Real-time Halfway Domain Reconstruction of Motion and Geometry

Lucas Thies[1] Michael Zollhöfer[2] Christian Richardt[2,3,4] Christian Theobalt[2] Günther Greiner[1]

[1]University of Erlangen-Nuremberg [2]Max Planck Institute for Informatics [3]Intel Visual Computing Institute [4]University of Bath

{lucas.thies,guenther.greiner}@fau.de {mzollhoef,richardt,theobalt}@mpi-inf.mpg.de

Abstract

We present a novel approach for real-time joint reconstruction of 3D scene motion and geometry from binocular stereo videos. Our approach is based on a novel variational halfway-domain scene flow formulation, which allows us to obtain highly accurate spatiotemporal reconstructions of shape and motion. We solve the underlying optimization problem at real-time frame rates using a novel data-parallel robust non-linear optimization strategy. Fast convergence and large displacement flows are achieved by employing a novel hierarchy that stores delta flows between hierarchy levels. High performance is obtained by the introduction of a coarser warp grid that decouples the number of unknowns from the input resolution of the images. We demonstrate our approach in a live setup that is based on two commodity webcams, as well as on publicly available video data. Our extensive experiments and evaluations show that our approach produces high-quality dense reconstructions of 3D geometry and scene flow at real-time frame rates, and compares favorably to the state of the art.

1. Introduction

Many tasks in computer vision, such as performance capture, free-viewpoint video and 3D motion understanding, require dynamic scene reconstruction from only a few video cameras. Dynamic scene reconstruction comprises the estimation of 3D geometry and its motion over time, which has been coined *scene flow* by Vedula et al. [49], in analogy to 'optical flow' which describes 2D motion of points over time. The 3D motion of points cannot be accurately estimated in isolation from the 3D geometry as depth information is required for computing the 3D motion of points. Unlike structure-from-motion, scene flow does not assume a static scene, but objects in the scene can move about freely and deform non-rigidly. The estimation of scene flow from RGB images in a geometrically well-constrained way therefore requires as input two sets of stereo (binocular) images for consecutive time steps. In recent years, scene flow has been an important ingredient in many real-world applications, including those mentioned before, like 3D motion understanding in automotive scenarios [34, 52], facial performance capture [48, 53] and free-viewpoint video [31].

Recently, real-time capable approaches for computing scene flow from specialized RGB-D cameras were proposed. However, existing approaches for computing dense scene flow from RGB images (without depth) require considerable computation time, in the order of minutes per frame (see KITTI scene flow evaluation 2015 [34]). This is because most dense scene flow approaches use variational formulations that result in large systems of equations with millions of unknowns that are computationally expensive to solve, despite efficient coarse-to-fine hierarchical optimization schemes. The high computational complexity severely limits the applicability of these existing approaches.

In this paper, we thus propose the first approach for estimating dense scene flow and scene geometry at real-time rates ($\geqslant$30 Hz) from binocular RGB video. Even with the computational processing power of modern GPUs, existing dense binocular approaches are far from real-time performance, or achieve at best near-real-time rates using FPGAs [52]. To achieve the real-time goal on a standard computer, we therefore introduce a new scene flow parameterization in terms of a spatiotemporal halfway domain that lies conceptually halfway between both camera viewpoints and between the two time steps (see flow illustration in Figure 1). In addition, we propose a novel, mesh-based coarse-to-fine warping scheme that accumulates pixel-level evidence within grid cells while dramatically reducing the number of unknown flow variables that need to be optimized. During the coarse-to-fine warping, we leverage the GPU to efficiently bootstrap the computation of occlusions masks and illumination correction maps. We implemented a new data-parallel optimization strategy that incorporates robust norms on a commodity graphics card, which enables our scene flow technique to be the first dense RGB-only method to achieve real-time frame rates (30 Hz). We show reconstruction results obtained with our live stereo webcam setup. In addition, we compare to the scene flow approach of Valgaerts et al. [47] (on the datasets of Valgaerts et al. [48]) and show reconstruction results on high-quality stereo pairs [8, 44].

2. Related Work

Stereo correspondence – the computation of a disparity map from two rectified input images – is a long-standing problem in computer vision that has seen a large variety of techniques published over the last few decades [3, 43]. While our approach is not primarily aimed at computing just stereo disparity maps, as we also compute 3D scene flow over time, our approach can be adapted to stereo reconstruction by disabling the temporal component. We therefore start by briefly reviewing the most relevant work from the stereo literature. The first real-time approaches for stereo correspondence required custom hardware [7], but the advent of graphics processing units (GPUs) made it even easier to achieve real-time performance. At first, techniques implemented simple local stereo matching approaches with different cost aggregation schemes such as sum-of-squared-differences [57], adaptive aggregation [51] or others [18]. Later, more advanced stereo matching techniques were also ported and adjusted to work efficiently on GPUs, such as hierarchical belief propagation [56] or adaptive support weights using the bilateral grid [41]. However, these approaches often sacrifice quality for speed, which is an inherent trade-off. High-resolution, high-quality disparity maps can for example be computed with approaches based on bilateral space stereo [4] or mesh-based image warping [8, 44, 45, 60]. Most recently, deep neural networks have shown remarkable performance in stereo correspondence finding [11, 32, 59], and even directly estimating homographies [13].

The goal of **scene flow** techniques is to compute the motion within a scene over time, for every visible 3D point between two time steps. Many approaches have been proposed in recent years for computing scene flow from different visual input modalities, in particular RGB or RGB-D videos. The proposed approaches include voxel coloring based on controlled multi-view camera setups [49], tracking of points and surfels [14], growing of correspondence seeds [10], non-rigid scene registration [5], particle-based estimation [19], semi-global [55] or wide-baseline matching [40]. The most common class of scene flow approaches, including ours, are variational methods, for both RGB [6, 24, 25, 37, 39, 47, 52] and RGB-D inputs [16, 27, 46], as they provide dense, continuous and strongly regularized solutions.

Many recent methods focus on estimating scene flow from RGB-D videos captured with consumer depth cameras [16, 19, 22, 23, 27, 29, 38, 46, 58]. Some of them also achieve real-time frame rates [2, 26], but in contrast to our RGB-based method, they use a special sensor to obtain depth maps. The best-performing methods on the (RGB-only) KITTI 2015 scene flow benchmark [34] enforce strong motion priors, like affine [62] or piece-wise rigid motions [34, 50] that are ideal for the driving scenario. However, our goal is to reconstruct general non-rigid dynamic scenes with a stereo camera pair, in which case many of these motion priors may be violated and counterproductive. Recently released datasets with synthetic ground truth also include non-rigid scenes [33]. Like most previous binocular RGB approaches [e.g. 47, 52], our technique computes scene flow between two consecutive time steps of stereo video. As opposed to the RGB-D domain, dense binocular RGB-only variational scene flow computation at true real-time frame rates of 30 Hz or more has not been shown so far. Wedel et al. [52] achieved 20 Hz for 320×240 resolution videos using an implementation with a GPU and an FPGA.

Algorithm 1 Variational Halfway Domain Scene Flow

```
($\mathcal{S}$, $\mathcal{O}$, $\mathcal{L}$) = Initialization();
for $i$ = 1 ... num_levels do
    $\mathcal{S}$ = Compute_Scene_Flow($\mathcal{S}$, $\mathcal{O}$, $\mathcal{L}$);
    $\mathcal{O}$ = Occlusion_Maps($\mathcal{S}$);
    $\mathcal{L}$ = Illumination_Maps($\mathcal{S}$);
    Prolongation($\mathcal{S}$, $\mathcal{O}$, $\mathcal{L}$);
end for
```

Scene flow estimation is also connected to non-rigid structure-from-motion [e.g. 1, 12, 20, 36, 42, 64], although these approaches often apply strong motion priors and work best for small motions. Another area of work related to ours is spatiotemporal stereo matching [28, 41, 61], which generally assumes static camera setups. As discussed in the introduction, scene flow is an essential ingredient for many applications, such as free-viewpoint video [31], facial performance capture [48, 53], and motion understanding [34, 52]. Our work lifts the major computational barrier of previous scene flow approaches by demonstrating the first technique for real-time dense variational scene flow estimation from two RGB videos.

3. Variational Halfway Domain Scene Flow

Given two synchronized input camera streams (this can be achieved in hardware or software [e.g. 15, 17, 21, 35]), the goal of our dynamic scene reconstruction approach is to compute the dense 3D geometry and its motion over time. In all our live experiments, we use a custom commodity stereo rig built using two Logitech HD Pro C920 webcams. The captured live streams are assumed to be synchronized.

Similar to Valgaerts et al. [47], we parametrize scene flow using three unique flow fields: the stereo, motion and difference flow field. We solve for the scene flow in a hierarchical coarse-to-fine fashion using a variational scene flow approach (see Algorithm 1). During optimization, we bootstrap the computation of occlusion maps by rendering a triangulated version of the scene. We also compute illumination correction maps based on the per-level results. These occlusion and illumination correction maps are computed after the optimization on a level is finished and are upsampled ('prolongated') to the next finer hierarchy level to constrain the energy. In the following, we provide more details on the used scene flow parameterization and how we check for flow validity. Details on the illumination and occlusion map computation are provided in Section 5.

3.1. Halfway Domain Scene Flow Geometry

We extend the idea of the halfway correspondence domain [30] to the context of scene flow, as illustrated in Figure 1. We consider two monochrome stereo image pairs $\mathcal{I}_c^t$, where $c \in \{0,1\}$ denotes the camera index (0: left, 1: right) and $t \in \{0,1\}$ denotes the time step (0: previous, 1: current). The four captured images define the corners of the scene flow geometry. The five in-between frames define intermediate states of warping between the captured images based on the flow data. We define the scene flow $\mathcal{S}$ as a combination of three flows (stereo, motion and difference) relative to the halfway domain in the middle. The two intermediate images to the left and right of the halfway domain can be thought of as being captured by virtual cameras at the halfway time step. The top and bottom intermediate images can be thought of as being captured by a virtual in-between camera. The pixels of the halfway domain (given by the integer pixel grid positions $\mathbf{x}_i \in \mathbb{N}^2$) can be mapped to the four input images by combining the per-pixel stereo $\{\mathbf{s}_i \in \mathbb{R}^2\}_{i=1}^N$ (blue), motion $\{\mathbf{m}_i \in \mathbb{R}^2\}_{i=1}^N$ (yellow) and difference flow $\{\mathbf{d}_i \in \mathbb{R}^2\}_{i=1}^N$ (red), where N is the number of pixels in the image. The direction of the arrows indicates the target space of the flow field. Arrows pointing from left to right and top to bottom represent positive signs, otherwise the sign is negative.

3.2. Binocular Scene Flow Consistency

We consider the flows between all combinations of input frame pairs consistent if every pixel of one input image is mapped to the corresponding pixel in all other input images that see the same 3D surface point. The consideration of all different mappings between the four input images gives rise to a total of six different consistency checks. Note that we only model the checks in forward direction for higher efficiency. Since checking for the same surface point is impossible, we at first relax the consistency condition to a brightness constancy check. The first two checks are the stereo flow consistency checks, which map between the two cameras of the stereo pairs at corresponding time steps:

$$d_0(\mathbf{x}_i) = \mathcal{I}_1^0(\mathbf{x}_i + \mathbf{s}_i - \mathbf{m}_i - \mathbf{d}_i) - \mathcal{I}_0^0(\mathbf{x}_i - \mathbf{s}_i - \mathbf{m}_i + \mathbf{d}_i), \quad (1)$$

$$d_1(\mathbf{x}_i) = \mathcal{I}_1^1(\mathbf{x}_i + \mathbf{s}_i + \mathbf{m}_i + \mathbf{d}_i) - \mathcal{I}_0^1(\mathbf{x}_i - \mathbf{s}_i + \mathbf{m}_i - \mathbf{d}_i). \quad (2)$$

The second pair of checks model motion flow consistency, which considers images captured by the same camera at consecutive time steps:

$$d_2(\mathbf{x}_i) = \mathcal{I}_0^1(\mathbf{x}_i - \mathbf{s}_i + \mathbf{m}_i - \mathbf{d}_i) - \mathcal{I}_0^0(\mathbf{x}_i - \mathbf{s}_i - \mathbf{m}_i + \mathbf{d}_i), \quad (3)$$

$$d_3(\mathbf{x}_i) = \mathcal{I}_1^1(\mathbf{x}_i + \mathbf{s}_i + \mathbf{m}_i + \mathbf{d}_i) - \mathcal{I}_1^0(\mathbf{x}_i + \mathbf{s}_i - \mathbf{m}_i - \mathbf{d}_i). \quad (4)$$

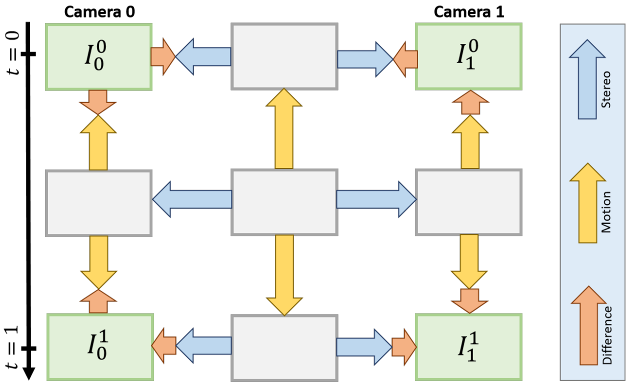

Figure 1. Binocular halfway-domain scene flow geometry. Note that arrows pointing from left to right and top to bottom represent flows with positive signs, otherwise the sign is negative.

Finally, the cross consistency checks consider the images captured by different cameras at different time steps:

$$d_4(\mathbf{x}_i) = \mathcal{I}_1^1(\mathbf{x}_i + \mathbf{s}_i + \mathbf{m}_i + \mathbf{d}_i) - \mathcal{I}_0^0(\mathbf{x}_i - \mathbf{s}_i - \mathbf{m}_i + \mathbf{d}_i), \quad (5)$$

$$d_5(\mathbf{x}_i) = \mathcal{I}_0^1(\mathbf{x}_i - \mathbf{s}_i + \mathbf{m}_i - \mathbf{d}_i) - \mathcal{I}_1^0(\mathbf{x}_i + \mathbf{s}_i - \mathbf{m}_i - \mathbf{d}_i). \quad (6)$$

Here we formulated consistency only in terms of brightness constancy. In the following, we will also consider gradient constancy to define a matching criterion that is more robust to appearance and lighting changes.

3.3. Scene Flow Parameterization

Different from previous approaches, we parameterize the per-pixel flow fields based on a uniform deformation lattice with a coarser resolution than the captured input images, to achieve real-time performance. This helps to resolve ambiguities in flow computation, since multiple input pixel observations influence the unknown displacement at each grid point. In addition, the introduction of this deformation proxy reduces the number of unknowns and thus leads to higher efficiency. In all our experiments, we used a coarsening factor of 2, meaning that we have a deformation grid point on every second pixel. Since we parameterize the per-pixel displacements based on a coarser resolution warp grid, we obtain in-between flow values via bilinear interpolation. For example, the per-pixel stereo flow can be obtained based on the G stereo flow deformation nodes $\{\mathbf{g}_k^s\}_{k=1}^G$ by:

$$\mathbf{s}_i = \sum_{k=1}^{G} \alpha_{i,k}^s \cdot \mathbf{g}_k^s. \quad (7)$$

Here, the $\alpha_{i,k}^s$ are the bilinear interpolation weights for the per-pixel stereo flow $\mathbf{s}_i$. Note that for a particular i, $\alpha_{i,k}^s$ defines a sparse partition of unity over the G grid points (only four k's have non-zero $\alpha_{i,k}^s$ for any i). Similar relations also hold for the motion and difference flow fields.

4. Spatiotemporal Scene Flow Objective

Similar to previous work [47], we cast finding the scene flow that best explains the binocular input images in two successive time steps as a variational energy minimization problem. The objective function takes into account both spatial alignment of the inputs warped to the halfway-domain reference frame and the validity of the flow fields. Therefore, our non-linear scene flow objective E is a mixture of spatial alignment E_{align} and regularization constraints E_{reg}:

$$E(\mathcal{S}) = E_{\text{align}}(\mathcal{S}) + w_{\text{reg}} E_{\text{reg}}(\mathcal{S}). \tag{8}$$

Here, we parameterize the unknown per-pixel scene flow based on a vector $\mathcal{S}$ that stacks the $3G$ unknown stereo $\{\mathbf{g}_k^s\}_{k=1}^G$, motion $\{\mathbf{g}_k^m\}_{k=1}^G$ and difference flow $\{\mathbf{g}_k^d\}_{k=1}^G$ control points. The regularization weight w_{reg} balances alignment accuracy with robustness against outliers due to noise and featureless regions. In the following, we provide details on the employed constraints.

Spatiotemporal Scene Flow Alignment The alignment objective E_{align} enforces that the two stereo pairs captured for two subsequent time steps align well in the halfway-domain reference frame. Since we consider both stereo and temporal motion constraints, this leads to the following spatiotemporal scene flow alignment constraint:

$$E_{\text{align}}(\mathcal{S}) = w_{\text{photo}} E_{\text{photo}}(\mathcal{S}) + w_{\text{grad}} E_{\text{grad}}(\mathcal{S}). \tag{9}$$

The quality of alignment of the warped observations in the input frame is quantified based on two terms that model photometric E_{photo} (Section 4.1) and gradient-domain E_{grad} (Section 4.2) alignment constraints, respectively. The weights w_{photo} and w_{grad} define the relative importance of these terms.

Spatial Regularization Constraints Recovering the unknown scene flow $\mathcal{S}$ from the two captured stereo pairs is a challenging problem due to noise in the image acquisition process and ambiguities due to featureless image regions. To allow for the robust estimation of high-quality scene flow despite these challenges, we propose an efficient regularization strategy based on three terms:

$$E_{\text{reg}}(\mathcal{S}) = w_{\text{smooth}} E_{\text{smooth}}(\mathcal{S}) + w_{\text{epi}} E_{\text{epi}}(\mathcal{S}) + w_{\text{mag}} E_{\text{mag}}(\mathcal{S}). \tag{10}$$

The first term E_{smooth} (Section 4.3) enforces the local smoothness of the estimated flow fields. This term allows to handle noisy input data and bridges the uncertainty created by missing or incorrect alignment constraints in featureless regions. The second term E_{epi} constrains the stereo flow to be consistent with the epipolar geometry of the binocular camera setup. Different to previous methods, such as Valgaerts et al. [47], we do not manually linearize this constraint, leading to a better approximation of the derivatives. Finally, the third term E_{mag} (Section 4.5) constrains the flows to a reasonable magnitude leading to higher robustness. The weights w_{smooth}, w_{epi} and w_{mag} influence the relative importance of the terms.

4.1. Photometric Alignment

We enforce the photometric alignment of the captured input images to the halfway-domain reference frame based on a brightness constancy constraint:

$$E_{\text{photo}}(\mathcal{S}) = \sum_{i=1}^{N} \sum_{k=0}^{5} V(\mathbf{x}_i) \cdot W(\mathbf{x}_i) \cdot \Phi\Big(d_k(\mathbf{x}_i)\Big). \tag{11}$$

The visibility map $V(\mathbf{x}_i)$ encodes the visibility of the associated 3D point (1: visible, 0: not visible). Different from many related methods, we take visibility into account based on computed per-pixel occlusion maps, which are bootstrapped based on a hierarchical optimization scheme (Section 5.3). The weight $W(\mathbf{x}_i)$ is used for pruning outliers based on color similarity: it is one if the residual per-pixel color distance in the reference frame is smaller than a threshold $\epsilon_{\text{p}} = 0.2$, and zero otherwise. The functions d_k are the flow consistency constraints in Section 3.2. Instead of a least-squares formulation, we use the robust pseudo-Huber penalty function for increased robustness against outlier correspondences:

$$\Phi(x) = \sqrt{x^2 + \epsilon^2}. \tag{12}$$

We use $\epsilon = 0.001$ in all our experiments.

4.2. Gradient Domain Alignment

In addition to the photometric alignment term, we also use a gradient domain alignment constraint in the reference frame:

$$E_{\text{grad}}(\mathcal{S}) = \sum_{i=1}^{N} \sum_{k=0}^{5} V(\mathbf{x}_i) \cdot W(\mathbf{x}_i) \cdot \Phi\Big(\|\nabla d_k(\mathbf{x}_i)\|^2\Big). \tag{13}$$

This gradient domain measure is more robust to differences in the response functions of the used cameras as well as temporal illumination changes than just a brightness constancy term would be. Φ again denotes the robust pseudo-Huber penalty function, V encodes visibility and W prunes outliers based on color dissimilarity.

4.3. Flow Field Smoothness

To increase the robustness against noise and featureless regions in the input images, we incorporate local smoothness of the three flow fields (motion, stereo and difference flow) by enforcing neighboring displacements to be similar:

$$E_{\text{smooth}}(\mathcal{S}) = \sum_{i=1}^{G} \sum_{j \in \mathcal{N}_i} \sum_{f \in \{\text{s,m,d}\}} w_i w_f \left\| \mathbf{g}_i^f - \mathbf{g}_j^f \right\|^2. \tag{14}$$

The three weights w_f (for $f \in \{\text{s}, \text{m}, \text{d}\}$) balance the smoothness of the stereo, motion and difference flow, respectively. The per-pixel weight w_i takes into account how discriminative a small 3×3 pixel region around the i^{th} grid point is and hence how well it can be tracked. For featureless regions, w_i is set to a high value, which strengthens regularization. We compute this weight by analyzing the two eigenvalues of the auto-correlation matrix of the image patches.

4.4. Epipolar Geometry Consistency Constraint

For increased robustness and to further constrain the flow fields, we enforce the stereo flow to be consistent with the epipolar geometry of the fixed stereo camera setup:

$$E_{\text{epi}}(\mathcal{S}) = \sum_{i=1}^{G} \left[(\mathbf{l}_i^0)^\top \mathbf{F}(\mathbf{r}_i^0)\right]^2 + \left[(\mathbf{l}_i^1)^\top \mathbf{F}(\mathbf{r}_i^1)\right]^2. \quad (15)$$

Here, $\mathbf{F}$ is the fundamental matrix and the 3D vectors $\mathbf{l}_i^t$ and $\mathbf{r}_i^t$ denote the homogeneous coordinates of reference grid point positions $\mathbf{g}_i^x$ transformed to the left and right camera, respectively. The transformed 2D reference positions are:

$$\hat{\mathbf{l}}_i^0 = \mathbf{g}_i^x - \mathbf{g}_i^s - \mathbf{g}_i^m + \mathbf{g}_i^d, \quad \hat{\mathbf{r}}_i^0 = \mathbf{g}_i^x + \mathbf{g}_i^s - \mathbf{g}_i^m - \mathbf{g}_i^d, \quad (16)$$

$$\hat{\mathbf{l}}_i^1 = \mathbf{g}_i^x - \mathbf{g}_i^s + \mathbf{g}_i^m - \mathbf{g}_i^d, \quad \hat{\mathbf{r}}_i^1 = \mathbf{g}_i^x + \mathbf{g}_i^s + \mathbf{g}_i^m + \mathbf{g}_i^d. \quad (17)$$

This constraint effectively enforces that corresponding pixels in the images are close to the corresponding epipolar line. Different to previous methods, e.g. Valgaerts et al. [47], we do not manually linearize this constraint and thus obtain a better approximation of the derivatives.

4.5. Flow Field Magnitude

We further stabilize the scene flow estimation by constraining the magnitude of the three flow fields. This Tikhonov regularization strategy is enforced based on the following soft-constraint:

$$E_{\text{mag}}(\mathcal{S}) = \sum_{i=1}^{G} \sum_{f \in \{\text{s,m,d}\}} m_f \left\| \mathbf{g}_i^f \right\|^2. \quad (18)$$

Since the stereo, motion and difference flows exhibit different properties, we use the weights m_f, $f \in \{s, m, d\}$, to balance these constraints. Due to temporal coherence, we assume the motion flow to be smaller than the stereo flow. The difference flow is assumed to be the smallest, since it only models the residual displacement. We introduce a hierarchical optimization strategy in Section 5.3 that still allows to handle large displacements on the coarser levels of the hierarchy.

4.6. Scene Flow Parameters

The choice of parameters influences our scene flow energy and the reconstruction results. Our approach proved quite robust to variation in the specific parameter values. Nevertheless, the best reconstruction results are obtained at the sweet spot between the data fitting term and the prior constraints. We provide the parameters used to generate the results in the supplemental document.

5. Data-Parallel Optimization

The number of unknowns of our non-linear scene flow objective $E(\mathcal{S})\colon \mathbb{R}^{6G} \to \mathbb{R}$ depends on the number of control points G of the used deformation grid (two unknowns for each node of the three different flows). Since this number directly depends on the image resolution and the grid step size (aka the coarsening factor), this leads to a large number of unknowns, even for smaller image resolutions, for example $(2 \times 3 \times 800 \times 600)/2^2 = 720$K unknowns for an image resolution of 800×600 pixels and a grid step size of 2 pixels. Since we aim to solve the scene flow problem at real-time frame rates, we devise a data-parallel hierarchical solver, following Zollhöfer et al. [66], that exploits the computational power of modern graphics cards. Our hierarchy encodes flows based on deltas to the next coarser level. This enables us to handle large displacements and allows for fast convergence based on a temporal propagation strategy.

We cast finding the scene flow $\mathcal{S}^*$ that best explains the input observations as a non-linear optimization problem:

$$\mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\, E(\mathcal{S}). \quad (19)$$

This is a general unconstrained optimization problem, since the alignment objective E_{align} (Equation 9) does not fit the canonical least-squares structure of the other objectives due to the robust pseudo-Huber penalty. Since it is challenging to devise real-time data-parallel solvers for such problems, we transform our problem to a non-linear least-squares problem by taking the square root of the residuals ($x \equiv (\sqrt{x})^2$).

5.1. Data-Parallel Gauss-Newton Solver

After this transformation, the optimization problem fulfills the canonical least-squares form, and can be written as a sum of squared residual terms r_m:

$$E(\mathcal{S}) = \sum_{m=1}^{M} r_m^2(\mathcal{S}). \quad (20)$$

We stack all M residuals into the residual vector operator $\mathbf{R}\colon \mathbb{R}^{6G} \to \mathbb{R}^M$ and rewrite the energy E using it:

$$E(\mathcal{S}) = \|\mathbf{R}(\mathcal{S})\|^2, \quad (21)$$

$$\mathbf{R}(\mathcal{S}) = \begin{bmatrix} r_1(\mathcal{S}) & \dots & r_M(\mathcal{S}) \end{bmatrix}^\top. \quad (22)$$

Our proposed objective function comprises $M = 2N + 14G$ residuals r_m due to the used photometric alignment (N), gradient-domain alignment (N), smoothness ($6G$), epipolar ($2G$) and flow magnitude ($6G$) constraints. Due to the large number of residuals (M) and unknowns ($6G$), a data-parallel optimization strategy is of paramount importance to achieve real-time frame rates. Since the residual vector $\mathbf{R}$ is still non-linear in the unknowns $\mathcal{S}$, Gauss-Newton explicitly linearizes $\mathbf{R}$ based on a first-order Taylor expansion:

$$\mathbf{R}(\mathcal{S}_{k+1}) \approx \mathbf{R}(\mathcal{S}_k) + \mathbf{J}(\mathcal{S}_k) \cdot \delta, \quad \delta = \mathcal{S}_{k+1} - \mathcal{S}_k. \quad (23)$$

Here, $\mathbf{J}(\mathcal{S}_k)$ is the Jacobian of $\mathbf{R}$ evaluated at the solution after k iterations. The Jacobian is computed based on analytical derivatives. The resulting least-squares problem to find optimal updates δ^* is:

$$\delta^* = \underset{\delta}{\operatorname{argmin}} \left\|\mathbf{R}(\mathcal{S}_k) + \mathbf{J}(\mathcal{S}_k) \cdot \delta\right\|^2. \tag{24}$$

The optimum is computed by solving the associated normal equations based on a data-parallel preconditioned conjugate gradient (PCG) [66] solver. Similar to Zollhöfer et al. [65] and Wu et al. [54], we also employ a domain decomposition strategy for higher performance, and a hierarchical optimization strategy to speed up convergence. However, in contrast, we employ a hierarchy of delta updates that allows for a better temporal initialization strategy and the computation of large displacements. This strategy also seamlessly integrates with our flow magnitude constraints E_{mag} (Section 4.5), enabling the computation of large flow displacements, since we only encourage the deltas to be small.

5.2. Domain Decomposition

We divide the problem into small subproblems based on a subdivision of the halfway-domain reference frame into small square subdomains of size 16×16 pixels (plus a boundary of 2 pixels). The optimization is then performed using multiple data-parallel *Alternating Schwarz* [63, 65] iterations. In each iteration, subproblems are locally solved based on one step of data-parallel Gauss-Newton (PCG for linear system), and the subdomain data exchange is handled via global memory. During PCG, all required data is kept in shared memory for increased performance.

In contrast to previous work [54, 65, 66], we precompute the non-zero entries of $\mathbf{J}^\top\mathbf{J}$ for the alignment term (9×3×2×2 per warp grid point) and read them on demand. This strategy is more efficient than evaluating them on the fly, since the computation of the system matrix for the alignment term is expensive due to the combinatorial explosion caused by every grid point depending on multiple pixels. Regularizers are still applied on-the-fly in each iteration step.

5.3. Delta Hierarchy for Fast Optimization

Our optimization strategy works in a coarse-to-fine manner, but in contrast to previous work [54, 65, 66], we use a hierarchy of delta flows (still with a downsampling factor of 2). This means that each level only stores and computes an offset with respect to the next coarser one. This helps fast convergence and the computation of large displacement flow fields. We flip-flop between solving and upsampling the results to the next finer level based on bilinear interpolation until the finest resolution level is reached. The number of levels used in our hierarchy depends on the resolution of the input images.

In the first frame, all flows are initialized to zero. In subsequent frames, we initialize the flow fields based on the results obtained in the previous time step. Based on the assumption of constant velocity, we use the computed motion flow to propagate all flow estimates from the previous to the next time step. Since we employ a delta hierarchy, we transfer the delta flows on each level separately.

We also use the hierarchy to bootstrap occlusion maps for visibility computation. To this end, we render the currently estimated geometry on every level from the camera views, and determine all visible pixels based on a z-buffer. The occlusion maps are interpolated to the next finer level and used to prune invisible pixels in the alignment term (Equation 9).

In a similar fashion, the illumination correction is applied on every hierarchy level. To this end, we compute the intensity residual between the two stereo pairs in the reference frame, and extract the low-frequency components by convolution with a Gaussian filter ($\sigma =$3.2 pixels). The extracted low-frequency components are attributed to illumination and/or differences in the cameras' response functions. We upsample the illumination differences using a box filter to the next finer level and use them to normalize the input images.

6. Results

We evaluate our approach on live data captured using a custom stereo webcam rig and also on publicly available datasets. The reconstructions based on our stereo rig are obtained at real-time frame rates. In addition, we apply our approach to the high-resolution, high-quality stereo data of Schneider et al. [44], and Blumenthal-Barby and Eisert [8]. Our approach scales well to this high-resolution data in terms of reconstruction quality and runtime performance. We also compare our approach to the slow, but high-quality, off-line scene flow approach of Valgaerts et al. [47]. Our approach obtains similar quality at much higher frame rate.

6.1. Live Results

We use two Logitech HD Pro C920 webcams to capture a stereo video stream at 1280×720 pixels (0.9 MP). The cameras' refresh rate is 30 Hz. Using our data-parallel solution strategy, we compute the scene flow at the refresh rate of the cameras. Figure 2 shows stereo reconstruction results and the corresponding scene flow obtained using our custom stereo webcam rig. As can be seen, we handle fully dynamic scenes and obtain detailed reconstructions.

6.2. Runtime Performance and Convergence

Figure 3 plots the runtime of our approach with respect to the resolution of the input images and different grid step sizes. As can been seen, the runtime of our approach scales linearly with the input image resolution. We obtain full real-time frame rates for up to 0.9 MP. For the resolution of our live setup (1280×720 pixels = 0.9 MP), we require 31 ms

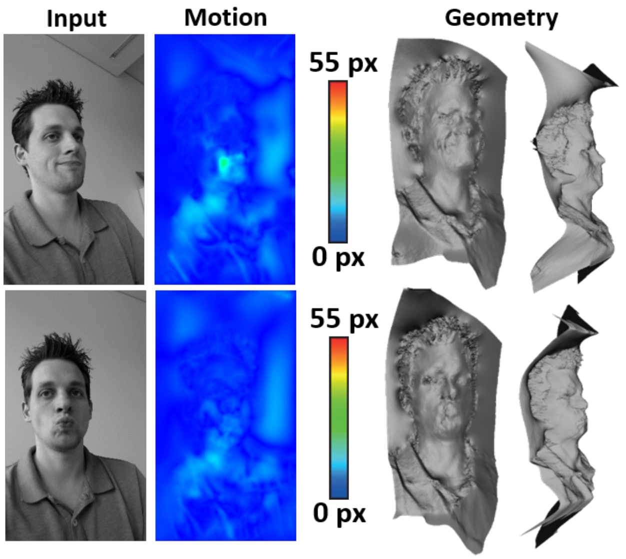

Figure 2. Live reconstruction results.

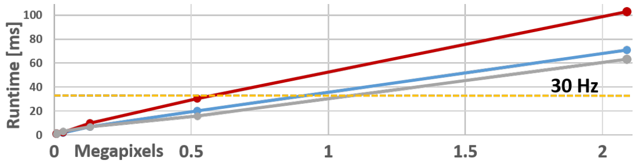

Figure 3. Runtime performance of our approach with different grid step sizes: red = 1, blue = 2, gray = 4.

to compute the scene flow. This high performance is a direct result of our data-parallel optimization strategy and our specifically tailored scene flow objective. For all timings, we used 5 hierarchy levels. On the two finest levels, we perform 2 non-linear iterations. On all other levels, we perform 5 non-linear iteration steps. In each non-linear Gauss-Newton step, we use 5 PCG iterations (with 5 patch iterations each) to solve the underlying system of normal equations.

We next analyze the convergence behavior of our solver for the finest hierarchy level. To this end, we reconstructed the scene flow between two time steps. In this evaluation, we apply 5 non-linear Gauss-Newton steps with 5 PCG steps (each using 5 patch iterations). Figure 4 plots the linear residual of the normal equations. For each single non-linear step, the error is always decreased by the PCG iteration steps. The error peaks every 5 steps, which marks the beginning of each new non-linear Gauss-Newton iteration. At these points, the problem is newly linearized using Taylor series expansion, leading to new normal equations. Therefore, the error of this new system is higher, but it is directly decreased in the following iterations. Note that these new systems are better approximations of the real function, since the linearization is performed closer to the optimum.

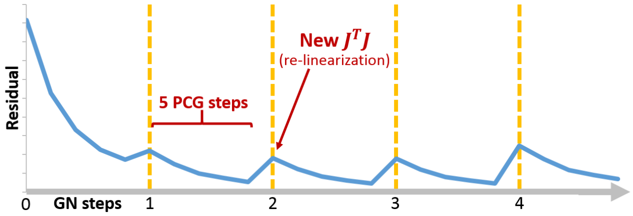

Figure 4. Convergence plot. See Section 6.2 for discussion.

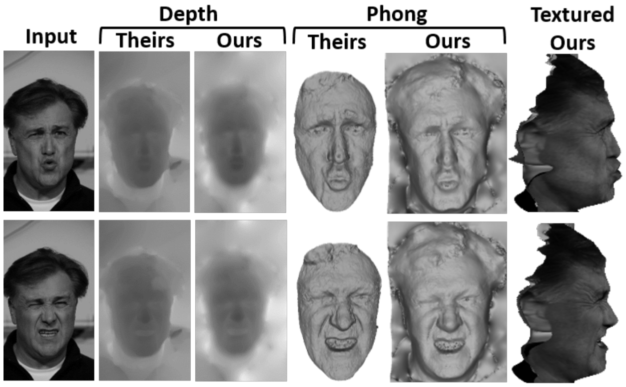

Figure 5. Comparison to the approach of Valgaerts et al. [47] on the 'Volker' dataset [48].

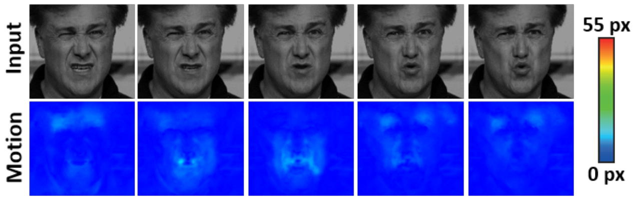

Figure 8. Our motion flow on the 'Volker' dataset [48].

6.3. Comparison to Valgaerts et al. [47]

We compare our approach to the slow, but high-quality offline state-of-the-art scene flow approach of Valgaerts et al. [47], see Figure 5. As can be seen, our approach obtains reconstructions of similar quality. Note that our approach is three orders of magnitudes faster than theirs. Due to the high resolution of the input data (1920×1088 pixels = 2.1 MP), we use 5 non-linear iterations with 5 PCG steps (each with 5 patch iterations). The employed hierarchy uses 5 levels, as before. With our approach, we obtain the scene flow for a single pair of frames in only 110 ms, while Valgaerts et al.'s method [47] requires more than 6 minutes per frame (more than 3,000 times slower). We attribute this performance advantage of our approach to the smart design of our objective function, including the reduction of unknowns through our coarser warp grid, that allows to apply our highly efficient data-parallel non-linear least-squares framework. The geometry and motion obtained by our approach is of very high quality, as shown in Figure 6 and Figure 8.

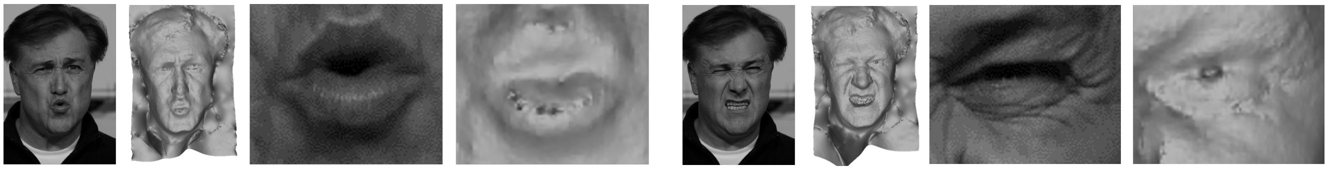

Figure 6. Our high-quality reconstruction results on the ‘Volker’ dataset [48].

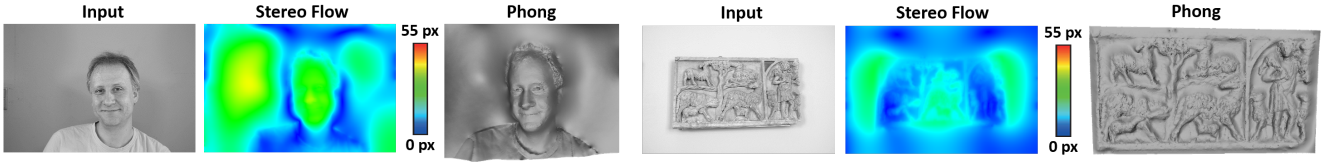

Figure 7. Our high-quality reconstruction results on the data of Schneider et al. [44] (left) and Blumenthal-Barby and Eisert [8] (right).

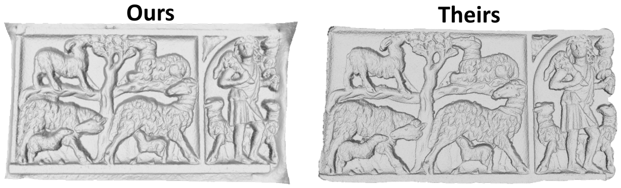

Figure 9. Comparison to Blumenthal-Barby and Eisert [8].

6.4. Comparison to Warping-based Approaches

We also applied our approach to the stereo reconstruction problem. To this end, we use publicly available high-quality stereo images [8, 44] with an input image resolution of 4288×2848 pixels (12 MP). Since only a single stereo pair is available for these scenes, we initialize both time steps using the same image pair. Our approach obtains high-quality reconstruction results, shown in Figure 7, which are on par with previous approaches, see also Figure 9. We obtain these results at a much shorter computation time. Our approach requires only 700 ms for reconstruction, and is two orders of magnitude faster than the approach of Blumenthal-Barby and Eisert [8], which requires several minutes. We attribute this difference in runtime performance to the design of our energy function and our data-parallel solution strategy. Note that if a sequence of stereo frames is available, our approach is also able to estimate the motion flow.

7. Limitations

Our approach obtains high-quality scene flow at real-time frame rates. Nevertheless, it is subject to a few limitations. We summarize them here and give ideas for future work in this domain. Sometimes our mip-map-based hierarchical downsampling strategy is too coarse. Therefore, distinctive regions are lost on coarser levels. This complicates the scene flow computation. A finer hierarchy in combination with feature-preserving downsampling [9] could alleviate this problem. Currently, our stereo setup has to be precalibrated before use. This is a cumbersome process and has to be repeated every time the camera setup changes. In the future, methods could be investigated to jointly optimize for the extrinsic camera parameters to allow for fully dynamic camera setups, similar to Valgaerts et al. [47]. Like every other passive stereo reconstruction approach, our approach suffers from problems in featureless regions of the scene. In these regions, the data term is not sufficiently discriminative and the used regularization terms take over. If the scene violates these prior assumptions, the obtained reconstructions do not match reality. Currently, in the first frame, we initialize the stereo and motion flow to zero. Therefore, our approach sometimes needs a few frames to converge. In the future, smart initialization strategies could be explored to jump-start the optimization process from the very first frame.

8. Conclusion

We presented an approach for real-time joint reconstruction of motion and geometry from stereo RGB videos. To this end, we extended the concept of the halfway domain to scene flow. Our approach achieves real-time performance based on a novel data-parallel solver that exploits the computational horsepower of modern graphics cards. Comparisons and evaluations show that high-quality scene flow estimates can be obtained at the cameras’ refresh rate using variational optimization. We believe that the availability of scene flow data at real-time frame rates is an important building block for many other approaches, such as real-time non-rigid structure-from-motion.

Acknowledgements

We thank Anna Hilsmann for the high-quality stereo pairs, and Levi Valgaerts for the high-quality stereo sequences. This research is funded by the ERC Starting Grant 335545 CapReal.

References

[1] S. Avidan and A. Shashua. Trajectory triangulation: 3D reconstruction of moving points from a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(4):348–357, April 2000. 2

[2] M. C. Bakkay and E. Zagrouba. Spatio-temporal filter for dense real-time scene flow estimation of dynamic environments using a moving RGB-D camera. *Pattern Recognition Letters*, 59:33–40, July 2015. 2

[3] S. T. Barnard and M. A. Fischler. Computational stereo. *ACM Computing Surveys*, 14(4):553–572, December 1982. 2

[4] J. T. Barron, A. Adams, Y. Shih, and C. Hernández. Fast bilateral-space stereo for synthetic defocus. In *CVPR*, 2015. 2

[5] T. Basha, S. Avidan, A. Hornung, and W. Matusik. Structure and motion from scene registration. In *CVPR*, 2012. 2

[6] T. Basha, Y. Moses, and N. Kiryati. Multi-view scene flow estimation: A view centered variational approach. *International Journal of Computer Vision*, 101(1):6–21, 2013. 2

[7] M. Bertozzi and A. Broggi. GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing*, 7(1):62–81, January 1998. 2

[8] D. C. Blumenthal-Barby and P. Eisert. High-resolution depth for binocular image-based modeling. *Computers & Graphics*, 39:89–100, 2014. 1, 2, 6, 8

[9] A. Bousseau, S. Paris, and F. Durand. User-assisted intrinsic images. *ACM Transactions on Graphics*, 28(5):130:1–10, December 2009. 8

[10] J. Čech, J. Sanchez-Riera, and R. Horaud. Scene flow estimation by growing correspondence seeds. In *CVPR*, 2011. 2

[11] C. B. Choy, J. Gawk, S. Savarese, and M. Chandraker. Universal correspondence network. In *NIPS*, 2016. 2

[12] Y. Dai, H. Li, and M. He. A simple prior-free method for non-rigid structure-from-motion factorization. *International Journal of Computer Vision*, 107(2):101–122, April 2014. 2

[13] D. DeTone, T. Malisiewicz, and A. Rabinovich. Deep image homography estimation. In *RSS Workshop on Limits and Potentials of Deep Learning in Robotics*, 2016. 2

[14] F. Devernay, D. Mateus, and M. Guilbert. Multi-camera scene flow by tracking 3-D points and surfels. In *CVPR*, 2006. 2

[15] A. Elhayek, C. Stoll, K. I. Kim, H.-P. Seidel, and C. Theobalt. Feature-based multi-video synchronization with subframe accuracy. In *Pattern Recognition*, 2012. 2

[16] D. Ferstl, G. Riegler, M. Rüther, and H. Bischof. CP-Census: A novel model for dense variational scene flow from RGB-D data. In *BMVC*, 2014. 2

[17] T. Gaspar, P. Oliveira, and P. Favaro. Synchronization of two independently moving cameras without feature correspondences. In *ECCV*, 2014. 2

[18] M. Gong, R. Yang, L. Wang, and M. Gong. A performance study on different cost aggregation approaches used in real-time stereo matching. *International Journal of Computer Vision*, 75(2):283–296, November 2007. 2

[19] S. Hadfield and R. Bowden. Scene particles: Unregularized particle-based scene flow estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):564–576, March 2014. 2

[20] R. Hartley and R. Vidal. Perspective nonrigid shape and motion recovery. In *ECCV*, 2008. 2

[21] N. Hasler, B. Rosenhahn, T. Thormählen, M. Wand, J. Gall, and H.-P. Seidel. Markerless motion capture with unsynchronized moving cameras. In *CVPR*, 2009. 2

[22] E. Herbst, X. Ren, and D. Fox. RGB-D flow: Dense 3-D motion estimation using color and depth. In *ICRA*, 2013. 2

[23] M. Hornáček, A. Fitzgibbon, and C. Rother. SphereFlow: 6 DoF scene flow from RGB-D pairs. In *CVPR*, 2014. 2

[24] F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *ICCV*, 2007. 2

[25] C. H. Hung, L. Xu, and J. Jia. Consistent binocular depth and scene flow with chained temporal profiles. *International Journal of Computer Vision*, 102(1-3):271–292, March 2013. 2

[26] M. Jaimez, M. Souiai, J. Gonzalez-Jimenez, and D. Cremers. A primal-dual framework for real-time dense RGB-D scene flow. In *ICRA*, 2015. 2

[27] M. Jaimez, M. Souiai, J. Stückler, J. Gonzalez-Jimenez, and D. Cremers. Motion cooperation: Smooth piece-wise rigid scene flow from RGB-D images. In *3DV*, 2015. 2

[28] H. Jiang, H. Liu, P. Tan, G. Zhang, and H. Bao. 3D reconstruction of dynamic scenes with multiple handheld cameras. In *ECCV*, 2012. 2

[29] A. Letouzey, B. Petit, and E. Boyer. Scene flow from depth and color images. In *BMVC*, 2011. 2

[30] J. Liao, R. S. Lima, D. Nehab, H. Hoppe, P. V. Sander, and J. Yu. Automating image morphing using structural similarity on a halfway domain. *ACM Transactions on Graphics*, 33(5):168:1–12, September 2014. 3

[31] C. Lipski, F. Klose, and M. Magnor. Correspondence and depth-image based rendering a hybrid approach for free-viewpoint video. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(6):942–951, June 2014. 1, 2

[32] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *CVPR*, 2016. 2

[33] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 2

[34] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *CVPR*, 2015. 1, 2

[35] B. Meyer, T. Stich, M. Magnor, and M. Pollefeys. Subframe temporal alignment of non-stationary cameras. In *BMVC*, 2008. 2

[36] H. S. Park, T. Shiratori, I. Matthews, and Y. Sheikh. 3D reconstruction of a moving point from a series of 2D projections. In *ECCV*, 2010. 2

[37] J.-P. Pons, R. Keriven, and O. Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision*, 72(2):179–193, April 2007. 2

[38] J. Quiroga, T. Brox, F. Devernay, and J. Crowley. Dense semi-rigid scene flow estimation from RGBD images. In *ECCV*, 2014. 2

[39] C. Rabe, T. Müller, A. Wedel, and U. Franke. Dense, robust, and accurate motion field estimation from stereo image sequences in real-time. In *ECCV*, 2010. 2

[40] C. Richardt, H. Kim, L. Valgaerts, and C. Theobalt. Dense wide-baseline scene flow from two handheld video cameras.

In *3DV*, 2016. 2

[41] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N. A. Dodgson. Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In *ECCV*, 2010. 2

[42] C. Russell, R. Yu, and L. Agapito. Video pop-up: Monocular 3D reconstruction of dynamic scenes. In *ECCV*, 2014. 2

[43] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1–3):7–42, April 2002. 2

[44] D. C. Schneider, M. Kettern, A. Hilsmann, and P. Eisert. Deformable image alignment as a source of stereo correspondences on portraits. In *CVPR Workshops*, 2011. 1, 2, 6, 8

[45] D. C. Schneider, M. Kettern, A. Hilsmann, and P. Eisert. A global optimization approach to high-detail reconstruction of the head. In *Vision, Modeling, and Visualization Workshop (VMV)*, 2011. 2

[46] D. Sun, E. B. Sudderth, and H. Pfister. Layered RGBD scene flow estimation. In *CVPR*, 2015. 2

[47] L. Valgaerts, A. Bruhn, H. Zimmer, J. Weickert, C. Stoll, and C. Theobalt. Joint estimation of motion, structure and geometry from stereo sequences. In *ECCV*, 2010. 1, 2, 4, 5, 6, 7, 8

[48] L. Valgaerts, C. Wu, A. Bruhn, H.-P. Seidel, and C. Theobalt. Lightweight binocular facial performance capture under uncontrolled lighting. *ACM Transactions on Graphics*, 31(6):187:1–11, November 2012. 1, 2, 7, 8

[49] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):475–480, March 2005. 1, 2

[50] C. Vogel, K. Schindler, and S. Roth. 3D scene flow estimation with a piecewise rigid scene model. *International Journal of Computer Vision*, 115(1):1–28, October 2015. 2

[51] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister. High-quality real-time stereo using adaptive cost aggregation and dynamic programming. In *3DPVT*, 2006. 2

[52] A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. Stereoscopic scene flow computation for 3D motion understanding. *International Journal of Computer Vision*, 95(1):29–51, October 2011. 1, 2

[53] C. Wu, C. Stoll, L. Valgaerts, and C. Theobalt. On-set performance capture of multiple actors with a stereo camera. *ACM Transactions on Graphics*, 32(6):161:1–11, November 2013. 1, 2

[54] C. Wu, M. Zollhöfer, M. Nießner, M. Stamminger, S. Izadi, and C. Theobalt. Real-time shading-based refinement for consumer depth cameras. *ACM Transactions on Graphics*, 33(6):200:1–10, November 2014. 6

[55] K. Yamaguchi, D. McAllester, and R. Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *ECCV*, 2014. 2

[56] Q. Yang, L. Wang, R. Yang, S. Wang, M. Liao, and D. Nistér. Real-time global stereo matching using hierarchical belief propagation. In *BMVC*, 2006. 2

[57] R. Yang and M. Pollefeys. Multi-resolution real-time stereo on commodity graphics hardware. In *CVPR*, 2003. 2

[58] A. Zanfir and C. Sminchisescu. Large displacement 3D scene flow with occlusion reasoning. In *ICCV*, 2015. 2

[59] J. Žbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17:1–32, 2016. 2

[60] C. Zhang, Z. Li, Y. Cheng, R. Cai, H. Chao, and Y. Rui. MeshStereo: A global stereo model with mesh alignment regularization for view interpolation. In *ICCV*, 2015. 2

[61] L. Zhang, B. Curless, and S. M. Seitz. Spacetime stereo: shape recovery for dynamic scenes. In *CVPR*, 2003. 2

[62] Y. Zhang and C. Kambhamettu. On 3-D scene flow and structure recovery from multiview image sequences. *IEEE Transactions on Systems, Man, and Cybernetics*, 33(4):592–606, August 2003. 2

[63] H.-K. Zhao. *Generalized Schwarz Alternating Procedure for Domain Decomposition*. PhD thesis, University of California, Los Angeles, 1996. 6

[64] E. Zheng, D. Ji, E. Dunn, and J.-M. Frahm. Sparse dynamic 3D reconstruction from unsynchronized videos. In *ICCV*, 2015. 2

[65] M. Zollhöfer, A. Dai, M. Innmann, C. Wu, M. Stamminger, C. Theobalt, and M. Nießner. Shading-based refinement on volumetric signed distance functions. *ACM Transactions on Graphics*, 34(4):96:1–14, July 2015. 6

[66] M. Zollhöfer, M. Nießner, S. Izadi, C. Rhemann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, and M. Stamminger. Real-time non-rigid reconstruction using an RGB-D camera. *ACM Transactions on Graphics*, 33(4):156:1–12, July 2014. 5, 6