that is useful for many video-based applications, such as 3D video stabilization [37], augmented reality (AR) and special video effects [59, 39], and for converting videos for virtual reality (VR) viewing [22]. However, estimating accurate and consistent depth maps for casually captured videos still remains very challenging. Cell phones contain small image sensors that may produce noisy images, especially in low lighting situations. They use a rolling shutter that may result in wobbly image deformations. Hand-held captured casual videos often contain camera shake and motion blur, and dynamic objects, such as people, animals, and vehicles. In addition to all these degradations, there exist well-known problems for 3D reconstruction that are not specific to video, including poorly textured image regions, repetitive patterns, and occlusions.

Traditional algorithms for dense reconstruction that combine Structure from Motion (SFM) and Multi-view Stereo (MVS) have difficulties dealing with these challenges. The SFM step suffers from the limitations of accuracy and availability of correspondence and often fails

entirely, as explained below, preventing further processing. Even when SFM succeeds, the MVS reconstructions typically contain a significant amount of holes and noises.

Learning-based algorithms [35, 34, 48] are better equipped to handle with this situation. Instead of matching points across frames and geometric triangulation they employ priors learned from diverse training datasets. This enables them to handle many of the challenging situations aforementioned. However, the estimated depth is only defined *up to scale*, and, while plausible, is not necessarily accurate, i.e., it lacks geometric consistency.

Hybrid algorithms [39, 36, 70, 56] achieve desirable characteristics of both approaches by combining learnt priors with geometric reasoning. These methods often assume precise per-frame camera poses as auxiliary inputs, which are typically estimated with SFM. However, SFM algorithms are not robust to the issues described above. In such situations, SFM might fail to register all frames or produce outlier poses with large errors. As a consequence, hybrid algorithms work well when the pose estimation succeeds and fail catastrophically when it does not. This problem of *robustness* makes these algorithms unsuitable for many real-world applications, as they might fail in unpredictable ways. Recently, a hybrid approach proposed in the DeepV2D work [56] attempts to interleave pose and depth estimations in inference for an ideal convergence, which performs reasonably well on static scenes but still does not prove the capability of tackling dynamic scenes.

We present a new algorithm that is more robust and does *not* require poses as input. Similar to Luo et al. [39], we leverage a convolutional neural network trained for single-image depth estimation as a depth prior and optimize the *alignment* of the depth maps. However, their test-time fine-tuning formulation requires a pre-established geometric relationship between matched pixels across frames, which, in turn, requires precisely calibrated camera poses and per-frame depth scale factors. In contrast, we *jointly optimize* extrinsic and intrinsic camera pose parameters as well as *3D alignment* of the estimated depth maps using continuous optimization. Naïve alignment using rigid-scale transformations does not result in accurate poses because the *independently* estimated per-frame depth maps usually contain random inaccuracies. These further lead to misalignment, which inevitably imposes noisy errors onto the estimated camera trajectory. We resolve it by turning to a more flexible deformation model, using spatially-varying splines. They provide a more exact alignment, which, in succession, results in smoother and more accurate trajectories.

The spline-based deformation achieves accurate *low-frequency* alignment. To further improve *high-frequency* details and remove residual jitter, we use a geometry-aware depth filter. This filter is capable of bringing out fine depth details, rather than blurring them because of the precise alignment from the previous stage.

As shown in previous work, the learning-based prior is resilient to moderate amounts of dynamic motion. We make our method even more robust to large dynamic motion by incorporating automatic segmentation-based masks to relax the geometric alignment requirements in regions containing people, vehicles, and animals.

We evaluate our method qualitatively (visually) by processing *all* 90 sequences from the DAVIS dataset (originally designed for dynamic video object segmentation) [46] and comparing against previous methods (of which many fail). We further, evaluate quantitatively on the 23 sequences from the Sintel dataset [4], for which ground truth pose and depth are available.

2. Related Work

Multi-view stereo. Multi-view stereo (MVS) algorithms estimate depth from a collection of images captured from different viewpoints [53, 17]. Geometry-based MVS systems (e.g., COLMAP [52]) follow the incremental Structure-from-Motion (SFM) pipeline (correspondence estimation, pose estimation, triangulation, and bundle adjustment). Several learning-based methods further improve the reconstruction quality by fusing classic MVS techniques (e.g., cost aggregation and plane-sweep volume) and data-driven priors [58, 24, 68, 25, 28]. In contrast to MVS algorithms that assume a *static* scene, our work aims to reconstruct fully dense depth from a dynamic scene video.

Single-image depth estimation. In recent years we have witnessed rapid progresses on *supervised* learning-based single-image depth estimation [13, 12, 30, 38, 16]. As diverse training images with the corresponding ground truth depth maps are difficult to obtain, existing work explores training models using synthetic datasets [40], crowd-sourced human annotations of relative depth [7] or 3D surfaces [9], pseudo ground truth depth maps from internet images/videos [35, 34, 8], or 3D movies [48, 61]. Another research line focuses on *self-supervised* approaches for learning single-image depth estimation models. Specific examples include learning from stereo pairs [18, 21, 20, 64] or monocular videos [73, 60, 11, 75, 69, 47, 50]. Most self-supervised learning methods minimize photometric reprojection errors (computed from the estimated depth and pose) and do not account for dynamic objects in videos. Several methods alleviate this problem by masking out dynamic objects [75], modeling the motion of individual objects [5] or estimating dense 3D translation field [32]. We use the state-of-the-art single-image depth estimation method [48] to obtain an initial dense depth map for each video frame. While these depth maps are plausible when viewed individually, they are *not* geometrically consistent across different frames. Our work aims to produce geometrically consistent camera poses and dense depth for a video.

Video-based depth estimation. Several methods integrate camera motion estimation and multi-view reconstruction from a pair of frames [58] or multiple frames [72, 2, 59]. However, these methods work well only on a static scene. To account for moving objects, a line of work use motion segmentation [26, 49] or semantic instance segmentation [5] to help constrain the depth of moving objects. State-of-the-art learning-based video depth estimation approaches can be grouped into two tracks: (1) MVS-based methods and (2) hybrid methods. MVS-based methods improve the conventional SFM and MVS workflow using differentiable pose/depth modules [56] or explicit modeling depth estimation uncertainty [36]. Both methods [36, 56] estimate depth based on the cost volume constructed by warping nearby frames to a reference viewpoint. These methods may produce erroneous depth estimation and fail to generate accurate camera trajectories for dynamic scenes. Hybrid methods integrate single-view depth estimation models and multi-view stereo for achieving *geometrically consistent* video depth, either through fusing depth predictions from single-view and multi-view [70] or fine-tuning single-view depth estimation model to satisfy geometric consistency [39]. While impressive results have been shown, the methods [70, 39] assume that precise camera poses are available as input and thus are not applicable for challenging sequences where existing SFM/MVS systems fail. Our method also leverages a pretrained single-view depth estimation model. Unlike [70, 39], we jointly optimize camera poses and 3D deformable alignment of the depth maps and thus can handle a broader range of challenging videos of highly dynamic scenes.

Visual odometry. Visual Odometry (VO) or Simultaneously Localization And Mapping (SLAM) aim to estimate the relative camera poses from image sequences [43, 51]. Conventional geometry-based methods [15, 14, 41, 49, 42] can be grouped into the four categories depending on using *direct* (feature-less) vs. *indirect* (feature-based) methods and *dense* or *sparse* reconstruction. While significant progress has been made, applying VO and SLAM for generic scenes remains challenging [67]. Recent years, numerous learning-based approaches have been proposed to tackle these challenges either via supervised [58, 62, 72, 27] or self-supervised learning [73, 33, 71, 19, 54, 65, 74, 66]. Similar to the existing VO/SLAM methods, our work also estimates both camera poses from a monocular video. Unlike prior work, our primary focus lies in estimating geometrically consistent dense depth reconstruction for *dynamic scenes*.

Temporal consistency. Per-frame processing often leads to temporal flickering results. Enforcing temporal consistency of a output video has been explored in many different applications, including style transfer [6], video completion [23], video synthesis [63], or as post-processing
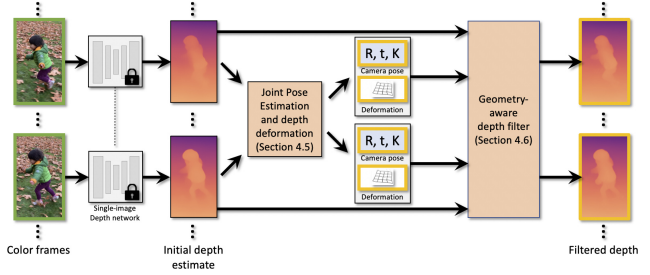

Figure 2: **Overview.** Our algorithm only takes a monocular color video as input. We first estimate per-frame depth maps using an existing single-frame CNN. We jointly optimize camera poses as well as flexible deformations to align the depth maps in 3D and resolve any large-scale misalignments. Finally, we resolve fine-scale details using a geometry-aware depth filter. Green frames: inputs; yellow frames: outputs.

step [31, 3, 29]. For video depth estimation, temporal consistency can either be explicitly constrained by optical flow [26] or implicitly applied using recurrent neural networks [44]. Our 3D depth filter is similar to that of [31] because we also filter the depth maps *across* time along the flow trajectory. Our approach differs in that our method is *geometry-aware*.

3. Overview

Our approach builds on the formulation established in the Consistent Video Depth Estimation (CVD) paper by Luo et al. [39], so let us start by recapping it, first. They iteratively fine-tune the weights of a CNN trained for single-image depth estimation until it learns to satisfy the geometry of a particular scene in question. To assess the progress against this goal, they relate pairs of images geometrically using *known* camera parameters (extrinsic and intrinsic, as well as per-frame depth scale factors). More precisely, their algorithm compares the *3D reprojection* of pixels from one image to the other with the corresponding *image-space motion*, computed by an optical flow method. The reprojection error is back-propagated to the network weights, so that it reduces over the course of the fine-tuning (see details in Section 4.2). This results in a very detailed and temporally consistent, i.e., flicker-free, depth video.

However, one key limitation of their approach is that *precise* camera parameters are needed as input, which are computed with SFM in their case. Unfortunately, SFM for video is a challenging problem in itself, and it frequently fails; for example, when a video does not contain sufficient camera motion, or in the presence of dynamic object motion, or in numerous other situations, as we explained earlier. In such cases, it might either fail entirely to produce an output, or fail to register a subset of the frames, or it might produce erroneous (outlier) camera poses. Inaccurate poses have a

strong degrading effect on the CVD optimization, as shown in their paper [39]. The key contribution of our paper is to remove this limitation, by replacing the test-time fine-tuning with *joint optimization* of the camera parameters and depth alignment.

We show in Section 4.3 that the same formulation can optimize the camera poses as in CVD. However, one complication is that the pose optimization only works well when we have precise depth, similarly to how depth fine-tuning only works when the poses are accurate (Section 4.4). If the depth is not accurate, which is the case in our setting, *misalignments* in the depth impose themselves as noisy errors onto the resulting camera pose trajectory.

We resolve this problem by improving the ability of the camera optimization to align the depth estimates, despite their initial inaccuracy (Section 4.5). Specifically, we achieve this by replacing the per-frame camera scale with a more flexible spatially-varying transformation, i.e., a bilinear spline. The improved alignment of the depth estimates enables computing smoother and more accurate pose trajectories.

The joint pose estimation and deformation resolves *low-frequency* inconsistencies in the depth maps. We further improve *high-frequency* alignment using a geometry-aware depth filter (Section 4.6). This filter low-pass filters the *reprojected* depth along flow trajectories. Because the input to the filter is well-aligned, due to the deformation, the filter resolves fine details, rather than blurring them.

4. Method

4.1. Pre-processing

We share some of the preprocessing steps with CVD [39]. However, importantly, we do *not* need to compute COLMAP [52], which considerably improves the robustness of our method.

In order to lower the overall amount of computation in the pairwise optimization below, we subsample a set of frame pairs spanning temporally near and distant frames:

$$P = \left\{ (i,\ j) \,\middle|\, |i-j| = k,\ i \bmod k = 0,\ k = 1, 2, 4, \dots \right\}. \quad (1)$$

For each frame pair $(i, j) \in P$ we compute a dense optical flow field $f_{i \to j}$ (mapping a pixel in frame i to its corresponding location in frame j using RAFT [57]), as well as a binary mask $m_{i \to j}^{flow}$ that indicates forward-backward consistent flow pixels. Please refer to [39] for details about these preprocessing steps. We also compute a binary segmentation mask m_i^{dyn} using Mask R-CNN that indicates likely-dynamic pixels (belonging to the categories "person", "animal", or "vehicle").

4.2. Depth Estimation

In this section, we establish CVD [39] from a technical point of view and some notation, and in the subsequent sections, we will then present our method.

Let p be a 2D pixel coordinate. We can lift it into a 3D coordinate $c_i(p)$ in frame i's *3D camera coordinate* system:

$$c_i(p) = s_i\, d_i(p)\, \tilde{p}. \quad (2)$$

Here, s_i is the per-frame scale coefficient, and d_i is the CNN-estimated depth map, and $\tilde{p}$ is the homogeneous-augmented pixel coordinate, i.e., $[p_x, p_y, 1]^\top$.

We can also project this 3D point into the camera coordinate system of *another* frame j:

$$c_{i \to j}(p) = K_j R_j^\top \left(R_i K_i^{-1} c_i(p) + t_i - t_j \right) \quad (3)$$

Here, R_i, R_j and t_i, t_j and K_i, K_j are the rotation, translation, and intrinsics of frames i and j, respectively.

The objective that CVD optimizes is a *reprojection loss* for every pixel (with valid flow) in every frame pair:

$$\underset{\theta^{depth}}{\mathrm{argmin}} \sum_{(i,j) \in P} \sum_{p \in m_{i \to j}^{flow}} \mathcal{L}_{i \to j}^{repro}(p), \quad \text{s.t. fixed } \theta^{cam} \quad (4)$$

The optimization variables, θ^{depth}, are the network weights of the depth CNN, and the fixed camera parameters are $\theta^{cam} = \{R_i, t_i, K_i, s_i\}$.

The reprojection loss is defined as follows:

$$\mathcal{L}_{i \to j}^{repro}(p) = \mathcal{L}_j^{sim}\Big(\underbrace{c_{i \to j}(p)}_{\text{3D-reprojection}},\ \underbrace{c_j\big(f_{i \to j}(p)\big)}_{\text{Flow-reprojection}} \Big), \quad (5)$$

i.e., it reprojects the pixel into the other frame's 3D camera coordinate system using (1) 3D geometry and (2) optical flow and measures the similarity of the two resulting 3D points. The exact form of the reprojection similarity loss $\mathcal{L}^{sim}$ is not critical for the overall understanding of the algorithm, so we will defer its definition to below.

4.3. Pose Optimization

As mentioned before, the camera parameters θ^{cam} are needed for the geometric reprojection mechanics, and it is critically important that they are precise. Otherwise, the depth optimization converges to poor results (Figure 3d). It would be desirable to have a more reliable way to obtain poses for our application than with SFM.

When examining Eq. 4 we notice that we can actually use this equation to compute the camera parameters if we reverse the role of θ^{depth} and θ^{cam}, i.e., *fixing θ^{depth}* (assuming that we know them) and *optimizing θ^{cam}*. This modified equation resembles the triangulation in bundle adjustment, but it can be more robustly solved since the depth of
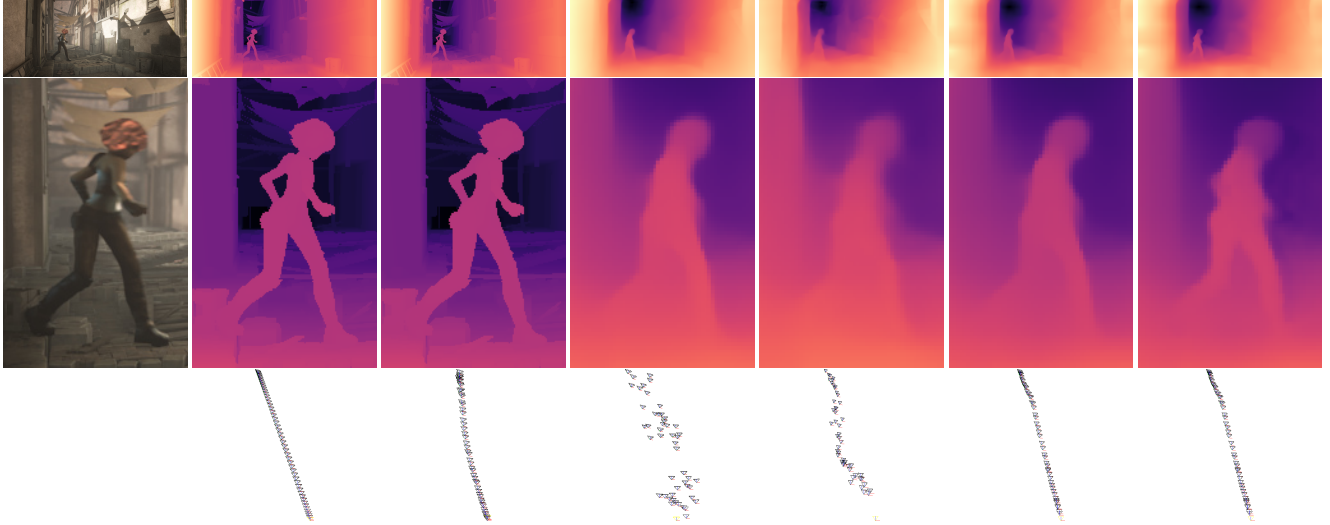
Depth:	Ground truth	Ground truth	Estimated [48]	Estimated [48]	Estimated [48]	**Estimated [48]**
Pose optimization:	Ground truth	Single scale (s_i)	Single scale (s_i)	Single scale (s_i)	Flexible (φ_i)	**Flexible (φ_i)**
Depth refinement:	N/A	N/A	N/A	Fine-tuning	N/A	**Depth filter**
	(a)	(b)	(c)	(d)	(e)	**(f)**

Figure 3: **Joint depth and pose optimization.** Various configurations of our algorithm: (a-b) Ground truth depth with ground truth and estimated poses, respectively. (c) Misalignments in estimated depth impose jittery errors on the optimized camera trajectories. (d) CVD-style fine-tuning fails in the absence of precise poses. (e) Our flexible deformations resolve depth misalignments, which results in smoother camera trajectories. (f) Using geometry-aware depth filtering we resolve fine depth details (our final result).

matched image points does not need to be estimated since it is known (up to scale). However, this time it is the *depth* that needs to be known precisely for this to work well. In the depth maps do not agree with each other, these misalignment errors will manifest as noisy errors in the estimated camera parameters (Figure 3c).

4.4. Pose and Depth Optimization with Fine-tuning

What about optimizing *both* quantities, θ^{cam} and θ^{depth}, jointly? One problem is that these two quantities are best optimized with different kinds of machinery. θ^{depth} is best optimized using standard training algorithms for CNNs, i.e., SGD. For θ^{cam}, however, that is not a good fit, since changes to one parameters have far-reaching influence, as the poses are chained in a trajectory. Global continuous optimization is a better solution for θ^{cam} and converges faster and more stably. We can optimize both quantities by alternating between optimizing depth and pose, each with their respective best optimization algorithm while keeping the other quantity fixed.

However, another significant problem is the sensitivity to the accuracy of the particular fixed parameters, as explained before. Starting with the initially inaccurate depth estimate will result in noisy poses (Figure 3c), because the misalignment errors will "push" the camera pose variables in erratic ways. The jittery poess will, in the next step, degrade the

depth estimate further. The algorithm does not converge to a good solution (Figure 3d).

4.5. Pose and Depth with Flexible Deformation

Our solution to this apparent dilemma is to improve the *depth alignment* in the pose estimation. We achieve this by injecting a smooth and flexible spatially-varying *depth deformation model* into the alignment. More precisely, we replace the depth scale coefficients s_i in Eq. 4 with a spatially-varying bilinear spline:

$$\varphi_i(p) = \sum_k b_k(p)\, s_i^k. \tag{6}$$

Here, the s_i^k are scale factors that are defined on a regular grid of "handles" across the image. $b_k(p)$ are bilinear coefficients, such that within a grid cell the four surrounding handles of a pixel p are bilinearly interpolated.

After this change the depth maps align better and will not impose jittery errors onto the estimated camera trajectory anymore (Figure 3e). In addition, this algorithm is considerably faster since we do not need to iterate between pose optimization and fine-tuning.

4.6. Geometry-aware Depth Filtering

The flexible deformation φ_i achieves a low-frequency alignment of the depth map, i.e., removing any large-scale
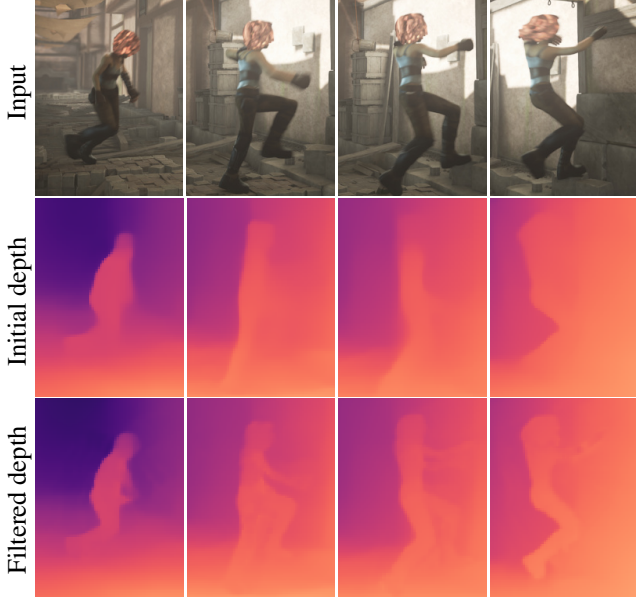
Figure 4: **Geometry-aware depth filter.** Top: frames from the input video. Middle: initial per-frame depth estimates (after flexible alignment). Bottom: the geometry-aware depth filter resolves fine-scale details in the depth maps.

misalignments. But what about fine depth details? We tried using alternating pose-depth optimization, as described in Section 4.4, with flexible deformation. This works and improves both the depth and poses slightly, but it does not reach the same level of quality that CVD achieves when using precise SFM pose as input. Instead, it converges quickly to a configuration where both depth and pose alignment are well-satisfied, but depth details smooth out considerably.

We solve this problem instead with a spatio-temporal depth filter that follows flow trajectories. Importantly, the filter is geometry-aware in the sense that it transforms the depths from other frames using the reprojection mechanics in Equations 2-3:

$$d_i^{final}(p) = \sum_{q \in N(p)} \sum_{j=i-\tau}^{i+\tau} z_{j \to i}\left(\tilde{f}_{i \to j}(q)\right) w_{i \to j}(q). \quad (7)$$

Here, $N(p)$ refers to a 3×3 neighborhood centered around the pixel p, $\tau = 4$, $z_{j \to i}$ is the scalar z-component of $c_{j \to i}$ (i.e., the reprojected depth), and $\tilde{f}$ is the flow between far frames obtained by *chaining* flow maps between consecutive frames. The weights $w_{i \to j}(q)$ make the filter edge-preserving by reducing the influence os samples across depth edges:

$$w_{i \to j}(q) = \exp\left(-\lambda_f \mathcal{L}^{ratio}\left(c_i(p), c_{j \to i}\left(\tilde{f}_{i \to j}(q)\right)\right)\right). \quad (8)$$

$\lambda_f = 3$ adjusts the strength of the filter, and $\mathcal{L}^{ratio}$ measures

the similarity of the reference and the reprojected pixel in camera coordinates (the definition is given in the implementation details below).

The improved provided by the depth filter can be seen in Figure 3f, and more results in Figure 4.

4.7. Implementation Details

Precomputation. For MiDaS we downscale the image to 384 pixels on the long side (the default resolution in their code). For RAFT we downscale the image to 1024 pixels on the long side, to lower memory requirements. In both cases we adjust the short image side according to image aspect ratio, while rounding to the nearest multiple of 16 pixels which is the alignment requirement of the respective CNNs. We use the pretrained `raft-things.pth` from the RAFT project page [57], which was trained on FlyingChairs and FlyingThings.

Pose optimization. We use the `SPARSE_NORMAL_CHO-LESKY` solver from the Ceres library [1] to solve the camera pose estimation problem. In order to reduce the computational complexity, we do not include every pixel in the optimization but instead, subsample a set of pairwise matches from the flow fields (so that there is a minimum distance of at least 10 pixels between any pair of matches). Since Eq. 4 assumes a static scene, we exclude any points matches within the m^{dyn} mask.

Since the objective is non-convex optimization, it is somewhat sensitive to local minima in the objective. We alleviate that problem by first optimizing a 1×1 grid (similar to the original s_i scalar coefficients), and then subdividing it in four steps until a grid resolution of 17×10 is reached (always using 17 for the long image dimension, and adjusting the short image dimension according to the aspect ratio). After every step, we optimize until convergence and use the result as initialization for the next step.

Since the scale of the depth maps estimated by the CNN is arbitrary, we initialize the scale of the first frame so that the median depth is 1, and use the same scale for all other frames.

Pose regularization. To encourage smoothness in the deformation field, we add a loss that penalizes large differences in neighboring grid values:

$$\mathcal{L}^{deform} = \sum_i \sum_{(k,r) \in N} \left\| s_i^k - s_i^r \right\|_2^2 \max(w_i^k, w_i^r), \quad (9)$$

where N refers to the set of all vertically and horizontally neighboring vertices. The weights are set to encourage more smoothness in parts of the image that are masked by m^{dyn}, since there are no point matches in these regions and they are unconstrained otherwise.

$$w_i^k = \lambda_1 + \lambda_2 \sum_p m_i^{dyn}(p) b_k(p) \quad (10)$$

$\lambda_1 = 0.1$, $\lambda_2 = 10$ are balancing coefficients. The λ_2 term computes the fraction of dynamic pixels in the handle's influence region.

We use the following form for the intrinsic matrices:

$$K_i = \begin{bmatrix} u_i & & \\ & u_i & \\ & & 1 \end{bmatrix}, \qquad (11)$$

i.e., the only degree of freedom is the focal length u_i.

We, further, add a small bias,

$$\mathcal{L}^{focal} = \sum_i (u_i - \hat{u})^2, \qquad (12)$$

where $\hat{u} = 0.35$ (corresponds to $\sim 40°$ field of view).

Reprojection loss. In Section 4.2 we omitted the definition of the reprojection similarity loss $\mathcal{L}^{sim}$. A naïve way would be to simple measure the Euclidean distance

$$\mathcal{L}^{euclidean}(a, b) = \|a - b\|_2^2. \qquad (13)$$

However, this biases the solution toward small depths. Shrinking the whole scene to a point would achieve a minimum.

To prevent this, Luo et al. [39] use a split loss where they measure the spatial component $\mathcal{L}^{spatial}$ in image space

$$\mathcal{L}^{spatial}(a, b) = \left\| \frac{a_{xy}}{a_z} - \frac{b_{xy}}{b_z} \right\|_2^2, \qquad (14)$$

and the depth $\mathcal{L}^{depth}$ component in disparity space.

$$\mathcal{L}^{disparity}(a, b) = \left| \frac{1}{a_z} - \frac{1}{b_z} \right|_2^2. \qquad (15)$$

The disparity loss actually has the opposite bias of Euclidean loss: it is minimized when scene scale grows very large (so that the disparities vanish). This is not a problem for Luo et al., since they use fixed poses. However, it affects our results negatively.

To alleviate this, we propose a new loss that measures the *ratio* of depth values:

$$\mathcal{L}^{ratio}(a, b) = \frac{\max(a_z, b_z)}{\min(a_z, b_z)} - 1. \qquad (16)$$

This loss does not suffer from any depth bias; it does neither encourage growing nor shrinking the scene scale. The measure $\mathcal{L}^{ratio}$ is also used to compute the depth-similarity of samples in the depth filter in Eq. 8.

In summary, we define the reprojection similarity loss as follows:

$$\mathcal{L}^{sim}(a, b) = \mathcal{L}^{spatial}(a, b) + \mathcal{L}^{ratio}(a, b) \qquad (17)$$

5. Experimental Results

5.1. Experimental setup

Datasets. We validate the effectiveness of the proposed method on three main video datasets, covering a wide range of challenging indoor and outdoor scenes. Here we focus on reporting results on the Sintel [4] dataset. The MPI Sintel dataset consists of 23 synthetic sequences of highly dynamic scenes. Each sequence comes with ground truth depth measured in meters as well as ground truth camera poses. We use both the *clean* and *final* versions of the dataset. The ground truth annotations allow us to conduct a quantitative comparison on both the estimated depth and pose. We refer the readers to our qualitative results on **DAVIS** [45] and **Cellphone videos** in the project page.

Note that we do not choose datasets focusing on *closed-domain applications* such as driving scenes (e.g., KITTI depth/odometry dataset or office (e.g., TUM RGB-D [55]) or *fully static* scenes (e.g., ScanNet [10]).

Compared methods. We compare our results with several state-of-the-art depth estimation algorithms.

- **COLMAP [52]**: Traditional SFM/MVS algorithm for 3D reconstruction. As COLMAP reconstruction is very sensitive to dynamic objects in the scene, we use the same m^{dyn} dynamic masks using by our algorithm (computed automatically Mask R-CNN) to exclude feature extraction/matching from those areas.

- **DeepV2D [56]**: Video depth estimation algorithm using differentiable motion estimation and depth estimation.

- **CVD [39]**: A hybrid method that combine multi-view and single-view depth estimation methods for producing consistent video depth.

- **MiDaS-v2 [48]**: State-of-the-art single-view depth estimation model.

5.2. Evaluation on MPI Sintel dataset [4]

Depth evaluation. As depth estimation from all the methods is up to an unknown scene scale, we follow the standard depth evaluation protocol to align the predicted depth and the ground truth depth using median scaling. We exclude depth values that are larger than 80 meters. Table 1 shows the quantitative comparisons with the state-of-the-art on various metrics.

Note that COLMAP [52] fails to estimate the camera pose for 11 in 23 sequences. CVD [39], which uses COLMAP poses as input, thus, also does not produce depth estimation results for these scenes. Furthermore, COLMAP discards single-pixel depth estimates deemed unreliable. As a result, we cannot evaluate these methods using standard *averaged* metrics. Instead, we store *all* the pixel-wise error metrics across all pixels, frames, and videos. We then

Table 1: **Quantitative evaluations of depth and pose on the MPI Sintel benchmark** (Top: *Sintel Clean*, Bottom: *Sintel Final*). For depth evaluation, we present per-frame evaluations on standard error and accuracy metrics. For pose evaluation, we present per-sequence evaluations on translational and rotational error metrics.

Method	Depth – Error metric↓				Depth – Accuracy metric↑			Pose – Error metric↓		
	Abs Rel	Sq Rel	RMSE	log RMSE	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	ATE (m)↓	RPE Trans (m)↓	RPE Rot (deg)↓
DeepV2D [56]	0.526	3.629	6.493	0.683	0.487	0.671	0.761	0.9526	0.3819	0.1869
Ours - Single-scale pose (aligned MiDaS)	0.380	2.617	5.773	0.533	0.562	0.736	0.832	0.1883	0.0806	0.0262
Ours - Single-scale pose + depth fine-tuning	0.472	3.444	6.340	0.635	0.534	0.694	0.790	0.1686	0.0724	0.0139
Ours - Single-scale pose + depth filter	**0.375**	**2.546**	**5.763**	**0.530**	**0.569**	0.738	0.835	0.1882	0.0806	0.0262
Ours - Flexible pose	0.379	2.702	5.795	0.533	0.565	0.744	0.836	0.1843	0.0723	0.0095
Ours - Flexible pose + depth fine-tuning	0.439	3.100	6.213	0.614	0.524	0.698	0.796	**0.1656**	**0.0651**	**0.0070**
Ours - Flexible pose + depth filter	0.377	2.657	5.786	0.531	0.568	**0.745**	**0.837**	0.1843	0.0723	0.0095
DeepV2D	0.526	3.620	6.470	0.670	0.486	0.674	0.760	0.9192	0.5834	0.2506
Ours - Single-scale pose (aligned MiDaS)	0.425	2.640	5.858	0.559	0.529	0.726	0.828	0.2210	0.0827	0.0258
Ours - Single-scale pose + depth fine-tuning	0.473	3.215	6.298	0.639	0.527	0.684	0.782	0.1620	0.0727	0.0116
Ours - Single-scale pose + depth filter	0.421	**2.616**	5.850	**0.556**	**0.533**	0.728	0.830	0.1803	0.0827	0.0258
Ours - Flexible pose	0.421	2.660	5.906	0.559	0.523	**0.730**	0.832	0.1831	0.0713	0.0088
Ours - Flexible pose + depth fine-tuning	0.438	3.053	6.300	0.605	0.525	0.705	0.807	**0.1594**	**0.0652**	**0.0073**
Ours - Flexible pose + depth filter	**0.419**	2.628	5.896	0.558	0.526	**0.730**	**0.833**	0.1831	0.0714	0.0088

sort these errors and plot the curves. Please refer to the supplemental material. The plots capture both the *accuracy* and *completeness* of each method.

Pose evaluation. We follow the standard evaluation protocol of visual odometry for pose evaluation and compare our methods against state-of-the-arts in terms of absolute trajectory error (ATE) and relative pose error (RPE). ATE measures the root-mean-square error between predicted camera poses $[x, y, z]$ and ground truth. RPE measures frame-to-frame relative pose error between frame pairs, including translation error (RPE-T) and rotational error (RPE-R). Still, since the scene scale is unknown, we scale and align the predictions to the ground truth associated poses during evaluation by minimizing ATE for fair comparisons.

We conduct pose evaluations on Sintel with ground truth pose annotations, and the quantitative results are presented in Table 1 and Figure 5. As demonstrated in Table 1, our proposed method outperforms MiDaS-v2 on all metrics (mean of ATE, RPE-T, and RPE-R) with a noticeable margin and significantly outperforms DeepV2D, on both the Clean and Final categories of Sintel. Note that since COLMAP and CVD fail on a significant portion of the dynamic scenes of Sintel, it is not directly comparable in terms of mean errors as in Table 1. For a fair comparison, we store RPE-T and RPE-R errors from all plausible pose predictions between frame pairs from all Sintel sequences, then sort them and plot the curves in Figure 5. The results show our proposed method consistently achieves more accurate pose predictions than completing methods.

5.3. Limitations

The main limitation that we observe is a kind of residual wobble of the aligned depth maps. It is apparent in most results we provide on the project page. We think that it can be addressed by better deformation models, in particular, replacing the spline-based deformation with a pixel-based de-
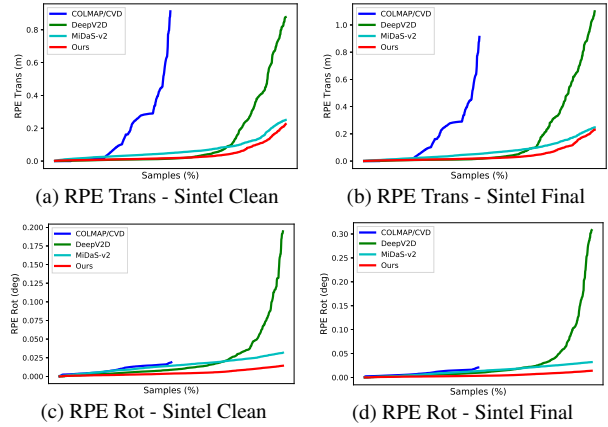

(a) RPE Trans - Sintel Clean

(b) RPE Trans - Sintel Final

(c) RPE Rot - Sintel Clean

(d) RPE Rot - Sintel Final

Figure 5: **Evaluation of translational and rotational Relative Pose Error (RPE) on Sintel**. All the frame pair-wise errors are stored and sorted for plotting the distributions. Note that COLMAP and CVD (which relies on COLMAP) fail on many Sintel sequences, resulting in partial data points.

formation field with appropriate regularization. However, this would require denser pairwise constraints, which our current formulation using continuous global optimization does not support, i.e., when we densify the constraints the Ceres Solver memory usage blows up and the performance goes down drastically. This is due to the global nature of the optimization problem. Finding a better formulation to resolve this problem is a great avenue for future work.

6. Conclusions

We presented a general optimization algorithm for consistent depth estimation on monocular videos, requiring neither input poses nor inference-time fine-tuning. Our method attains robust reconstruction for challenging dynamic videos casually captured by hand-held devices, and achieves better performances on diverse test beds.

References

[1] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. http://ceres-solver.org. 6

[2] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *CVPR*, 2018. 3

[3] Nicolas Bonneel, James Tompkin, Kalyan Sunkavalli, De-qing Sun, Sylvain Paris, and Hanspeter Pfister. Blind video temporal consistency. *ACM Transactions on Graphics (TOG)*, 34(6):196, 2015. 3

[4] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation (Sintel movie © copyright Blender Foundation, www.sintel.org). In *ECCV*, 2012. 2, 7

[5] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *AAAI*, 2019. 2, 3

[6] Dongdong Chen, Jing Liao, Lu Yuan, Nenghai Yu, and Gang Hua. Coherent online video style transfer. In *ICCV*, 2017. 3

[7] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. In *NeurIPS*, 2016. 2

[8] Weifeng Chen, Shengyi Qian, and Jia Deng. Learning single-image depth from videos using quality assessment networks. In *CVPR*, 2019. 2

[9] Weifeng Chen, Shengyi Qian, David Fan, Noriyuki Kojima, Max Hamilton, and Jia Deng. Oasis: A large-scale dataset for single image 3d in the wild. In *CVPR*, 2020. 2

[10] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 7

[11] Qi Dai, Vaishakh Patil, Simon Hecker, Dengxin Dai, Luc Van Gool, and Konrad Schindler. Self-supervised object motion and depth estimation from video. *arXiv preprint arXiv:1912.04250*, 2019. 2

[12] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 2

[13] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NeurIPS*, 2014. 2

[14] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017. 3

[15] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *ECCV*, 2014. 3

[16] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018. 2

[17] Yasutaka Furukawa, Carlos Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015. 2

[18] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 2

[19] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *ICCV*, 2019. 3

[20] Ariel Gordon, Hanhan Li, Rico Jonschkowski, and Anelia Angelova. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In *ICCV*, 2019. 2

[21] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy Ren, and Xiaogang Wang. Learning monocular depth by distilling cross-domain stereo networks. In *ECCV*, 2018. 2

[22] Jingwei Huang, Zhili Chen, Duygu Ceylan, and Hailin Jin. 6-dof vr videos with a single 360-camera. In *2017 IEEE Virtual Reality (VR)*, 2017. 1

[23] Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. Temporally coherent completion of dynamic video. *ACM TOG (Proc. SIGGRAPH Asia)*, 35(6):1–11, 2016. 3

[24] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. DeepMVS: Learning multi-view stereopsis. In *CVPR*, 2018. 2

[25] Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. DPSNet: end-to-end deep plane sweep stereo. In *ICLR*, 2019. 2

[26] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth transfer: Depth extraction from video using non-parametric sampling. *TPAMI*, 36(11):2144–2158, 2014. 3

[27] Jatavallabhula Krishna Murthy, Soroush Saryazdi, Ganesh Iyer, and Liam Paull. gradslam: Dense slam meets automatic differentiation. *arXiv*, 2020. 3

[28] Uday Kusupati, Shuo Cheng, Rui Chen, and Hao Su. Normal assisted stereo depth estimation. In *CVPR*, 2020. 2

[29] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *ECCV*, 2018. 3

[30] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV)*, 2016. 2

[31] Manuel Lang, Oliver Wang, Tunc Aydin, Aljoscha Smolic, and Markus Gross. Practical temporal consistency for image-based graphics applications. *ACM TOG (Proc. SIGGRAPH)*, 31(4):1–8, 2012. 3

[32] Hanhan Li, Ariel Gordon, Hang Zhao, Vincent Casser, and Anelia Angelova. Unsupervised monocular depth learning in dynamic scenes. In *CoLR*, 2020. 2

[33] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *ICRA*, 2018. 3

[34] Zhengqi Li, Tali Dekel, Forrester Cole, Richard Tucker, Noah Snavely, Ce Liu, and William T Freeman. Learning the depths of moving people by watching frozen people. In *CVPR*, 2019. 2

[35] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 2

[36] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa G Narasimhan, and Jan Kautz. Neural rgb (r) d sensing: Depth and uncertainty from a video camera. In *CVPR*, 2019. 2, 3

[37] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving warps for 3d video stabilization. *ACM TOG (Proc. SIGGRAPH)*, 28(3), 2009. 1

[38] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian Reid. Learning depth from single monocular images using deep convolutional neural fields. *TPAMI*, 38(10):2024–2039, 2015. 2

[39] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM TOG (Proc. SIGGRAPH)*, 39(4), 2020. 1, 2, 3, 4, 7

[40] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 2

[41] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. 3

[42] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *ICCV*, 2011. 3

[43] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *CVPR*, 2004. 3

[44] Vaishakh Patil, Wouter Van Gansbeke, Dengxin Dai, and Luc Van Gool. Don't forget the past: Recurrent depth estimation from monocular video. *arXiv preprint arXiv:2001.02613*, 2020. 3

[45] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 7

[46] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 2

[47] Xiaojuan Qi, Renjie Liao, Zhengzhe Liu, Raquel Urtasun, and Jiaya Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *CVPR*, 2018. 2

[48] René Ranftl, Katrin Lasinger, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *TPAMI*, 2020. 2, 5, 7

[49] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *CVPR*, 2016. 3

[50] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *CVPR*, 2019. 2

[51] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, 18(4):80–92, 2011. 3

[52] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2, 4, 7

[53] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, 2006. 2

[54] Lu Sheng, Dan Xu, Wanli Ouyang, and Xiaogang Wang. Unsupervised collaborative learning of keyframe detection and visual odometry towards monocular deep slam. In *ICCV*, 2019. 3

[55] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *International Conference on Intelligent Robot Systems (IROS)*, 2012. 7

[56] Zachary Teed and Jia Deng. DeepV2D: Video to depth with differentiable structure from motion. In *ICLR*, 2020. 2, 3, 7, 8

[57] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 4, 6

[58] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *CVPR*, 2017. 2, 3

[59] Julien Valentin, Adarsh Kowdle, Jonathan T Barron, Neal Wadhwa, Max Dzitsiuk, Michael Schoenberg, Vivek Verma, Ambrus Csaszar, Eric Turner, Ivan Dryanovski, et al. Depth from motion for smartphone ar. *ACM Transactions on Graphics (TOG)*, 37(6):1–19, 2018. 1, 3

[60] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017. 2

[61] Chaoyang Wang, Simon Lucey, Federico Perazzi, and Oliver Wang. Web stereo video supervision for depth prediction from dynamic scenes. In *3DV*, 2019. 2

[62] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *ICRA*, 2017. 3

[63] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *NeurIPS*, 2018. 3

[64] Jamie Watson, Michael Firman, Gabriel J Brostow, and Daniyar Turmukhambetov. Self-supervised monocular depth hints. In *ICCV*, 2019. 2

[65] Fei Xue, Xin Wang, Shunkai Li, Qiuyuan Wang, Junqiu Wang, and Hongbin Zha. Beyond tracking: Selecting memory and refining poses for deep visual odometry. In *CVPR*, 2019. 3

[66] Nan Yang, Lukas von Stumberg, Rui Wang, and Daniel Cremers. D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry. In *CVPR*, 2020. 3

[67] Nan Yang, Rui Wang, Xiang Gao, and Daniel Cremers. Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect. *IEEE Robotics and Automation Letters*, 3(4):2878–2885, 2018. 3

[68] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSNet: Depth inference for unstructured multi-view stereo. In *ECCV*, 2018. 2

[69] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, 2018. 2

[70] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *CVPR*, 2020. 2, 3

[71] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *CVPR*, 2018. 3

[72] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. Deeptam: Deep tracking and mapping. In *ECCV*, 2018. 3

[73] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 2, 3

[74] Yuliang Zou, Pan Ji, Quoc-Huy Tran, Jia-Bin Huang, and Manmohan Chandraker. Learning monocular visual odometry via self-supervised long-term modeling. In *ECCV*, 2020. 3

[75] Yuliang Zou, Zelun Luo, and Jia-Bin Huang. DF-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *ECCV*, 2018. 2