Learning Rigidity in Dynamic Scenes with a Moving Camera for 3D Motion Field Estimation

Zhaoyang Lv[1][*], Kihwan Kim[2], Alejandro Troccoli[2], Deqing Sun[2], James M. Rehg[1], Jan Kautz[2]

[1] Georgia Institute of Technology, Atlanta, U.S.
{zhaoyang.lv,rehg}@gatech.edu
[2] NVIDIA, Santa Clara, U.S.
{kihwank,atroccoli,deqings,jkautz}@nvidia.com

Abstract. Estimation of 3D motion in a dynamic scene from a temporal pair of images is a core task in many scene understanding problems. In real-world applications, a dynamic scene is commonly captured by a moving camera (i.e., panning, tilting or hand-held), increasing the task complexity because the scene is observed from different viewpoints. The primary challenge is the disambiguation of the camera motion from scene motion, which becomes more difficult as the amount of rigidity observed decreases, even with successful estimation of 2D image correspondences. Compared to other state-of-the-art 3D scene flow estimation methods, in this paper, we propose to *learn* the rigidity of a scene in a supervised manner from an extensive collection of dynamic scene data, and directly infer a rigidity mask from two sequential images with depths. With the learned network, we show how we can effectively estimate camera motion and projected scene flow using computed 2D optical flow and the inferred rigidity mask. For training and testing the rigidity network, we also provide a new semi-synthetic dynamic scene dataset (synthetic foreground objects with a real background) and an evaluation split that accounts for the percentage of observed non-rigid pixels. Through our evaluation, we show the proposed framework outperforms current state-of-the-art scene flow estimation methods in challenging dynamic scenes.

Keywords: Rigidity Estimation · Dynamic Scene Analysis · Scene Flow · Motion Segmentation

1 Introduction

The estimation of 3D motion from images is a fundamental computer vision problem, and key to many applications such as robot manipulation [3], dynamic scene reconstruction [14,23], autonomous driving [8,27,29,44], action recognition [43], and video analysis [13]. This task is commonly referred as *3D motion field* or *scene flow estimation*. 3D motion field estimation in a dynamic environment is, however, a challenging and still open problem when the scene is

---

[*] This work started during an internship that the author did at NVIDIA.

(a) Two RGB-D frames   (b) Rigidity (red for dynamic scene)   (c) Ego motion flow   (d) Projected scene flow
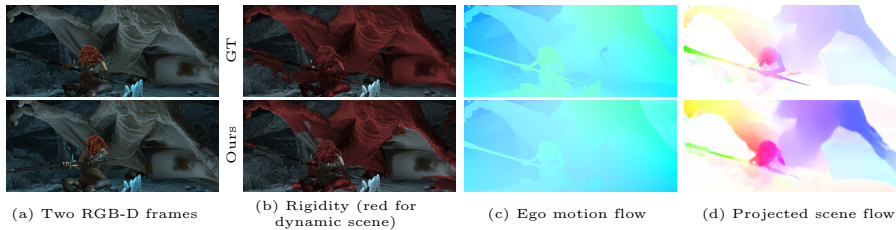
Fig. 1: Our estimated Rigidity (b), Ego-motion Flow (c) and Projected scene flow (d) (bottom row) compared to the ground truth (top row). The rigidity mask allows us to solve for the relative camera transform and compute the 3D motion field given the optical flow.

observed from different view points and the amount of coverage of moving objects in each image is significant. This is mainly because the disambiguation of camera motion (ego-motion) from object motion requires the correct identification of *rigid static structure* of a scene. Unlike other methods solving the problem with piecewise rigid motion [41,19,9], clustering local motions [16], and semantic segmentation [32,45], our network can infer per-pixel rigidity by jointly learning rigidity and the relative camera transform from large-scale dynamic scene data. A brief example of our results is shown in Fig. 1.

Our framework, shown in Fig. 2, takes a sequential image pair with color and depth (RGBD) as the input and mainly focuses on dynamic scenes with a moving camera (e.g., panning), where camera motion and objects motions are entangled in each observation. To solve for 2D correspondences, our framework relies on 2D optical flow, and is not tied to any particular algorithm. We use the method by Sun et al. [33], which we evaluate together with the rigidity network to estimate both ego-motion and scene-motions. The network that learns the per-pixel rigidity also solves for the relative camera pose between two images, and we can accurately refine the pose as a least square problem with the learned dense flow correspondences and rigidity region. To provide better supervision during training and encourage generalization, we develop a tool and methodology that enables the creation of a scalable semi-synthetic RGB-D dynamic scene dataset, which we call *REFRESH*. This dataset combines real-world static rigid background with non-rigid synthetic human motions [36] and provides ground truth color, depth, rigidity, optical flow and camera pose.

In summary, our major contributions are:

1. A learning-based rigidity and pose estimation algorithm for dynamic scenes with a moving camera.
2. An RGBD 3D motion field estimation framework that builds on inference from rigidity, pose, and existing 2D optical flow, which outperforms the state-of-the-art methods.
3. A new semi-synthetic dynamic scene data and its creation tool: REal 3D From REconstruction with Synthetic Humans (REFRESH).
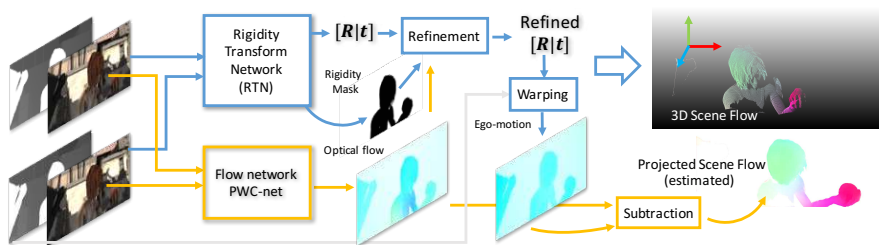
Fig. 2: **An overview of our proposed inference architecture for 3D motion field estimation.** Our method takes two RGB-D frames as inputs independently processed by two networks. The Rigidity Transform Network (RTN) estimates the relative camera transform and rigid/non-rigid regions. The flow network [33] computes dense flow correspondences. We further refine the relative pose with dense flow over the rigid region. With the refined pose, we compute 3D motion field and projected scene flow from the egomotion flow.

2   Related Work

Scene Flow: Scene flow estimation in dynamic scenes brings together fundamental computer vision algorithms in optical flow, and pose estimation of camera and objects. Vedula et al. [37] defined the 3D motion field as *scene flow*, and proposed a method to compute *dense non-rigid* 3D motion fields from a fixed multi-view set-up. Its extension to a moving camera case needs to disambiguate the camera ego-motion from object scene motions in 3D. Due to the intrinsic complexity of such task, existing methods often address it with known camera parameters [1,35] or assume scene motions are piecewise rigid [19,21,9,39,40,42]. When depth is known, scene flow can be more accurately estimated. Quiroga et al. estimates RGB-D scene flow as a rigid flow composited with a non-rigid 6DoF transforms [25]. Sun et al. estimates scene flow as a composition of finite rigid moving objects [32]. Jaimez et al. separately solve rigid region as visual odometry and non-rigid regions as moving clustered patches conditioned on rigidity segmentation [16]. They solve the rigidity segmentation based on the robust residuals of two frame alignment, similar to [23,18] for camera tracking in dynamic environments. All of these approaches use rigidity as a prior, but can fail as the complexity of the dynamic scene increases. None of these methods use learned models. We show that the 3D motion field can be more accurately estimated using learned models for rigidity and optical flow.

Learning Camera Transform and Rigidity: Recently, various learning-based methods have been introduced for the joint estimation of camera transform and depth (or rigid structure) [34,38,49], and rigid motion tracking [3]. Most of them assume that the scene is either static [34], quasi-static (scene motions are minimal and can be dealt as outliers) [49], or that the camera remains static when a rigid scene motion occurs [3]. More recently, a few approaches[45,47] demonstrated the importance of learning *rigidity* to handle dynamic scenes. Wulff et al.

[45] assume the rigidity can be learned by finetuning the semantic segmentation network from a single image, while we posit that rigidity correlates spatially to the epipolar geometry. Yin and Shi [47] unsupervised learn the non-rigid flow residual in the 3D urban scene. We are interested in more general dynamic scenes with unconstrained scene motions observed from moving cameras, and we address this by directly learning the per-pixel rigidity in the supervised manner which can generalize to unseen scenes.

3   Rigidity, Scene Flow and Moving Camera

We focus on solving for the 3D motion field in the physical scene observed from a moving camera, commonly termed as scene flow [16,37]. Here we define the relationship between 2D image correspondences and scene flow in physical 3D scenes with object motions and camera motion derived from relative camera poses between two temporal views.

Let $\mathbf{x}_t \in \mathbb{R}^3$ be the location of a point $\mathbf{x}$ on a non-rigid surface Ω_t of a moving object with respect to a fixed world coordinate system at time t. We define $\delta\mathbf{x}_{t \to t+1}$ as the 3D motion vector of $\mathbf{x}$ from time t to time $t+1$, also referred as scene flow in this paper. When $\mathbf{x}_t$ is observed by a camera with known intrinsics, we define $\pi(\mathbf{x}_t)$ to be the projection of $\mathbf{x}_t$ to image coordinates $\mathbf{u}_t$, and $\pi^{-1}(\mathbf{u}_t, z_t)$ the inverse projection into 3D camera coordinates given the known depth z_t in the camera reference frame.
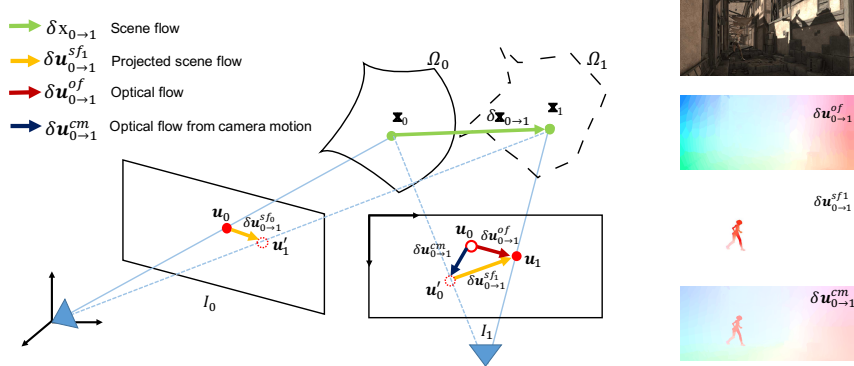

Fig. 3: **The geometry of two-frame scene flow**, where the camera moves from I_0 to I_1, and point $\mathbf{x}_0$ moves to $\mathbf{x}_1$ (green circles), and their projections in the two images are shown as $\mathbf{u}_0, \mathbf{u}_1$ respectively (red circles). Note that $\mathbf{u}_0'$ is a projected location of $\mathbf{x}_0$ in I_1, as if $\mathbf{x}_0$ were observed by I_1, and can be computed by camera motion as $\delta\mathbf{u}_{0 \to 1}^{cm}$, and $\mathbf{u}_0$ in I_1 is visualizing the pixel location it had in I_0. If the camera was static and observed both $\mathbf{x}_0$ and $\mathbf{x}_1$ at the position of I_1, optical flow $\delta\mathbf{u}_{0 \to 1}^{of}$ would be same to a projected scene flow $\delta\mathbf{u}_{0 \to 1}^{sf_1}$. The right image shows each flow in I_1 of dynamic scene under camera panning.

Scene flow, 2D Optical Flow, and Camera Pose Optical flow offers direct 2D associations of measurements in I_t and I_{t+1}. Suppose $\mathcal{C}_t$ is the known camera extrinsics matrix for I_t, then the optical flow $\delta\mathbf{u}_{t\rightarrow t+1}$ from I_t to I_{t+1} can be defined as follows:

$$\delta\mathbf{u}^{of}_{t\rightarrow t+1} = \pi(\mathcal{C}_{t+1}(\mathbf{x}_t + \delta\mathbf{x}_{t\rightarrow t+1})) - \pi(\mathcal{C}_t\mathbf{x}_t) \tag{1}$$

Equation 1 states the two-view geometric relationship between 2D optical flow and 3D scene flow. We can simplify it by considering the camera's relative motion from I_0 to I_1, i.e. assuming $t = 0$ and setting $\mathcal{C}_0$ to identity:

$$\delta\mathbf{u}^{of}_{0\rightarrow 1} = \pi(\mathcal{C}_1(\mathbf{x}_0 + \delta\mathbf{x}_{0\rightarrow 1})) - \pi(\mathbf{x}_0) \tag{2}$$

Given the optical flow $\delta\mathbf{u}^{of}_{0\rightarrow 1}$ and the depth from the RGBD data, the 3D scene flow vector can be computed as:

$$\delta\mathbf{x}_{0\rightarrow 1} = \mathcal{C}_1^{-1}\pi^{-1}(\mathbf{u}_0 + \delta\mathbf{u}^{of}_{0\rightarrow 1}, z_1) - \pi^{-1}(\mathbf{u}_0, z_0) \tag{3}$$

Note that $\mathcal{C}_1$ can be computed from 2D correspondences that follow two-view epipolar geometry [11], and the corresponding points should lie on the rigid and static background structure. This is especially challenging when the scene contains dynamic components (moving objects) as well as a rigid and stationary background structure. As such, identifying inliers and outliers using *rigidity* is a key element for successful relative camera pose estimation, and thus is necessary to achieve reaching accurate scene flow estimation in a dynamic scene [16], which we will discuss in Sec. 4.

Egomotion Flow from a Moving Camera in a Static Scene: When an observed $\mathbf{x}$ in a scene remains static between the two frames, $\delta\mathbf{x}_{0\rightarrow 1} = \mathbf{0}$ and therefore $\mathbf{x}_1 = \mathbf{x}_0$. Then, the observed optical flow is purely induced by the camera motion and we refer it as a camera egomotion flow:

$$\delta\mathbf{u}^{cm}_{0\rightarrow 1} = \pi(\mathcal{C}_1\mathbf{x}_0) - \pi(\mathbf{x}_0) \tag{4}$$

Projected Scene Flow and Rigidity: As described in Fig. 3, the projected scene flow is a projection of a 3D scene flow $\delta\mathbf{x}_{0\rightarrow 1}$ in I_1 if $\mathbf{x}_0$ was observed from I_1, which can be computed from camera ego-motion and optical flow:

$$\delta\mathbf{u}^{sf}_{0\rightarrow 1} = \delta\mathbf{u}^{of}_{0\rightarrow 1} - \delta\mathbf{u}^{cm}_{0\rightarrow 1} \tag{5}$$

The projected scene flow (in a novel view) is also referred as non-rigid residual [25,47]. All locations with zero values in projected scene flow indicate the rigidity region in ground truth data. As demonstrated in Fig. 3, the projected scene flow is a useful tool to evaluate the results of dense scene flow estimation in the 2D domain which requires accurate estimation of both camera pose and optical flow. Thus, we use it as the evaluation metric in Sec. 6.

4   3D Motion Field Estimation Pipeline

We introduce a framework that refines the relative camera transform and the optical flow with a rigidity mask for accurate scene flow estimation. Fig. 2 shows the overview of our proposed pipeline. Given a temporal pair of RGB-D images, we concurrently run the optical flow and rigidity-transform network. The optical flow network [33] offers the 2D correspondence association between frames, and our proposed rigidity-transform network provides an estimate of the camera transform and the rigidity mask.

4.1   Rigidity-Transform Network

Previous work on camera pose estimation using CNNs focused on either static or quasi-static scenes, where scene motions are absent or their amount is minimal [34,38,49]. In dynamic scenes with a moving camera, camera pose estimation can be challenging due to the ambiguity induced by the camera motion and scene (object) motion. Although existing approaches leverage prior information in motion or semantic knowledge [16,32,25,28] to disambiguate the two, the priors are usually not general for different scenes.

  We propose to infer the rigidity from epipolar geometry by a fully-convolutional network that jointly learns camera motion and segmentation of the scene into dynamic and static regions from RGB-D inputs. We represent this rigidity segmentation as a binary mask with the static scene masked as rigid. The rigid scene components will obey the rigid transform constraints induced by the camera ego-motion and serve as the regions of *attention of the camera transform*. We name it rigidity-transform network (RTN), shown in Fig. 4.

RTN: Given a pair of RGB-D frames, we pre-process each frame into a 6 channel tensor $[(u-c_x)/f_x, (v-c_y)/f_y, 1/d, r, g, b]$, from camera intrinsic parameters
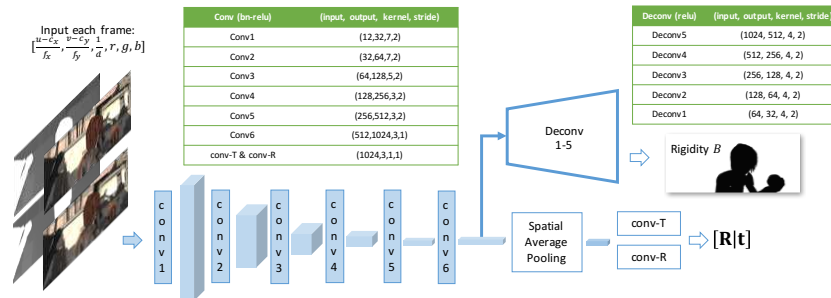

Fig. 4: **Rigidity-Transform network (RTN) architecture** The inputs to the RTN are 12 channel tensors encoded with $[(u-c_x)/f_x, (v-c_y)/f_y, 1/d, r, g, b]$ computed from a pair of RGB-D images and their intrinsics. It is a fully convolutional encoder-decoder architecture predicting pose as a translation and euler angles, and scene rigidity as a binary mask.

$[f_x, f_y, c_x, c_y]$ and the depth d. Due to the range of depth values, this representation is numerical stable in training and delivers good generalization performance. We truncate $1/d$ to the range $[1e-4, 10]$, which can cover scenes of various scales. We concatenate the two-frame tensors to a 12-channel tensor as input to our network. The network is composed of an encoder followed by pose regression and a decoder followed by the rigidity segmentation.

Encoder: We use five stride-2 conv-layers which gradually reduce spatial resolution and one stride-1 convolution as the conv-6 layer. Each convolution is followed by a batchnorm and ReLU layer. In the bottleneck layer, the target is to predict the camera relative translation $\mathbf{t}$ and rotation Θ. After the conv-6 layer, we use a spatial-average pooling (SAP) to reduce the feature into a $1024D$ vector. With two 1×1 convolution layers that output 3 channels, we separately estimate the $\mathbf{t}$ and Θ. We assume the relative camera transformation between two frames is small and thus we represent the rotation $\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_x(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_z(\gamma)$ with Euler angles $\Theta = [\alpha, \beta, \gamma]$. The regression loss is a weighted combination of the robust Huber loss $\rho(\cdot)$ for translation and rotation as:

$$\mathcal{L}_p = \rho(\mathbf{t} - \mathbf{t}^\star) + w_\Theta \rho(\Theta - \Theta^\star) \tag{6}$$

Decoder: The decoder network is composed of five deconvolution (transpose convolution) layers which gradually upsample the conv-6 feature into input image scale and reshape it into the original image resolution. We estimate the rigidity attention as a binary segmentation problem with binary cross-entropy loss $\mathcal{L}_r$. The overall loss is a weighted sum of both loss functions: $\mathcal{L}_c = w_p\mathcal{L}_p + \mathcal{L}_r$.

Enforcing Learning from Two Views: We enforce the network to capture both scene structures and epipolar constraints using two views rather than a single-view short-cut. First, our network is fully convolutional, and we regress the camera pose from the SAP layer which preserves feature distributions spatially. Features for rigidity segmentation and pose regression can interact directly with each other spatially across each feature map. We do not use any skip layer connections. Our experiments in Sec. 6 show that joint learning of camera pose and rigidity can help RTN to achieve better generalization in complex scenes. Second, we randomly use two identical views as input, and a fully rigid mask as output with 20% probability during data augmentation, which prevents the network from only using a single view for its prediction.

4.2   Pose Refinement from Rigidity and Flow

To solve for the 3D motion field accurately from two views from equation 3, we require a precise camera transformation. Moreover, the pose output from RTN may not always precisely generalize to new test scenes. To overcome this, we propose a refinement step based on the estimated rigidity B and bidirectional dense optical flow $\delta\mathbf{u}^{of}_{0\to1}$ and $\delta\mathbf{u}^{of}_{1\to0}$ (with forward and backward pass). We view the estimation of $\mathcal{C}_1$ as a robust least square problem as:

$$\underset{\mathcal{C}_1}{\operatorname{argmin}} \sum_{\{\mathbf{x}_0,\mathbf{x}_1\}\in\Omega(B)} [\mathbf{I}]\rho(\mathcal{C}_1\mathbf{x}_0 - \mathbf{x}_1) \tag{7}$$

where $\mathbf{x}_i = \pi^{-1}(\mathbf{u}_i, z_i)$ in all background regions B, predicted by the RTN. [$\mathbf{I}$] is an Iverson bracket for all the inlier correspondences. We filter the inlier correspondences in several steps. We first use forward-backward consistency check for bidirectional optical flow with a threshold of 0.75 to remove all flow correspondences which are not consistent. The removed region approximates the occlusion map O. We use a morphological operator with patch size 10 to dilate B and O to further remove the outliers on boundaries. From all correspondences, we uniformly sample bidirectional flow correspondences with a stride of 4 and select 1e4 points among them that are closest to the camera viewpoint. These help to solves the optimization more efficiently and numerically stable. We also use the Huber norm $\rho(\cdot)$ as a robust way to handle the remaining outliers. We solve equation 7 efficiently via Gauss-Newton with $\mathcal{C}_1$ initialized from the RTN output. Note that in most cases correspondences are mostly accurate, the initialization step trivially helps but can also be replaced by an identity initialization.

5   REFRESH Dataset

Training our network requires a sufficient amount of dynamic RGB-D images over scenes and ground truth in the form of known camera pose, rigidity mask, and optical flow. However, acquiring such ground truth from the real-world data is difficult or even infeasible. Existing dataset acquisition tools include rendered animations like SINTEL[2] and Monka[20], and frames captured from games [26]. SINTEL [2] has a small number of frames, so we use it for testing instead of training. Most approaches render scenes using rigid 3D object models [31,7,20] with the concept. Among all existing tools and datasets, only Things3D[20] provides sufficient 3D training samples for learning 3D flow with moving camera ground truth. However, it only uses a small set of 3D objects with textured images at infinity as static scene context and rigid objects as the dynamic scene, which does not provide realistic 3D scene structure for motion estimation that can generalize well.

  To overcome the dataset issue, we propose a semi-synthetic scene flow dataset: REal 3D from REconstruction with Synthetic Humans, which we name as **REFRESH**. For this task we leverage the success of state of the art 3D reconstruction systems [5,10,46], which directly provide dense 3D meshes and optimized camera trajectories. We use a pre-captured RGB-D dataset and create dynamic 4D scenes by rendering non-rigid 3D moving objects with pre-defined trajectories. We overlay synthetic objects over the original footage to obtain a composite image with the ground truth as shown in Fig. 5.

Real 3D Reconstructed Scenes: We use the 3D meshes created with Bundle-Fusion [5]. The authors released eight reference 3D meshes with the 25K input RGB-D images, camera intrinsic and extrinsic parameters.

Synthetic humans: We create non-rigid scene elements with the method introduced in SURREAL [36]. Each synthetic body is created from realistic articulated human body models [17] and pose actions are from the CMU MoCap database [15] with more than 20K sequences of 23 action categories. The human

(a) reconstructed 3D mesh   (b) raw color image   (c) raw depth image   (d) synthetic humans

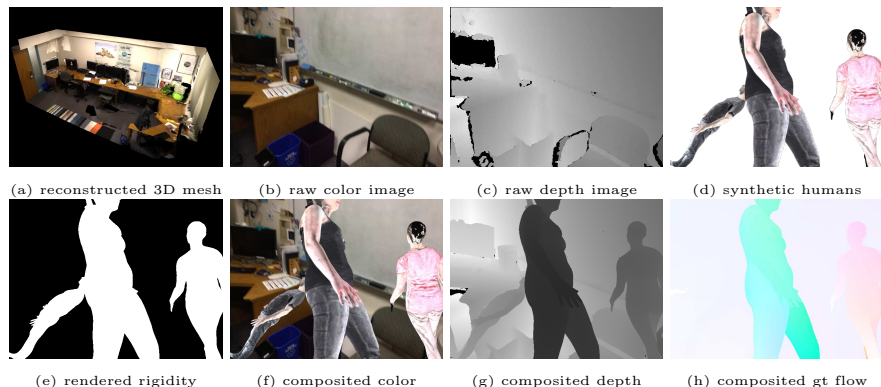(e) rendered rigidity   (f) composited color   (g) composited depth   (h) composited gt flow

Fig. 5: **REFRESH dataset creation pipeline** With a captured RGB-D trajectory, the scene is reconstructed as a 3D mesh by BundleFusion [5] (a), with raw RGB-D input as (b) and (c). With sampled frames from the camera trajectory, we load synthetic human models [36] with motions randomly into the 3D as (d), and render the rigidity mask (e), Finally we composite the rendered synthetic ground truth with its corresponding rendered 3D views and the final semi-synthetic RGB-D views (f) and (h), with optical flow ground truth as (i).

textures are composed of SMPL CAESAR scans and real clothing registered with 4Cap [24]. We create each synthetic human with random gender, body shape, cloth texture, action and their positions in the 3D scene which guarantees the diversity of dynamic scenes. We control the visibility of human models along the trajectory by putting the pelvis point of each human model in the free space w.r.t. the ego-centric viewpoint from a selected frame along the trajectory. The free space is sampled by the corresponding depth. For every 100 frames, we select n frames (n sample from $\sim \mathcal{N}(15, 5)$) and insert n human models into the scene.

Rendering and ground-truth generation: We use Cycles from the Blender [3] suite as our rendering engine. The lighting is created using spherical harmonics, as in Varol et al. [36]. First, we set the virtual camera using the same 3D scene camera intrinsic and spatial resolution. The camera extrinsic follows the real-data trajectory (computed from BundleFusion [5]). Thus, we can use the raw color image rather than rendered image as background texture which is photo-realistic and contains artifacts such as motion blur. With the same camera settings, we separately render the 3D reconstructed static mesh and the synthetic humans, and composite them using alpha-matting. Different from the color image, the depth map is rendered from the 3D mesh, which is less noisy and more complete than raw depth. Since the camera movement during the 3D acquisition is small between frames, we sub-sample frames at intervals of [1,2,5,10,20] to create larger motions. We employ a multi-pass rendering approach to generate depth, optical flow and rigidity mask as our ground truth.

---

[3] Blender: https://www.blender.org/

6   Experiments

We implemented the RTN in PyTorch, and the pose refinement in C++ with
GTSAM 4.0 [6]. The PWCNet [33] is trained in Caffe. We integrate all the
modules through Python. We use 68K images from our REFRESH dataset for
training [4]. We train RTN from scratch using weight initialization from He et
al.[12] and Adam optimizer ($\beta_1 = 0.9$ and $\beta_2 = 0.999$, learning rate of $2e^{-4}$) on
3 GPUs for 12 epochs. During training, the rigidity mask loss is accumulated
over 5 different scales with balanced weights, and we choose $w_\Theta = 100$. We follow
the same training as PWC-net Sun et al. [33]. We will release our code, datasets
and REFRESH toolkit [5].

　　We evaluate our approach under various settings to show the performance
of rigidity and pose estimation and their influence on scene flow estimation.
For the effective analysis in scenes with different levels of non-rigid motions, we
create a new test split from SINTEL data [2] based on the non-rigid number of
pixels percentage. In Sec. 6.1, we provide a comparison of the performance with
different settings for RTN, refinement and other state-of-the-arts methods. In
Sec. 6.2, we qualitative evaluate of our method using real world images. Please
also refer to our video for more qualitative evaluations.

6.1   Quantitative Evaluations

We first evaluate our approach on the challenging SINTEL dataset [2], which
is a 3D rendered animation containing a sequence of 23 dynamic scenes with
cinematic camera motion. The dataset has two versions with different render-
ing settings: *clean* and *final*. The latter set contains motion blur and depth of
field effects, which are not present in the *clean* set. Since the official SINTEL
test dataset does not provide RGB-D 3D flow evaluation, we split the SINTEL
training set into train, validation, and test split. For the test split, to effec-
tively evaluate and analyze the impact of different levels of non-rigid motions
in the estimation, we choose *alley_2*(1.8%), *temp_2*(5.8%), *market_5*(27.04%),
ambush_6(38.96%), *cave_4*(47.10%), where (·) indicates the average non-rigid
regions in each scene sequence. These examples also contain a sufficient amount
of camera motion. We use the first 5 frames in the rest of the 18 scenes as a
validation set, and the remaining images for training in our finetuning setting.

　　We show our quantitative evaluations using flow metric in Table 1, rel-
ative pose metric in Table 2, and the rigidity IOU in Table 3. We list the
end-point-error (EPE) in the ego-motion flow (EF) and projected scene flow
(PSF) as defined in Sec.3. Our proposed metrics overcomes the traditional dif-
ficulty of 3D motion flow evaluation. We compare our method to two state-of-
art optimization-based RGB-D scene flow solutions: SRSF [25] and VO-SF [16]
which estimate the camera pose as part of the solution to flow correspondence.

---

[4] More details about the dataset are included in the supplementary materials.
[5] Code repository: `https://github.com/NVlabs/learningrigidity.git`

Two-frame Overlay

Ground Truth Rigidity

Our Rigidity (No Finetuning)

Semantic Rigidity [45] (No Finetuning)

Ground Truth Projected Scene Flow

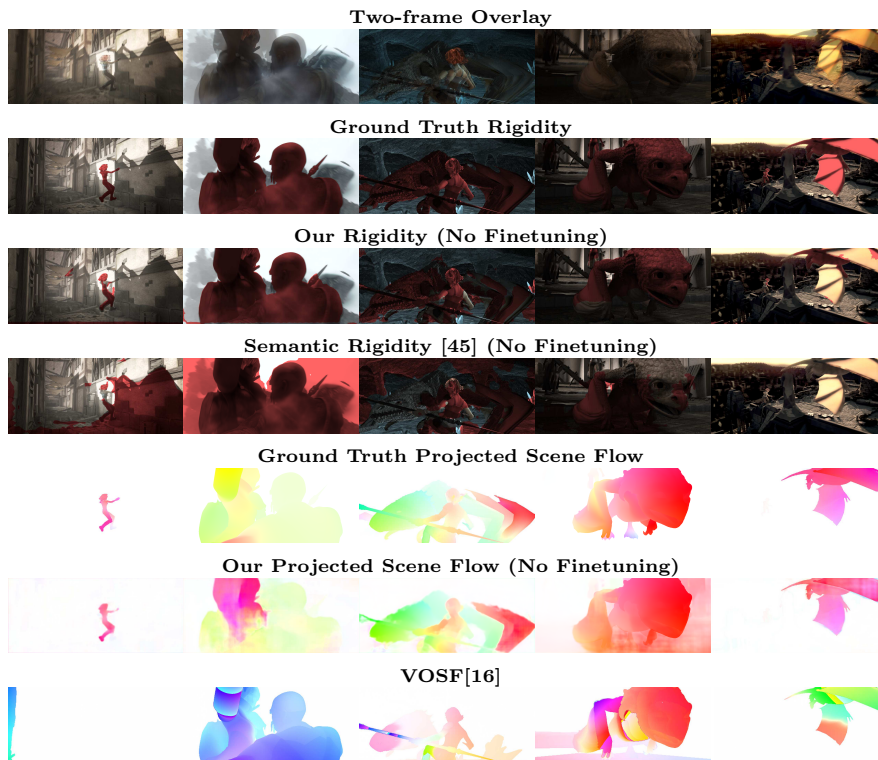**Our Projected Scene Flow (No Finetuning)**

VOSF[16]

Fig. 6: **Qualitative visualization** on our SINTEL test split. We compare our rigidity prediction with the output using semantic rigidity [45] trained on our REFRESH dataset and our projected scene flow with output of VOSF [16].

In addition, we evaluate three types of baselines. The first one solves the refinement stage from flow without any inputs from RTN (*Refine Only*), which assumes rigidity often dominates the scene [23,16,18]. Secondly, we use three-point RANSAC to calculate the camera pose from the flow. Third, to fairly evaluate the rigidity of (RTN) and its generalization, we compare it to semantic rigidity estimation [45], which assumes that the non-rigid motion can be predicted from its semantic labeling. We follow Wulff et al [45] and use the DeepLab [4] architecture initialized from the pre-trained MS-COCO model, but trained over the same data we used for our model. In the pose refinement stage, we substitute our rigidity from RTN with the semantic rigidity. For the fine-tuned evaluation on SINTEL, we re-train both our RTN and the semantic rigidity network. All methods use the same optical flow network and weights, and all use the same depth from SINTEL ground truth. The qualitative comparison is shown in Fig.6.

The Flow Metrics in Table 1 show that: (1) compared to SRSF[25] and VOSF [16], our proposed algorithm with learned rigidity can improve scene flow accuracy by a significant margin with no further fine-tuning (NO FT) (rows (a),(b)vs(h); (k),(l)vs(r)); (2) the rigidity mask from our RTN performs bet-

Table 1: Quantitative evaluation in flow residuals using SINTEL dataset on our test split. The ratio of Nonrigid (NR) Region indicates the average ratio of pixels in the scene which represents the complexity of dynamic motion in the scene. We report the EPE in egomotion flow (EF) and projected scene flow (PSF). For all the baseline methods in both non-finetuning (NO FT) and finetuning (FT) setting, we use the same optical flow network trained as our method. The lowest residual under the same setting (e.g. NO FT, clean set) is highlighted as **bold**.

		NR Region<10%				NR Region 10%-40%				NR Region>40%		All Test	
		alley_2		temple_2		market_5		ambush_6		cave4		Average	
		EF	PSF	EF	PSF	EF	PSF	EF	PSF	EF	PSF	EF	PSF
	CLEAN (no motion blur)												
	(a) SRSF [25]	4.24	7.25	7.59	16.55	25.26	31.67	17.84	37.21	10.77	11.82	12.47	18.57
	(b) VOSF [16]	6.53	1.13	5.13	10.36	16.02	35.24	13.39	28.31	6.05	9.30	8.86	15.24
NO FT	(c) Refine only	0.29	0.48	0.90	2.95	8.81	22.34	3.59	14.39	2.18	5.88	3.09	8.47
NO FT	(d) Semantic[45]+Refine	0.25	0.53	1.07	3.87	5.77	15.74	1.70	9.58	0.85	4.34	1.96	6.42
	(e) RANSAC+Flow	0.31	0.57	0.47	2.73	7.36	19.19	3.86	14.89	2.17	5.94	2.69	7.78
	(f) RTN(use Things[20])+Refine	0.34	0.60	1.47	3.98	7.21	18.73	21.84	23.97	1.17	4.90	4.20	5.85
	(g) RTN(no-pose)+Refine	**0.13**	**0.45**	0.49	2.79	5.78	16.24	3.72	16.92	1.67	5.37	2.07	7.09
	(h) **RTN+Refine**	0.18	0.48	**0.46**	**2.72**	**1.61**	**11.86**	**0.97**	**8.61**	0.63	**4.05**	**0.74**	**5.10**
FT	(i) Semantic[45]+Refine	0.19	**0.46**	0.50	2.73	2.73	13.45	1.13	9.94	2.07	5.87	1.35	5.98
FT	(j) **RTN+Refine**	0.18	0.47	**0.42**	**2.64**	1.69	11.53	**0.47**	**7.74**	0.91	4.34	**0.77**	**5.03**
	FINAL (with motion blur)												
	(k) SRSF [25]	4.33	7.78	7.59	15.51	24.93	31.29	17.26	39.08	10.80	13.29	12.37	18.86
	(l) VOSF [16]	6.29	1.54	5.69	8.91	15.99	35.17	13.37	24.02	6.23	9.28	8.96	14.61
NO FT	(m) Refine only	0.28	0.57	0.90	3.77	8.80	20.64	3.59	20.41	2.18	6.52	3.09	8.95
NO FT	(n) Semantic[45]+refine	0.25	0.52	0.96	3.83	>100	>100	20.23	35.46	11.05	12.81	>100	>100
	(o) RANSAC+Flow	0.36	0.61	0.62	3.41	4.68	18.69	5.79	20.86	2.28	6.55	2.31	8.47
	(p) RTN(use Things[20])+Refine	0.25	0.52	5.06	9.82	4.88	16.99	33.44	52.21	1.05	5.07	5.44	11.88
	(q) RTN(no-pose)+Refine	0.19	0.48	**0.82**	**3.58**	2.15	13.97	3.34	20.02	1.52	5.72	1.36	7.14
	(r) **RTN+Refine**	**0.18**	**0.47**	0.88	3.93	**0.79**	**11.87**	2.82	19.42	0.66	4.66	0.82	6.29
FT	(s) Semantic[45]+Refine	**0.19**	**0.48**	1.91	5.19	1.58	13.02	2.58	19.11	2.13	6.50	1.55	7.39
FT	(t) **RTN+Refine**	0.21	**0.48**	**0.66**	**3.27**	**0.97**	**11.35**	2.34	19.08	0.74	**4.75**	**0.79**	**6.12**

ter than the single-view semantic segmentation based approach [45], particularly in the more realistic *final pass* with no fine-tuning (row (d)vs(g),(h); (n)vs(q),(r)); (3) as shown in *RTN+refine*, the simultaneous learning of rigidity with pose transform achieves better generalization than learning rigidity alone (row (g)vs(h); (q)vs(r)); (4) RTN trained on our dataset generalizes better compared to the same RTN trained using Things3D[20] (row (f)vs(h); (p)vs(r)); and (5) the final approaches consistently outperforms all baselines. Note that the semantic rigidity [45] can achieve better performance (from Table 1) relying on fine-tuning on SINTEL, our learned rigidity can generalize to unseen complex scenes and perform as well as the fine-tuned model. Our rigidity prediction can capture unseen objects well, as shown by the dragon in Fig. 6.

The Pose Metrics evaluations in Table 2 include two other baselines: depth-based ORB-SLAM[22] and point cloud registration [48]. As mentioned, the ac-

Table 2: Quantitative evaluation in relative camera transfrom using on our SIN-TEL test split. We report the relative pose error [30] (RPE) composed of translation (t) error and rotation error (r) in Euler angles (degree) in SINTEL depth metric averaged on from outputs using *clean* and *final* pass.

	NR Region <10%			NR Region 10% - 40%			NR Region >40%		All Test	
	alley_2		temple_2		market_5		ambush_6		cave4		AVERAGE
	RPE(t)	RPE(r)	RPE(t)	RPE(r)	RPE(t)	RPE(r)	RPE(t)	RPE(r)	RPE(t)	RPE(r)	RPE(t)	RPE(r)
ORB-SLAM [22]	0.0300	0.0190	0.1740	0.0220	0.1500	0.0160	0.0550	0.0280	0.0167	0.0277	0.0894	0.0218
SRSF [25]	0.0487	0.0141	0.1763	0.0117	0.1566	0.0105	0.0672	0.0729	0.0218	0.0150	0.0980	0.0180
VOSF[16]	0.1043	0.0316	0.1055	0.0155	0.0605	**0.0006**	0.0375	0.0190	0.0438	0.0046	0.0750	0.0136
Registration [1]	0.0400	0.0094	0.3990	0.0381	0.0269	0.0073	0.0698	0.0225	0.0551	0.0076	0.1251	0.0162
RANSAC+Flow	0.0026	0.0047	0.0258	0.0033	0.0446	0.0043	0.0318	0.0082	0.0318	0.0411	0.0267	0.0039
Our RTN Pose	0.0349	0.0237	0.1589	0.0120	0.1520	0.0208	0.0455	0.0493	0.0233	0.0212	0.0883	0.0220
Ours (no ft)	**0.0015**	**0.0036**	**0.0215**	**0.0010**	**0.0059**	0.0009	**0.0153**	**0.0061**	**0.0053**	**0.0009**	**0.0091**	**0.0020**

Table 3: Evaluation of rigidity using mean IOU of rigid and nonrigid scenes.

mean IOU	REFRESH val	SINTEL clean val	SINTEL final val
Semantic Rigidity [2] trained on REFRESH	0.934	0.392	0.446
RTN trained on Things [4]	-	0.283	0.286
RTN trained on our REFRESH	**0.956**	**0.542**	**0.627**

curacy of all relevant methods in dynamic scenes with moving camera highly relies on the ability ignore the non-rigid surfaces. As shown in the table, our pose directly predicted from RTN can achieve same or better accuracy with all relevant methods, and our final solution *without fine-tunning* can out-perform all state-of-the-art methods by a significant margin.

The Rigidity Metric in Table 3 further shows the generalization in rigidity estimation. Our approach trained on our dataset generalizes significant better compared to the same approach trained using Things3D[20] and the semantic rigidity[45] using the same data.

6.2   Evaluation on Real-world Images

To test our algorithm in real-world dynamic scenarios, we use three TUM RGB-D sequences [30] which contains dynamic motions observed from a moving Kinect camera. The depth input is noisy with missing observations and the color images contain severe motion blur. We use the raw color and depth input with provided calibrated camera intrinsics as input, and mark the regions as invalid region when the depth value is not within $[0.1, 8]$. In invalid regions, we ignore the rigidity prediction and treat the flow correspondence as outliers. Considering there is no 3D motion flow ground truth for our real data, we visualize the rigidity prediction and projected scene flow to qualitatively show the performance in Fig. 7. Our results show that our trained model on semi-synthetic data can also generalize well to real noisy RGB-D data with significant motion blur.
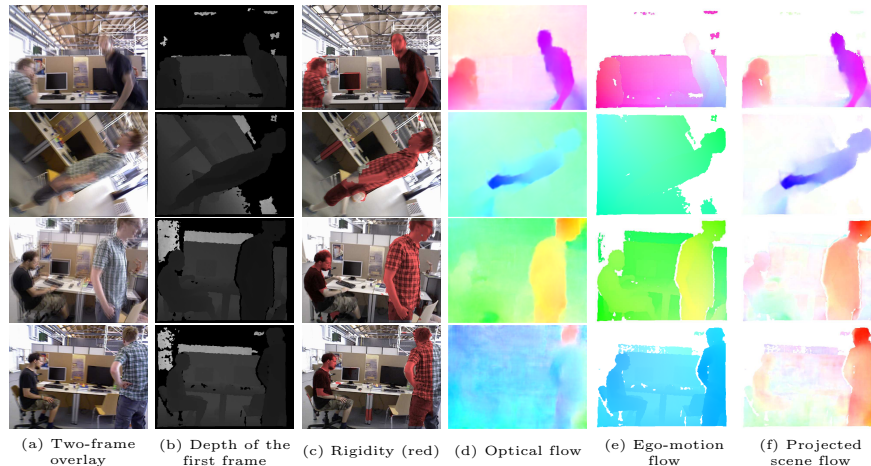
|(a) Two-frame overlay|(b) Depth of the first frame|(c) Rigidity (red)|(d) Optical flow|(e) Ego-motion flow|(f) Projected scene flow|

Fig. 7: **Qualitative visualization** of dynamic sequences in TUM [30] sequences.

7   Conclusion and Future Work

We have presented a learning-based approach to estimate the rigid regions in dynamic scenes observed by a moving camera. Furthermore, we have shown that our framework can accurately compute the 3D motion field (scene flow), and the relative camera transform between two views. To provide better supervision to the rigidity learning task and encourage the generalization of our model, we created a novel semi-synthetic dynamic scene dataset, REFRESH, which contains real-world background scenes together with synthetic foreground moving objects. Through various tests, we have shown that our proposed method can outperform state-of-the-art solutions. We also included a new guideline for dynamic scene evaluation regarding the amount of scene motion and camera motion.

We observed some cases where the rigidity mask deviates from the ground-truth. We noticed that in these situations the moving object size is small, or the temporal motions between the two frames are small. In these cases, the error and deviations scales are small, which does not significantly affect the 3D scene flow computed as a result. Note that the success of this method also depends on the accuracy of optical flow. In scenarios when the optical flow fails or produces a noisy result, the errors in the correspondences will also propagate to 3D motion field. In future work, we can address these problems by exploiting rendering more diverse datasets to encourage generalization in different scenes. We will also incorporate both rigidity and optical flow to refine the correspondence estimation and explore performance improvements with end-to-end learning, including correspondence refinement and depth estimation from RGB inputs.

References

1. Basha, T., Moses, Y., Kiryati, N.: Multi-view scene flow estimation: A view centered variational approach. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 1506–1513 (2010)
2. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: European Conf. on Computer Vision (ECCV). pp. 611–625. The Royal Society (2012)
3. Byravan, A., Fox, D.: SE3-Nets: Learning rigid body motion using deep neural networks. In: IEEE Intl. Conf. on Robotics and Automation (ICRA). pp. 173–180. IEEE (2017)
4. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. arXiv:1606.00915 (2016)
5. Dai, A., Nießner, M., Zollöfer, M., Izadi, S., Theobalt, C.: BundleFusion: real-time globally consistent 3D reconstruction using on-the-fly surface re-integration. ACM Transactions on Graphics 2017 (TOG) (2017)
6. Dellaert, F.: Factor graphs and GTSAM: a hands-on introduction. Tech. Rep. GT-RIM-CP&R-2012-002, GT RIM (Sept 2012), https://research.cc.gatech.edu/borg/sites/edu.borg/files/downloads/gtsam.pdf
7. Dosovitskiy, A., Fischer, P., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., v.d. Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: Intl. Conf. on Computer Vision (ICCV). pp. 2758–2766 (2015)
8. Geiger, A., Lauer, M., Wojek, C., Stiller, C., Urtasun, R.: 3D traffic scene understanding from movable platforms. IEEE Trans. Pattern Anal. Machine Intell. (2014)
9. Golyanik, V., Kim, K., Maier, R., Nießner, M., Stricker, D., Kautz, J.: Multiframe scene flow with piecewise rigid motion. In: Intl. Conf. on 3D Vision. Qingdao, China (October 2017)
10. Handa, A., Whelan, T., McDonald, J., Davison, A.: A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In: IEEE Intl. Conf. on Robotics and Automation (ICRA). Hong Kong, China (May 2014)
11. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, ISBN: 0521540518, second edn. (2004)
12. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV). pp. 1026–1034. Intl. Conf. on Computer Vision (ICCV), IEEE Computer Society, Washington, DC, USA (2015). https://doi.org/10.1109/ICCV.2015.123, http://dx.doi.org/10.1109/ICCV.2015.123
13. Hung, C.H., Xu, L., Jia, J.Y.: Consistent binocular depth and scene flow with chained temporal profiles. Intl. J. of Computer Vision **102**(1-3), 271–292 (2013)
14. Innmann, M., Zollhöfer, M., Nießner, M., Theobalt, C., Stamminger, M.: VolumeDeform: Real-time volumetric non-rigid reconstruction. In: European Conf. on Computer Vision (ECCV) (2016)
15. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: Human3.6M: large scale datasets and predictive methods for 3d human sensing in natural environments. IEEE Trans. Pattern Anal. Machine Intell. **36**(7), 1325–1339 (jul 2014)
16. Jaimez, M., Kerl, C., Gonzalez-Jimenez, J., Cremers, D.: Fast odometry and scene flow from rgb-d cameras based on geometric clustering. In: IEEE Intl. Conf. on Robotics and Automation (ICRA) (2017)

17. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: SMPL: A skinned multi-person linear model. SIGGRAPH Asia **34**(6), 248:1–248:16 (Oct 2015)
18. Lv, Z.: KinfuSeg: a Dynamic SLAM Approach Based on Kinect Fusion. Master's thesis, Imperial College London (2013)
19. Lv, Z., Beall, C., Alcantarilla, P.F., Li, F., Kira, Z., Dellaert, F.: A continuous optimization approach for efficient and accurate scene flow. In: European Conf. on Computer Vision (ECCV). pp. 757–773. Springer (2016)
20. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 4040–4048 (2016)
21. Menze, M., Geiger, A.: Object scene flow for autonomous vehicles. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 3061–3070 (2015)
22. Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. IEEE Transactions on Robotics **33**(5), 1255–1262 (2017). https://doi.org/10.1109/TRO.2017.2705103
23. Newcombe, R.A., Fox, D., Seitz, S.M.: DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (June 2015)
24. Pons-Moll, G., Romero, J., Mahmood, N., Black, M.J.: Dyna: A model of dynamic human shape in motion. SIGGRAPH **34**(4), 120:1–120:14 (Aug 2015)
25. Quiroga, J., Brox, T., Devernay, F., Crowley, J.: Dense semi-rigid scene flow estimation from RGB-D images. In: European Conf. on Computer Vision (ECCV). vol. 8695 LNCS, pp. 567–582 (2014)
26. Richter, S.R., Hayder, Z., Koltun, V.: Playing for benchmarks. In: Intl. Conf. on Computer Vision (ICCV) (2017)
27. Scharwächter, T., Enzweiler, M., Franke, U., Roth, S.: Stixmantics: A medium-level model for real-time semantic scene understanding. In: European Conf. on Computer Vision (ECCV) (2014)
28. Sevilla-Lara, L., Sun, D., Jampani, V., Black, M.J.: Optical flow with semantic segmentation and localized layers. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (Jun 2016)
29. Shashua, A., Gdalyahu, Y., Hayun, G.: Pedestrian detection for driving assistance systems: Single-frame classification and system level performance. In: IEEE Intelligent Vehicles Symposium (IV). pp. 1–6 (June 2004)
30. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS) (Oct 2012)
31. Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for CNN: viewpoint estimation in images using cnns trained with rendered 3d model views. In: Intl. Conf. on Computer Vision (ICCV) (December 2015)
32. Sun, D., Sudderth, E.B., Pfister, H.: Layered rgbd scene flow estimation. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 548–556. IEEE (2015)
33. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2018)
34. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: DeMoN: depth and motion network for learning monocular stereo. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2017)

35. Valgaerts, L., Bruhn, A., Zimmer, H., Weickert, J., Stoll, C., Theobalt, C.: Joint estimation of motion, structure and geometry from stereo sequences. In: European Conf. on Computer Vision (ECCV). vol. 6314 LNCS, pp. 568–581 (2010)
36. Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M.J., Laptev, I., Schmid, C.: Learning from Synthetic Humans. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2017)
37. Vedula, S., Baker, S., Rander, P., Collins, R., Kanade, T.: Three-dimensional scene flow. In: Intl. Conf. on Computer Vision (ICCV). vol. 2, pp. 722–729 vol.2 (1999)
38. Vijayanarasimhan, S., Ricco, S., Schmid, C., Sukthankar, R., Fragkiadaki, K.: SfM-Net: learning of structure and motion from video. arXiv **abs/1704.07804** (2017)
39. Vogel, C., Schindler, K., Roth, S.: 3D scene flow estimation with a rigid motion prior. In: Intl. Conf. on Computer Vision (ICCV). pp. 1291–1298 (2011)
40. Vogel, C., Schindler, K., Roth, S.: Piecewise rigid scene flow. In: Intl. Conf. on Computer Vision (ICCV). pp. 1377–1384 (2013)
41. Vogel, C., Schindler, K., Roth, S.: 3d scene flow estimation with a piecewise rigid scene model. Intl. J. of Computer Vision **115**(1), 1–28 (2015)
42. Vogel, C., Schindler, K., Roth, S.: View-consistent 3d scene flow estimation over multiple frames. In: European Conf. on Computer Vision (ECCV). vol. 8692, pp. 263–278 (2014)
43. Wang, P., Li, W., Gao, Z., Zhang, Y., Tang, C., Ogunbona, P.: Scene Flow to Action Map: A new representation for RGB-D based action recognition with convolutional neural networks. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (July 2017)
44. Wedel, A., Brox, T., Vaudrey, T., Rabe, C., Franke, U., Cremers, D.: Stereoscopic scene flow computation for 3D motion understanding. Intl. J. of Computer Vision **95**(1), 29–51 (2011)
45. Wulff, J., Sevilla-Lara, L., Black, M.J.: Optical flow in mostly rigid scenes. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (July 2017)
46. Xiao, J., Owens, A., Torralba, A.: SUN3D: a database of big spaces reconstructed using SfM and object labels. In: Intl. Conf. on Computer Vision (ICCV). pp. 1625–1632. IEEE Computer Society, Los Alamitos, CA, USA (2013)
47. Yin, Z., Shi, J.: GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2018)
48. Zhou, Q., Park, J., Koltun, V.: Fast global registration. In: European Conf. on Computer Vision (ECCV). pp. 766–782 (2016)
49. Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2017)