Temporally Consistent Online Depth Estimation in Dynamic Scenes

Zhaoshuo Li[1], Wei Ye[2], Dilin Wang[2], Francis X. Creighton[1], Russell H. Taylor[1],
Ganesh Venkatesh[2], Mathias Unberath[1]

[1] Johns Hopkins University [2] Reality Labs, Meta Inc.

Abstract

Temporally consistent depth estimation is crucial for online applications such as augmented reality. While stereo depth estimation has received substantial attention as a promising way to generate 3D information, there is relatively little work focused on maintaining temporal stability. Indeed, based on our analysis, current techniques still suffer from poor temporal consistency. Stabilizing depth temporally in dynamic scenes is challenging due to concurrent object and camera motion. In an online setting, this process is further aggravated because only past frames are available. We present a framework named Consistent Online Dynamic Depth (CODD) to produce temporally consistent depth estimates in dynamic scenes in an online setting. CODD augments per-frame stereo networks with novel motion and fusion networks. The motion network accounts for dynamics by predicting a per-pixel SE3 transformation and aligning the observations. The fusion network improves temporal depth consistency by aggregating the current and past estimates. We conduct extensive experiments and demonstrate quantitatively and qualitatively that CODD outperforms competing methods in terms of temporal consistency and performs on par in terms of per-frame accuracy.

1. Introduction

For online applications such as augmented reality, estimating consistent depth across video sequences is important, as temporal noise in depth estimation may corrupt visual quality and interfere with downstream processing such as surface extraction. One way to acquire metric depth (*i.e.* without scale ambiguity) is to use calibrated stereo images. Recent developments in stereo depth estimation have been focusing on improving disparity accuracy on a per-frame basis [5, 2, 4, 24, 11]. However, none of these approaches considers temporal information nor attempts to maintain temporal consistency. We examine the temporal stability of different per-frame networks and find that current solutions suffer from poor temporal consistency. We quantify such temporal inconsistency in predictions in Sect. 4 and provide qualitative visualization of resulting artifacts in the video supplement to further illustrate the case.

We posit that stabilizing depth estimation temporally requires reasoning between current and previous frames, *i.e.* establishing cross-frame correspondences and correlating the predicted values. In the simplest case where the scene is entirely static and camera poses are known [13, 14], camera motion can be corrected by a single SE3 transformation. Considering geometric constraints of multiple camera views [1], the aligned cameras have the same viewpoint onto the static scene and therefore, depth values are expected to be the same, which allows pixel-wise aggregation of depth estimates from different time for consistency.

However, in a *dynamic* environment with moving and deforming objects, multi-view constraints do not hold. Even if cross-frame correspondences are established, independent depth estimates for corresponding points cannot simply be fused. This is because depth is not translation invariant and thus, fusion requires aligning depth predictions into a common coordinate frame. Therefore, prior works [16, 10] explicitly remove moving objects and only stabilize the static background to comply with the constraint. Given additional 3D motion, *e.g.* estimated by scene flow, depth of both moving and static objects can be aligned, which then enables temporal consistency processing [33]. However, as do the previously mentioned approaches, Zhang *et al.* [33] use information from all video frames and optimizes network parameters at application time, limiting itself to *offline* use.

In an *online* setting, prior works [23, 32, 3] incorporate temporal information by appending a recurrent network to the depth estimation modules. However, these recurrent networks do not provide explicit reasoning between frames. Moreover, these approaches [23, 32, 3] consider depth estimation from single images, not stereo pairs. Due to the scale ambiguity of monocular depth estimation, prior works mainly focus on producing estimates with consistent scale [10, 33] instead of reducing the inter-frame jitter that arises in metric depth estimation.

Traditionally, to encourage temporal consistency in *metric depth*, prior techniques have relied on hand-crafted probability weights [8, 18, 27]. One example is the Kalman filter

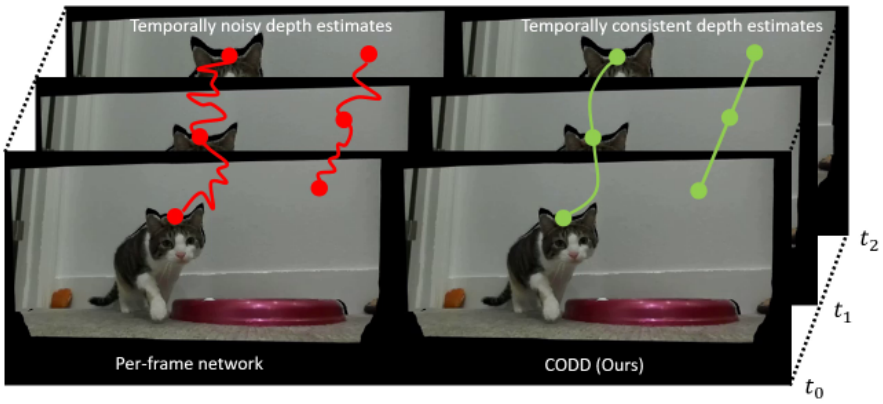

Figure 1: Temporal consistency comparison of per-frame stereo depth estimation solutions and our proposed CODD framework. The point cloud visualization is generated given the depth estimates. **This animated figure is best viewed in Adobe Reader (click button to play).** More visualizations can be found in the video supplement.

[8], which combines previous and current predictions based on associated uncertainties. However, it assumes Gaussian measurement error, which often fails in scenarios with occlusion and de-occlusion between frames.

We present a framework named Consistent Online Dynamic Depth (CODD) that produces temporally consistent depth predictions. To account for inter-frame motion, we integrate a motion network that predicts a per-pixel SE3 transformation that aligns previous estimates to the current frame. To remove temporal jitters and outliers from estimates, a fusion network is designed to aggregate depth predictions temporally. Compared with existing methods, CODD produces temporally consistent metric depth and is capable of handling dynamic scenes in an online setting. Qualitative results of improved temporal consistency of CODD framework is shown in Fig. 1.

For evaluation, we first show empirically that current stereo depth estimation solutions indeed suffer from poor temporal stability. We quantify such inconsistency with a set of temporal metrics and find that networks with *better* per-frame accuracy may have *worse* temporal consistency. We then benchmark CODD on varied datasets, including synthetic data of rigid [19] or deforming [29] objects, real-world footage of driving [28, 20], indoor and medical scenes. CODD improves over competing methods in terms of temporal metrics by up to 31%, and performs on par in terms of per-frame accuracy. The improvement is attributed to the temporal information that our model leverages. We conduct extensive ablation studies of different components of CODD and further demonstrate the performance upper bound of our proposed setup empirically, which may motivate future research. CODD can run at 25 FPS on modern hardware.

Our contribution can be summarized as following:

– We study an important yet under-studied problem for online applications such as augmented reality: *temporal consistency* in depth estimation from stereo images. We demonstrate that contemporary per-frame solutions suffer from temporal noise.

– We present a general framework CODD that builds on per-frame stereo depth networks for improved temporal stability. We design a novel *motion* network to accommodate dynamics and a novel *fusion* network to encourage temporally consistency in an online setting.

– We conduct experiments across varied dataset to demonstrate the favorable temporal performance of CODD without sacrificing per-frame accuracy.

2. Related Work

Stereo depth networks compute disparity between left and right images to obtain the depth estimate given camera parameters. Classical approaches, such as SGM [5], use mutual information to guide disparity estimation. In recent years, different network architectures have been proposed including 3D-convolution-based networks such as PSMNet [2], correlation-based networks such as HITNet [24], hybrid approaches with both 3D convolution and correlation such as GwcNet [4], and transformer-based networks such as STTR [11]. CODD builds on top of these per-frame methods to improve temporal consistency.

Temporally consistent depth networks aim to produce coherent depth for a video sequence. *Offline* approaches assume no temporal constraints. Some methods [13, 14] are only applicable in static scenes while others [16, 10] explicitly mask out moving objects and optimize for static background only. Zhang *et al*. [33] extend the aforementioned prior methods to dynamic scenes by using additional 3D scene flow estimation. *Online* approaches [23, 32] have used recurrent modules to aggregate temporal information.

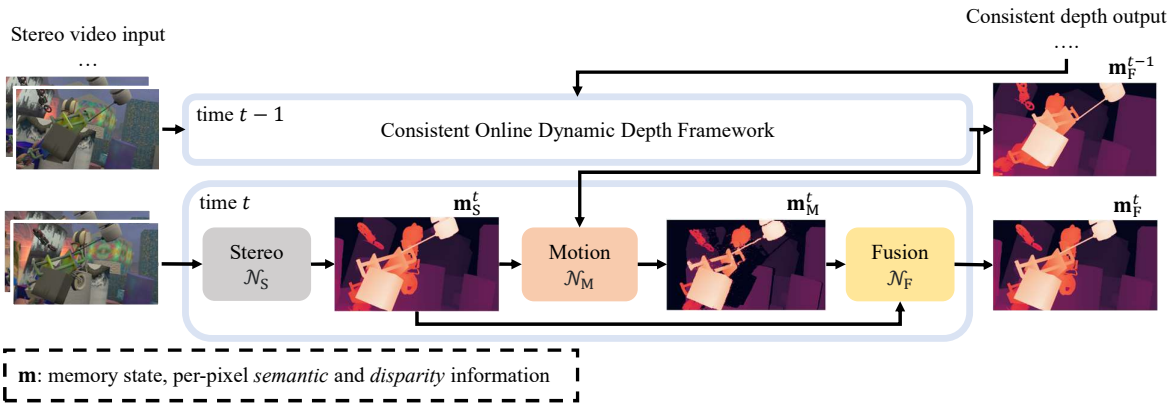

Figure 2: Consistent Online Dynamic Depth (CODD) framework has three sub-networks: *stereo* (S), *motion* (M) and *fusion* (F) networks. Each sub-network extracts or updates a memory state $\mathbf{m}$, containing semantic and disparity information. Given an input video sequence, the stereo network extracts an initial estimate $\mathbf{m}_\mathrm{S}^t$ on a per-frame basis. The motion network then aligns the preceding fusion memory state $\mathbf{m}_\mathrm{F}^{t-1}$ with the current frame, generating the motion memory state $\mathbf{m}_\mathrm{M}^t$. A fusion network lastly fuses the two memory states as $\mathbf{m}_\mathrm{F}^t$, containing the temporally consistent disparity $\mathbf{d}_\mathrm{F}^t$. Images from FlyingThings3D dataset [19].

However, these approaches were studied in the context of monocular depth, where errors are dominated by scale inconsistency [10, 33]. Moreover, the recurrent modules do not provide an explicit mechanism of how temporal information is used. CODD is instead designed for metric depth from stereo images and provides explicit reasoning between frames in an online setting.

Scene flow estimation seeks to recover inter-frame 3D motion from a set of stereo or RGBD images [17, 30, 26]. While previous algorithms aim at generating accurate scene flow between frames, our work uses the 3D motion as an intermediary such that previous estimates can be aligned with the current frame. Following RAFT3D [26], we predict the inter-frame 3D motion as a per-pixel SE3 transformation.

Simultaneous localization and mapping (SLAM) jointly estimates camera poses and a mapping of the scene. Many approaches [21, 7, 31, 15], such as DynamicFusion, accumulate information over all past frames. These approaches may include objects that have already exited the scene and are thus not relevant anymore, leading to excess compute. Another common assumption is that only a single or few objects are in the scene, which restricts applicability. CODD can be seen as a "temporally local" SLAM system where only the immediately preceding frames are considered and no prior information of the scene is assumed.

3. Consistent Online Dynamic Depth

The goal of CODD is to estimate temporally consistent depth in dynamic scenes in an online setting. A stereo video stream is taken as input, where each image is of dimension $\mathbb{R}^{I_H \times I_W \times 3}$. Let a memory state $\mathbf{m} \in \mathbb{R}^{I_H \times I_W \times (3+C+1)}$ be a combination of per-pixel $3+C$-channel semantic and 1-channel disparity information. CODD, as shown in Fig. 2, consists of three sub-networks:

- A *stereo* network $\mathcal{N}_\mathrm{S}$ (Sect. 3.1) estimates the initial disparity and semantic feature map on a per-frame basis. The semantic information is encoded as feature maps and RGB values.
- A *motion* network $\mathcal{N}_\mathrm{M}$ (Sect. 3.2) accounts for motion across frames by aligning the previous predictions to the current frame. Such motion information is also added to the semantic information for better fusion.
- A *fusion* network $\mathcal{N}_\mathrm{F}$ (Sect. 3.3) that fuses the disparity estimates across time to promote the temporal consistency in an online setting.

We introduce the high-level concepts of each sub-network in the following sections and detail the designs in Appx A.

3.1. Stereo Network

The objective of the stereo network is to extract an initial estimate of disparity and semantic feature map on a per-frame basis. The stereo network $\mathcal{N}_s$ takes the current stereo images as input and outputs the stereo memory state $\mathbf{m}_\mathrm{S}^t$ at time t. Internally, it extracts $\mathbb{R}^{I_H \times I_W \times C}$ semantic feature maps from the stereo images, and computes $\mathbb{R}^{I_H \times I_W \times 1}$ disparity by finding matches between the feature maps. In this paper, we use a recent network, HITNet [24], as our building block to estimate per-frame disparity due to its real-time speed and superior performance. In the subsequent sections, we discuss how to extend the stereo network to stabilize the disparity prediction temporally.

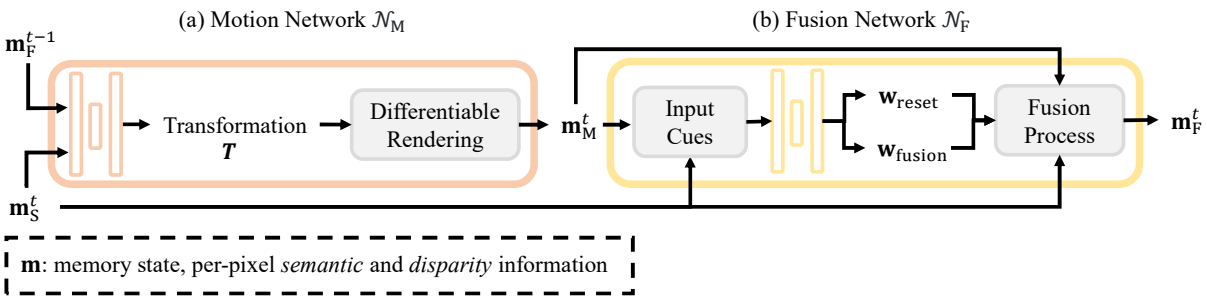

Figure 3: (a) The motion network computes a per-pixel SE3 transformation $\boldsymbol{T}$. The transformation $\boldsymbol{T}$ is used to align $\mathbf{m}_F^{t-1}$ to current stereo state $\mathbf{m}_S^t$ by differentiable rendering, generating the motion memory state $\mathbf{m}_M^t$. (b) The fusion network first extracts a set of input cues from the memory states $\mathbf{m}_M^t$ and $\mathbf{m}_S^t$. It then regresses the reset and fusion weights, where $\mathbf{w}_{\text{reset}}$ removes outliers and $\mathbf{w}_{\text{fusion}}$ aggregates the predictions. The fusion network outputs $\mathbf{m}_F^t$, containing the temporally consistent disparity estimate $\mathbf{d}_F^t$.

3.2. Motion Network

The motion network aligns the previous memory state with that of the current frame. Let $\mathbf{m}_F^{t-1}$ denote the preceding memory state from our online consistent depth network at time $t-1$. Our motion network $\mathcal{N}_M$, as shown in Fig. 3a, transforms $\mathbf{m}_F^{t-1}$ into current frame based on $\mathbf{m}_S^t$:

$$\mathbf{m}_M^t \leftarrow \mathcal{N}_M\big(\mathbf{m}_F^{t-1},\ \mathbf{m}_S^t\big),$$

where $\mathbf{m}_M^t$ is the state from $t-1$ aligned to t.

To perform such alignment, the inter-frame motion must be recovered. In a dynamic scene with camera movement and object movement/deformation, the motion prediction needs to be on a per-pixel level. Our motion network builds on top of RAFT3D [26] to predict a per-pixel SE3 transformation map $\boldsymbol{T} \in \mathbb{SE}(3)^{I_H \times I_W}$. The motion is predicted using a GRU network and a Gauss-Newton optimization mechanism based on matching confidence for $\mathcal{K}$ iterations. Once the motion between frames is recovered, we project the previous memory state $\mathbf{m}_F^{t-1}$ to the current frame similar to [12] using differentiable rendering [22]:

$$\mathbf{m}_M^t \leftarrow \pi\Big(\boldsymbol{T}\pi^{-1}\big(\mathbf{m}_F^{t-1}\big)\Big), \tag{1}$$

where π and π^{-1} are perspective and inverse perspective projection, respectively. Thus, the motion memory state $\mathbf{m}_M^t$ resides in the current camera coordinate frame and has pixel-wise correspondence with the current prediction, which enables temporal aggregation of disparity. Additionally, we estimate a binary visibility mask by identifying the regions in the current frame that is not visible in previous one. We also compute the confidence of motion via Sigmoid and compute the motion magnitude as the L2 norm of the scene flow. These information are added to the memory state $\mathbf{m}_M^t$ for the fusion network to adaptively fuse the predictions. We detail differences between our motion network and RAFT3D and provide quantitative comparison in Appx A.2.

3.3. Fusion Network

The objective of the fusion network (Fig. 3b) is to promote temporal consistency by aggregating the disparities of the motion and stereo memory states. The output of the fusion network is the fusion memory state $\mathbf{m}_F^t$:

$$\mathbf{m}_F^t \leftarrow \mathcal{N}_F\big(\mathbf{m}_M^t,\ \mathbf{m}_S^t\big), \tag{2}$$

where $\mathcal{N}_F$ is the fusion network and $\mathbf{m}_F^t$ contains the temporally consistent disparity estimate $\mathbf{d}_F^t$. We first discuss the fusion process (Sect. 3.3.1) and then cover the set of cues extracted from the memory states (Sect. 3.3.2) to guide such fusion process.

3.3.1 Fusion Process

The temporally consistent disparity $\mathbf{d}_F^t$ is obtained by fusing the aligned and current disparity estimates. Let $\mathbf{d}_M^t$ be the disparity from the motion network and $\mathbf{d}_S^t$ be the disparity from the stereo network. The fusion network computes a reset weight $\mathbf{w}_{\text{reset}}$ and a fusion weight $\mathbf{w}_{\text{fusion}}$ both of dimension $\mathbb{R}^{I_H \times I_W}$. The fusion process of disparity estimates is formulated as:

$$\mathbf{d}_F^t = \big(1 - \mathbf{w}_{\text{reset}}\,\mathbf{w}_{\text{fusion}}\big)\,\mathbf{d}_S^t + \mathbf{w}_{\text{reset}}\,\mathbf{w}_{\text{fusion}}\,\mathbf{d}_M^t\,. \tag{3}$$

The fusion memory state $\mathbf{m}_F^t$ is thus formed by the fused disparity $\mathbf{d}_F^t$ and semantic features extracted by the stereo network.

The intuition behind the fusion process is two-fold. First, it filters out outliers using $\mathbf{w}_{\text{reset}}$. The outliers can be either induced by inaccurate disparity or motion predictions. In

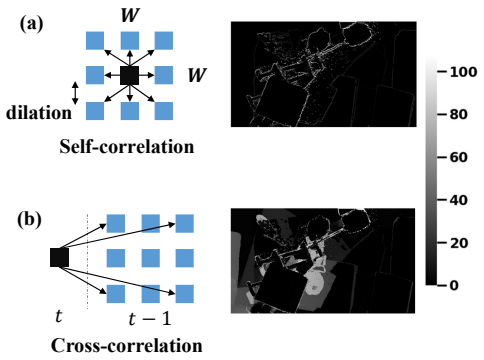

Figure 4: The pixel-to-patch correlation mechanisms and example visualizations of a single channel of disparity correlations. Black pixels: source pixels. Blue pixels: correlated pixels. We note that, qualitatively, self-correlation identifies local discontinuities and cross-correlation identifies inter-frame disagreements. Images from FlyingThings3D dataset [19].

our work, $\mathbf{w}_{\text{reset}}$ is supervised to identify outliers whose errors are larger than the other disparity estimate by a threshold of τ_{reset}. Second, the fusion process encourages temporal consistency by fusing current disparity prediction with reliable predictions propagated from previous frame using $\mathbf{w}_{\text{fusion}}$. When disparity estimates are considered to be equally reliable within a threshold τ_{fusion}, the fusion network aggregates them with a regressed value between 0 and 1. As reset weights should already reject the most significant outliers, we set $\tau_{\text{fusion}} < \tau_{\text{reset}}$.

3.3.2 Input Cues

To determine the reset and fusion weights, we collect a set of input cues from $\mathbf{m}_{\text{M}}^t$ and $\mathbf{m}_{\text{S}}^t$. We find explicit input cues are advantageous over channel-wise concatenation of the two memory states.

First, the ***disparity confidence*** of $\mathbf{d}_{\text{S}}^t$ and $\mathbf{d}_{\text{M}}^t$ are computed. As disparities are computed mainly based on appearance similarity between the left and right images, we approximate the confidence of disparity prediction by computing the ℓ_1 distance between the left and right features extracted from stereo network. For robustness against matching ambiguity, we additionally offset each disparity estimate by -1 and 1 to collect local confidence information, forming a 3 channel confidence feature.

However, in the case of stereo occlusion, *i.e.* regions that are only visible in the left but not the right image, the disparity confidence based on appearance similarity becomes ill-posed. Thus, we additionally use the local smoothness information as a cue. We implement a pixel-to-patch ***self-correlation*** (Fig. 4a) to approximate the local smoothness information, which computes the correlation between a pixel and its neighboring pixels in a local patch of size $W \times W$, forming a $\mathbb{R}^{I_H \times I_W \times (W^2-1)}$ correlation feature. A dilation may be used to increase the receptive field. We apply the pixel-to-patch self-correlation for both disparity and semantic features to acquire local disparity and appearance smoothness.

In the case of inaccurate motion predictions, the aligned memory state may contain wrong cross-frame correspondences. Therefore, the predictions in these regions need to be discarded as outliers. To promote such outlier rejection, the fusion network applies a pixel-to-patch ***cross-correlation*** (Fig. 4b) to evaluate the cross-frame disparity and appearance similarities. In cross-correlation, each pixel attends to a local $W \times W$ patch of the previous image centered at the same pixel location after motion correction, forming a $\mathbb{R}^{I_H \times I_W \times W^2}$ correlation feature. In our implementation, we use the ℓ_1 distance for disparity and dot-products for appearance correlation.

Lastly, we observe that when the inter-frame motion is large, the motion estimate is less reliable and may thus result in wrong cross-frame correspondences. Therefore, we provide the flow magnitude and confidence as a motion cue. We additionally use the visibility mask from the projection process to identify the invalid regions and provide the semantic features for context information.

3.4. Supervision

Stereo and Motion Network We supervise the stereo network on the per-frame disparity estimate $\mathbf{d}_{\text{S}}^t$ against the ground truth following [24]. We supervise the motion network on the transformation prediction $\boldsymbol{T}$, with a loss imposed on its projected scene flow against the ground truth following [26].

Fusion Network We supervise the fusion network to promote temporal consistency of disparity estimates. We note that optimizing for disparity changes only may not be ideal, as errors in the previous frame will propagate to the current frame even if the predicted disparity change is correct. Therefore, we impose losses on the disparity prediction $\mathbf{d}_{\text{F}}^t$, and predicted weights $\mathbf{w}_{\text{reset}}, \mathbf{w}_{\text{fusion}}$.

We supervise the predicted disparity $\mathbf{d}_{\text{F}}^t$ against the ground truth using Huber loss [6], denoted as the disparity loss ℓ_{disp}.

Further, we supervise the reset weights $\mathbf{w}_{\text{reset}}$ such that they reject the worse prediction between the stereo and motion network disparity estimates. Let $e_{\text{M}} = |\mathbf{d}_{\text{M}}^t - \mathbf{d}_{\text{gt}}^t|$ be the error of the motion disparity and $e_{\text{S}} = |\mathbf{d}_{\text{S}}^t - \mathbf{d}_{\text{gt}}^t|$ be the error of the stereo disparity against the ground truth $\mathbf{d}_{\text{gt}}^t$. Because $\mathbf{w}_{\text{reset}}$ rejects the motion disparity estimate $\mathbf{d}_{\text{M}}^t$ when its value is zero (Eqn. 3), we impose a loss such that it favors zero when e_{M} is worse and favors one when e_{M} is better.

Thus, we have

$$\ell_{\text{reset}} = \begin{cases} \mathbf{w}_{\text{reset}}, & 1) \text{ if } e_{\text{M}} > e_{\text{S}} + \tau_{\text{reset}}, \\ 1 - \mathbf{w}_{\text{reset}}, & 2) \text{ if } e_{\text{M}} < e_{\text{S}} - \tau_{\text{reset}}, \\ 0, & 3) \text{ otherwise}, \end{cases}$$

where e_{M} is worse in condition 1) while e_{M} is better in condition 2). Otherwise, the loss is zero as shown in condition 3).

The fusion weights $\mathbf{w}_{\text{fusion}}$ are supervised such that they aggregate the past two disparity estimates correctly:

$$\ell_{\text{fusion}} = \begin{cases} \mathbf{w}_{\text{fusion}}, & 1) \text{ if } e_{\text{M}} > e_{\text{S}} + \tau_{\text{fusion}}, \\ 1 - \mathbf{w}_{\text{fusion}}, & 2) \text{ if } e_{\text{M}} < e_{\text{S}} - \tau_{\text{fusion}}, \\ \alpha_{\text{reg}} \cdot |\mathbf{w}_{\text{fusion}} - 0.5|, & 3) \text{ otherwise.} \end{cases}$$

Different from the reset weights, the fusion weights are not only trained to identify the better estimate as shown in condition 1) and 2), but also trained with an additional regularization term such that the fusion weights are around 0.5 when both estimates are considered equally good as shown in condition 3).

The final loss for fusion network training is computed as:

$$\ell_{\text{F}} = \alpha_{\text{disp}}\ell_{\text{disp}} + \alpha_{\text{fusion}}\ell_{\text{fusion}} + \alpha_{\text{reset}}\ell_{\text{reset}}\,, \tag{4}$$

which is the weighted sum of $\ell_{\text{disp}}, \ell_{\text{fusion}}$ and ℓ_{reset}.

4. Experimental Setup

4.1. Implementation Details

We use a batch of 8 and Adam [9] as the optimizer on Nvidia V100 GPUs. Following [26], we use a linearly decayed learning rate of $2\text{e}{-4}$ for pre-training and $2\text{e}{-5}$ for fine-tuning. We pre-train motion and fusion networks for 25000 and 12500 steps, and halve the steps during fine-tuning. We perform $\mathcal{K} = 1$ steps of incremental updates in the motion network when datasets contain small motion (*e.g.* TartanAir [29]) and $\mathcal{K} = 16$ otherwise (*e.g.* FlyingThings3D [19], KITTI Depth [28] and KITTI 2015 [20]). In the fusion network, we use a patch size $W = 3$ with a dilation of 2 for pixel-to-patch correlation to increase the receptive field. By default, we set $\tau_{\text{reset}} = 5$, $\tau_{\text{fusion}} = 1$, $\alpha_{\text{reg}} = 0.2$, and $\alpha_{\text{disp}} = \alpha_{\text{fusion}} = \alpha_{\text{reset}} = 1$. Due to the sparsity of ground truth in KITTI datasets, supervising the fusion weights can be ill-posed as only few pixels are aligned across time. We set $\alpha_{\text{fusion}} = \alpha_{\text{reset}} = 0$.

4.2. Metrics

We propose to quantify the temporal inconsistency by a temporal end-point-error (TEPE) metric and the relative error (TEPE_{r}) given cross-frame correspondences:

$$\text{TEPE} = |\Delta d - \Delta d_{\text{gt}}|\,,\ \text{TEPE}_{\text{r}} = \frac{\text{TEPE}}{|\Delta d_{\text{gt}}| + \epsilon}\,, \tag{5}$$

where $\Delta d, \Delta d_{\text{gt}}$ are *signed* disparity change and $\epsilon = 1\text{e}{-3}$ avoids division by zero. Intuitively, TEPE and TEPE_{r} reflects the absolute and relative error between predicted and ground truth depth motion between two time points. TEPE is generally proportional to the ground truth magnitude and thus better reflects consistencies in pixels with large motion. TEPE_{r} better captures the consistencies of static pixels due to the $1/\epsilon$ weight. We also report threshold metrics of 3px for TEPE and 100% for TEPE_{r} ($\delta^{\text{t}}_{3\text{px}}$, $\delta^{\text{t}}_{100\%}$). Temporal metrics themselves can be limited as a network can be temporally consistent but wrong. Therefore, we also report the per-pixel disparity error using EPE and threshold metric of 3px ($\delta_{3\text{px}}$). We exclude pixels with extreme scene flow ($>$ 210px) or disparity ($<$1px or $>$210px) following [26] as they are generally outliers in our intended application. For all metrics, lower is better.

5. Results and Discussion

We first show quantitatively that current stereo depth estimation solutions suffer from poor temporal consistency (Sect. 5.1). We then show that CODD improves upon per-frame stereo networks and outperforms competing approaches across varied datasets, *sharing the same stereo network without the need of re-training or fine-tuning* (Sect. 5.2–Sect. 5.3). We lastly present ablation studies (Sect. 5.4) and inference time (Sect. 5.5) to characterize CODD.

5.1. Temporal Consistency Evaluation

We examine the temporal consistency of stereo depth techniques of different designs that operate on a per-frame basis. We use FlyingThings3D finalpass [19] dataset as it is commonly used for training stereo networks.

Results As shown in Tab. 1, all considered approaches have large TEPE and TEPE_{r}, with TEPE often larger than EPE and more than 14 times the ground truth disparity change as implied by TEPE_{r}. This suggests that the considered approaches suffer from poor temporal stability de-

Table 1: Temporal and per-pixel metrics of contemporary approaches evaluated on the FlyingThings3D dataset [19] with the official checkpoints when provided. †: Classical approach. ‡: Non-occluded regions only. We use HITNet [24] as stereo network due to its real-time speed and superior performance.

	TEPE	$\delta^{\text{t}}_{3\text{px}}$	TEPE_{r}	$\delta^{\text{t}}_{100\%}$	EPE	$\delta_{3\text{px}}$
SGM [5] †	2.355	0.065	78.482	0.591	2.965	0.090
STTR [11] ‡	0.482	0.014	11.434	0.374	0.449	0.014
PSMNet [2]	1.371	0.056	35.136	0.466	1.079	0.045
GwcNet	0.959	0.041	22.598	0.409	0.752	0.032
HITNet [24]	0.812	0.040	16.840	0.291	0.607	0.030

Table 2: Results on the FlyingThings3D dataset [19].

	TEPE	δ^{t}_{3px}	$TEPE_r$	$\delta^{t}_{100\%}$	EPE	δ_{3px}
Stereo [24]	0.812	0.040	16.840	0.291	0.607	0.030
Motion	0.875	0.036	24.533	0.390	0.777	0.030
Kalman filter [8]	0.793	0.040	15.843	0.230	0.610	0.030
CODD (Ours)	**0.741**	**0.034**	**15.205**	**0.214**	**0.595**	**0.028**

spite good per-pixel accuracy. A qualitative visualization is shown in Fig. 1. We show the resulting artifacts in video supplement to further illustrate the case.

5.2. Pre-training on FlyingThings3D

We first demonstrate that CODD improves the temporal stability of per-frame networks by using the pre-trained HITNet [24] as our stereo network and freezing its parameters during training for fair comparison. We follow the official split of FlyingThings3D. Tab. 2 summarizes the result.

Results To ensure temporal consistency, one naive way is to only forward the past information to current frame. Thus, we compare the results of the per-frame prediction $\mathbf{d}^t_S$ of the stereo network with the the aligned preceding prediction $\mathbf{d}^t_M$ of the motion network. To evaluate fairly, we fill occluded regions that are not observable in the past with $\mathbf{d}^t_S$. As shown in Tab. 2, the motion estimate $\mathbf{d}^t_M$ is worse than stereo estimate $\mathbf{d}^t_S$ due to outliers caused by wrong motion predictions as shown blue in Fig. 5, suggesting that only forwarding past information is not feasible. Nonetheless, $\mathbf{d}^t_M$ performs on par with $\mathbf{d}^t_S$ most of the cases (white) and can even mitigate errors of $\mathbf{d}^t_S$ that is hard to predict in current frame (red). Thus, techniques to adaptively handle these cases can greatly reduce temporal noise.

While Kalman filter [8] successfully improves the temporal consistency by combining the two outputs, it leads to worse EPE. The result indicates that temporal consistency is indeed a *different* problem from per-frame accuracy. In contrast, CODD performs better in all metrics, which suggests that CODD achieves better stability across frames by pushing predictions towards the ground truth instead of propagating errors in time. We further show that our pipeline is applicable to other stereo networks in Appx C.1.

5.3. Benchmark Results

We benchmark CODD across varied datasets. We first fine-tune the stereo network on each dataset to ensure good per-frame accuracy and keep it frozen for fair comparison. We then fine-tune the motion and fusion networks, using the output of stereo network as input. We summarize results in Tab. 3, provide end-to-end results in Appx C.2 and zero-shot results in Appx C.3.

Dataset TartanAir [29] is a synthetic dataset with simulated drone motions in different scenes. We use 15 scenes (219434 images) for training, 1 scene for validation (6607 images) and 1 scene (5915 images) for testing.

KITTI Depth [28] has footage of real-world driving scenes, where ground truth depth is acquired from LiDAR. We follow the official split and train CODD on 57 scenes (38331 images), validate on 1 scene (1100 images), and test on 13 scenes (3426 images). We use pseudo ground truth information inferred from an off-the-shelf optical flow network [25] trained on KITTI 2015.

KITTI 2015 [20] is a subset of KITTI Depth with 200 temporal image pairs. The data is complementary to KITTI Depth as ground truth optical flow information is provided, however the long-term temporal consistency is not captured as there are only two frames for each scene. We train on 160 image pairs, validate on 20 image pairs and test on 20 image pairs. Given the small dataset size, we perform five-fold cross-validation experiments and report the average results.

Results We find that CODD consistently outperforms the per-frame stereo network [24] across all metrics similar to findings in Sect. 5.2. The $TEPE_r$ is improved by up to 31%, from 9.039 to 6.206 in the TartanAir dataset. Compared to the Kalman filter, CODD leads to smaller EPE in contrast to Kalman filter, indicating both improved temporal and per-pixel performance. This again demonstrates the improved temporal performance does *NOT* guarantee improved per-pixel accuracy. We note that all settings in KITTI 2015 have the same EPE and δ_{3px}, because we do not have ground truth information to evaluate the per-pixel accuracy of the fusion result of second frame. More qualitative visualizations can be found in video supplement and Appx B.

5.4. Ablation Experiments

We conduct experiments on the FlyingThings3D dataset [19] to ablate the effectiveness of different sub-components.

5.4.1 Fusion Network

We summarize the key ablation experiments of fusion network in Tab. 4 and provide additional results in Appx C.4.

a) Reset weights: In theory, only the fusion weight $\mathbf{w}_{\text{fusion}}$ is needed between two estimates as it can reject outliers by estimating extreme values (*e.g.*, 0 or 1). However, we find that predicting additional reset weights $\mathbf{w}_{\text{reset}}$ improves performance across all metrics. This may be attributed to the difference in supervision, where reset weights $\mathbf{w}_{\text{reset}}$ are trained for outlier detection, while fusion weights $\mathbf{w}_{\text{fusion}}$ are trained for aggregation.

b) Fusion input cues: Other than the disparity confidence, self- and cross- correlation, we incrementally add the flow confidence/magnitude (+FL), visibility mask (+V) and semantic feature map (+SM) to the fusion networks. We find that the metrics improve marginally with additional

Figure 5: Temporal consistency comparison between stereo $\mathbf{d}_{\mathrm{S}}^{t}$ and motion $\mathbf{d}_{\mathrm{M}}^{t}$ predictions. While $\mathbf{d}_{\mathrm{M}}^{t}$ contains outliers (blue), it can also mitigate errors of $\mathbf{d}_{\mathrm{S}}^{t}$ (red). Our fusion networks learns to adaptively fuse the estimates. Images from FlyingThings3D dataset [19].

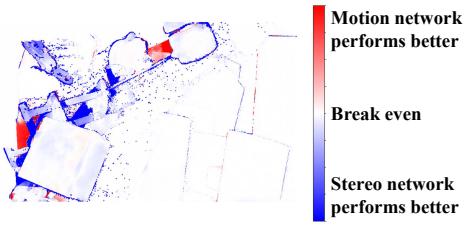

Table 3: Results on TartanAir [29], KITTI Depth [28], KITTI 2015 [20].

		TEPE	$\delta^{\mathrm{t}}_{3\mathrm{px}}$	$\mathrm{TEPE}_{\mathrm{r}}$	$\delta^{\mathrm{t}}_{100\%}$	EPE	$\delta_{3\mathrm{px}}$
TartanAir dataset [29]	Stereo [24]	0.876	0.053	9.039	0.339	0.934	0.055
	Kalman Filter [8]	0.829	0.053	7.501	0.252	0.935	0.055
	CODD (Ours)	**0.751**	**0.045**	**6.206**	**0.240**	**0.904**	**0.053**
KITTI Depth dataset [28]	Stereo [24]	0.289	**0.001**	3.630	0.156	0.423	0.004
	Kalman Filter [8]	0.278	**0.001**	**2.615**	**0.125**	0.431	0.005
	CODD (Ours)	**0.258**	**0.001**	2.764	0.132	**0.418**	**0.003**
KITTI 2015 dataset [20]	Stereo [24]	0.570	0.026	10.672	0.126		
	Kalman Filter [8]	0.533	0.026	9.668	0.117	0.811	0.033
	CODD (Ours)	**0.507**	**0.022**	**8.740**	**0.112**		

Table 4: Ablation studies of fusion network. Underline: base setting.

		TEPE	$\delta^{\mathrm{t}}_{3\mathrm{px}}$	$\mathrm{TEPE}_{\mathrm{r}}$	$\delta^{\mathrm{t}}_{100\%}$	EPE	$\delta_{3\mathrm{px}}$
a) Reset weight	×	0.783	0.036	15.953	0.217	0.618	0.030
	✓	**0.756**	**0.035**	**15.013**	**0.211**	**0.604**	**0.029**
b) Fusion input cues	+FL	0.763	0.035	15.103	0.211	**0.604**	0.029
	+V	0.758	0.035	15.082	**0.210**	0.605	0.029
	+SM	**0.756**	0.035	**15.013**	0.211	**0.604**	0.029
c) Training sequence length	2	0.756	0.035	15.013	**0.211**	0.604	0.029
	3	0.753	0.035	**14.942**	**0.211**	0.600	0.029
	4	**0.741**	**0.034**	15.205	0.214	**0.595**	**0.028**

inputs, especially for TEPE.

c) Training sequence length: During training, by default we train with sequences with length of two frames, where the second frame takes the stereo outputs of first frame as input. However, during inference process, the preceding outputs are from the fusion network. Thus, to better approximate the inference process, we further extend sequence length to three and four. We find that fusion network consistently benefits from the increasing training sequence length. However, this elongates training time proportionally.

5.4.2 Empirical Best Case

We provide an empirical study of the "best case" of motion or fusion network in Tab. 5. For empirical best motion network $\mathcal{N}_{\mathrm{M}}$, we use ground truth scene flow and set flow confidence to one. For empirical best fusion network $\mathcal{N}_{\mathrm{F}}$, we pick the better disparity estimates pixel-wise between the stereo $\mathbf{d}_{\mathrm{S}}^{t}$ and motion $\mathbf{d}_{\mathrm{M}}^{t}$ estimates given ground truth disparity. We find that perfect motion leads to substantial reduction in $\mathrm{TEPE}_{\mathrm{r}}$ while perfect fusion leads to substantial reduction in TEPE. In both cases, EPE also improves. While CODD performs well against competing approaches, advancing the motion or fusion networks has the potential to substantially improve temporal consistency.

Table 5: Ablation study of empirical best case by using ground truth information.

	TEPE	$\delta^{\mathrm{t}}_{3\mathrm{px}}$	$\mathrm{TEPE}_{\mathrm{r}}$	$\delta^{\mathrm{t}}_{100\%}$	EPE	$\delta_{3\mathrm{px}}$
CODD (Ours)	0.741	0.034	15.205	0.214	0.595	0.028
Empirical best $\mathcal{N}_{\mathrm{M}}$	0.879	0.043	**5.401**	**0.125**	0.571	0.027
Empirical best $\mathcal{N}_{\mathrm{F}}$	**0.529**	**0.025**	9.936	0.214	**0.455**	**0.021**

5.5. Inference Speed and Number of Parameters

The inference speed of CODD on images of resolution 640×480 with $\mathcal{K} = 1$ is 25 FPS (stereo 26ms, motion 13ms, fusion 0.3ms) on an Nvidia Titan RTX GPU. The total number of parameters is 9.3M (stereo 0.6M, motion 8.5M and fusion 0.2M). Compared to the stereo network, the overhead is mainly introduced by the motion network.

5.6. Limitations

While CODD outperforms competing methods across datasets, we recognize that there is still a gap between CODD and the empirical best cases in Sect. 5.4.2. Furthermore, CODD cannot correct errors when both current and previous frame estimates are wrong, as the weights between estimates in the fusion process are bounded to $(0, 1)$. Lastly, we design CODD to look at the immediate preceding frame only. Exploiting more past information may potentially lead to performance improvement.

6. Conclusions

We present a general framework to produce temporally consistent depth estimation for dynamic scenes in an online setting. CODD builds on contemporary per-frame stereo depth approaches and shows superior performance across different benchmarks, both in terms of temporal metrics and per-pixel accuracy. Future work may extend upon our motion or fusion network for better performance and extend to multiple past frames.

References

[1] Alex M Andrew. Multiple view geometry in computer vision. *Kybernetes*, 2001. 1

[2] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018. 1, 2, 6

[3] Chanho Eom, Hyunjong Park, and Bumsub Ham. Temporally consistent depth prediction with flow-guided memory units. *IEEE Transactions on Intelligent Transportation Systems*, 21(11):4626–4636, 2019. 1

[4] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3273–3282, 2019. 1, 2

[5] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2007. 1, 2, 6

[6] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992. 5

[7] Matthias Innmann, Michael Zollhöfer, Matthias Nießner, Christian Theobalt, and Marc Stamminger. Volumedeform: Real-time volumetric non-rigid reconstruction. *ArXiv*, abs/1603.08161, 2016. 3

[8] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960. 1, 2, 7, 8

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[10] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1621, 2021. 1, 2, 3

[11] Zhaoshuo Li, Xingtong Liu, Nathan Drenkow, Andy Ding, Francis X Creighton, Russell H Taylor, and Mathias Unberath. Revisiting stereo depth estimation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6197–6206, 2021. 1, 2, 6

[12] Zhaoshuo Li, Amirreza Shaban, Jean-Gabriel Simard, Dinesh Rabindran, Simon DiMaio, and Omid Mohareri. A robotic 3d perception system for operating room environment awareness. *arXiv preprint arXiv:2003.09487*, 2020. 4

[13] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10986–10995, 2019. 1, 2

[14] Xiaoxiao Long, Lingjie Liu, Wei Li, Christian Theobalt, and Wenping Wang. Multi-view depth estimation using epipolar spatio-temporal networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8258–8267, 2021. 1, 2

[15] Yonghao Long, Zhaoshuo Li, Chi Hang Yee, Chi Fai Ng, Russell H Taylor, Mathias Unberath, and Qi Dou. E-dssr: Efficient dynamic surgical scene reconstruction with transformer-based stereoscopic depth perception. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 415–425. Springer, 2021. 3

[16] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Transactions on Graphics (TOG)*, 39(4):71–1, 2020. 1, 2

[17] Wei-Chiu Ma, Shenlong Wang, Rui Hu, Yuwen Xiong, and Raquel Urtasun. Deep rigid instance scene flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3614–3622, 2019. 3

[18] Sergey Matyunin, Dmitriy Vatolin, Yury Berdnikov, and Maxim Smirnov. Temporal filtering for depth maps generated by kinect depth camera. In *2011 3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)*, pages 1–4. IEEE, 2011. 1

[19] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 2, 3, 5, 6, 7, 8

[20] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences*, 2:427, 2015. 2, 6, 7, 8

[21] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015. 3

[22] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 4

[23] Denis Tananaev, Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. Temporally consistent depth estimation in videos with recurrent architectures. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 1, 2

[24] Vladimir Tankovich, Christian Hane, Yinda Zhang, Adarsh Kowdle, Sean Fanello, and Sofien Bouaziz. Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14362–14372, 2021. 1, 2, 3, 5, 6, 7, 8

[25] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 7

[26] Zachary Teed and Jia Deng. Raft-3d: Scene flow using rigid-motion embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8375–8384, 2021. 3, 4, 5, 6

[27] Iraklis Tsekourakis and Philippos Mordohai. Measuring the effects of temporal coherence in depth estimation for dy-

namic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 1

[28] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *2017 international conference on 3D Vision (3DV)*, pages 11–20. IEEE, 2017. 2, 6, 7, 8

[29] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4909–4916. IEEE, 2020. 2, 6, 7, 8

[30] Gengshan Yang and Deva Ramanan. Upgrading optical flow to 3d scene flow through optical expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1334–1343, 2020. 3

[31] Tao Yu, Kaiwen Guo, Feng Xu, Yuan Dong, Zhaoqi Su, Jianhui Zhao, Jianguo Li, Qionghai Dai, and Yebin Liu. Bodyfusion: Real-time capture of human motion and surface geometry using a single depth camera. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 910–919, 2017. 3

[32] Haokui Zhang, Chunhua Shen, Ying Li, Yuanzhouhan Cao, Yu Liu, and Youliang Yan. Exploiting temporal consistency for real-time video depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1725–1734, 2019. 1, 2

[33] Zhoutong Zhang, Forrester Cole, Richard Tucker, William T Freeman, and Tali Dekel. Consistent depth of moving objects in video. *ACM Transactions on Graphics (TOG)*, 40(4):1–12, 2021. 1, 2, 3