Piecewise Rigid Scene Flow

Christoph Vogel Konrad Schindler
Photogrammetry & Remote Sensing, ETH Zurich

Stefan Roth
Department of Computer Science, TU Darmstadt

Abstract

Estimating dense 3D scene flow from stereo sequences remains a challenging task, despite much progress in both classical disparity and 2D optical flow estimation. To overcome the limitations of existing techniques, we introduce a novel model that represents the dynamic 3D scene by a collection of planar, rigidly moving, local segments. Scene flow estimation then amounts to jointly estimating the pixel-to-segment assignment, and the 3D position, normal vector, and rigid motion parameters of a plane for each segment. The proposed energy combines an occlusion-sensitive data term with appropriate shape, motion, and segmentation regularizers. Optimization proceeds in two stages: Starting from an initial superpixelization, we estimate the shape and motion parameters of all segments by assigning a proposal from a set of moving planes. Then the pixel-to-segment assignment is updated, while holding the shape and motion parameters of the moving planes fixed. We demonstrate the benefits of our model on different real-world image sets, including the challenging KITTI benchmark. We achieve leading performance levels, exceeding competing 3D scene flow methods, and even yielding better 2D motion estimates than all tested dedicated optical flow techniques.

1. Introduction

Scene flow estimation is the task of estimating dense 3D surface shape as well as a dense 3D motion field from two (or more) views of a scene taken at two (or more) time steps [20]. Applications include motion analysis and motion capture, driver assistance and autonomous navigation, and virtual or augmented reality. The 3D scene flow generalizes two classical problems of computer vision, dense stereo matching and dense optical flow estimation. Yet, despite significant progress in both stereo [4, 9, 26] and 2D optical flow estimation [5, 16, 17], existing 3D scene flow techniques [*e.g.*, 3, 10, 24] have remained quite limited in comparison. Perhaps surprisingly, the additional information available in stereo motion sequences has not been leveraged to the extent that 3D scene flow outperforms dedicated stereo or 2D optical flow techniques at their respective task.

Much like stereo or 2D motion estimation, scene flow estimation is ill-posed due to the 3D equivalent of the aperture problem, and thus requires prior assumptions on geometry and motion. Shortcomings of general-purpose regularization have prompted the development of stronger priors, *e.g.*, encouraging locally rigid motion [22] as is common to many scenes. This mirrors a general trend toward more expressive priors in stereo [*e.g.*, 4] and optical flow [12, 16].

We posit that existing 3D scene flow techniques have been limited by the underlying representation, and propose to model the scene as a collection of planar regions, each undergoing a rigid motion. Following prior work in stereo [4], we argue that most scenes of interest consist of regions with a consistent motion pattern, into which they can be segmented – at least implicitly – during scene flow estimation. Since a larger support is required to fit a plane and its rigid motion (9 unknowns) reliably, we base the initial estimation not on individual pixels, but on a superpixel segmentation of the reference image. From these segments we generate a large number of candidate *planes* in 3D object space, each with an associated rigid motion. Scene flow estimation is then cast as a labeling problem, which assigns each pixel to a segment and each segment to a rigidly moving 3D plane.

Although the superpixels significantly simplify and stabilize the inference, they lead to inaccuracies at flow boundaries, since the initial segmentation does not take into account depth or motion discontinuities. We address this by going back to the pixel level and updating the assignment of pixels to segments, thus removing artifacts due to the superpixel discretization. We also show how to explicitly include occlusion reasoning both at the segment and pixel level.

We make the following contributions: *(i)* We propose a novel 3D scene flow approach based on piecewise planar, rigidly moving regions, including regularization between these regions as well as explicit occlusion reasoning; *(ii)* we formulate an appropriate (discrete, non-submodular) energy toward inference in this model; and *(iii)* report scene flow estimates of hitherto unmatched accuracy. In experiments on challenging, realistic data the proposed approach substantially outperforms three state-of-the-art 3D scene flow methods. To the best of our knowledge, our method is moreover the first to realize the theoretical advantage afforded by the additional information from stereo sequences, and

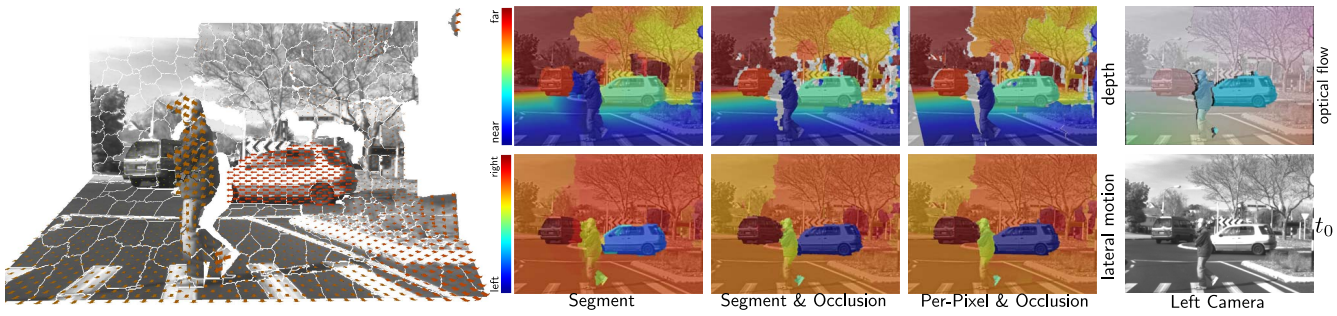

Figure 1. Example scene from [19]: *(left)* Jointly estimated 3D geometry, 3D motion vectors, and superpixel boundaries, rendered from a slightly different viewpoint. *(right)* Processing steps and final result of piecewise rigid scene flow estimation. Estimated depth, the lateral 3D motion component, and the re-projected 2D flow are shown. Occlusion areas are highlighted in white.

outperforms recent dedicated stereo and optical flow algorithms in challenging settings on their respective task.

2. Related Work

The term "scene flow" was coined by Vedula *et al*. [20], who were among the first, if not the first, to estimate both dense 3D geometry and a dense 3D motion field from multi-view image data. Estimation proceeds in two independent steps: First, 2D optical flow fields are estimated for all views (without requiring that they must be projections of the same 3D flow). Then a 3D flow field is fitted to them. Wedel *et al*. [24] proceed the other way around. Stereo disparity is precomputed for each time step; then the optical flow for a reference view and the disparity differences for the other view are estimated. Rabe *et al*. [13] integrate a Kalman filter into this approach to yield smooth flow fields over multiple frames. One of the limitations of these approaches is that a 2D regularizer is used, which encourages smooth projections, and not smooth 3D scene flow.

Huguet and Devernay [10] were possibly the first to estimate geometry and flow in an integrated manner with a variational formulation. Basha *et al*. [3] parameterize the scene flow by depth and a 3D motion vector w.r.t. a reference view, and estimate all parameters jointly with a 3D extension of the widely used optical flow method of Brox *et al*. [5]. This approach was modified by Vogel *et al*. [22], who argue that the total variation prior on the 3D motion field is biased for realistic baselines, and instead encourage locally rigid motion. The local rigidity assumption, which for sparse motion estimation dates back to at least Adiv [1], has also been used in 3D motion capture with explicit surface models [*e.g*., 6]. Also related is the optical flow approach of Nir *et al*. [12], in which the flow field is (over-) parameterized by explicitly searching for rigid motion parameters, and then encouraging their smoothness.

Valgaerts *et al*. [18] generalize the problem by assuming that only the camera intrinsics, but not the relative pose are known. In the presence of a dominant rigid motion ("background motion") they alternatingly estimate both the relative camera pose and the scene flow.

Common to these previous approaches to 3D scene flow is that they penalize deviations from spatial smoothness, typically in a robust way. In the context of stereo disparity and optical flow, explicit modeling of discontinuities by means of segmentation or layer-based formulations has a long history [23] and has recently gained renewed attention: Bleyer *et al*. [4] estimate disparity by assuming the scene to be segmented into planar superpixels and parameterizing their geometry. Segment-based stereo is also advocated by Yamaguchi *et al*. [26], who additionally penalize deviations from the (not segment-based) initialization. This method was further extended to epipolar flow, *i.e*. optical flow that enforces epipolar motion as hard constraint [27]. Sun *et al*. [16] estimate general 2D motion by decomposition into several layers, which enables occlusion reasoning. Unger *et al*. [17] compute optical flow by parameterizing the motion per segment with 2D affine transformations, and also perform occlusion handling. A key difference, aside from estimating 2D and not 3D motion, is that they do not consider any inter-patch regularization, such that the motion fields assigned to different segments are independent of each other. Discrete optimization based on fusion of proposals has been applied before to 2D optical flow estimation by Lempitsky *et al*. [11]. Here, such an optimization scheme is employed for 3D scene flow.

3. Piecewise Rigid Model for 3D Scene Flow

In contrast to typical approaches to 3D scene flow, our novel model parameterizes the scene as a collection of piecewise planar regions, each of which moves rigidly over time. As we show below, each region can be described using nine parameters, which are estimated by means of energy minimization. To that end we define an energy function that assigns each pixel to a segment and each segment to the 3D geometry and motion of a plane. This allows us to estimate the 3D scene flow and depth for every pixel of a reference view. The segmentation of the scene is only part of the internal representation and is not returned as an output.

We formulate our model for the classical case of two consecutive image pairs acquired with a calibrated stereo

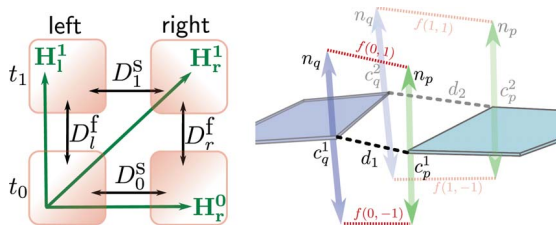

Figure 2. *(left)* Data terms in the two-view case; green: homographies. *(right)* Illustration of the regularization scheme.

rig: Irrespective of their true configuration the two views will be referred to as "left" and "right", denoted with subscripts l, r, while the two time steps will be denoted with superscripts $0, 1$. The scene is parameterized w.r.t. a reference frame I_l^0 (the "left" camera at time 0). For convenience of notation we assume w.o.l.g. that all (perspective) cameras have identical intrinsics $\mathbf{K}$. The reference camera hence has the projection matrix $(\mathbf{K}|\mathbf{0})$; the projection matrix of the "right" image at time 0 is written as $(\mathbf{M}|\mathbf{m})$.

Each moving 3D plane $\pi = \pi(\mathbf{R}, \mathbf{t}, \overline{\mathbf{n}})$ in the scene is described by nine parameters: A scaled normal vector $\overline{\mathbf{n}}$, a rotation matrix $\mathbf{R}$, and a translation vector $\mathbf{t}$. Note that since any plane visible in the reference image cannot contain the origin (camera center), we can denote a plane $\mathbf{n}^t\mathbf{x} = d$ with normal $\mathbf{n}$ and distance d to the origin by the scaled normal $\overline{\mathbf{n}} := \mathbf{n}/d$. Also note that we do not (need to) assume that the pixels belonging to each moving plane form a connected component. Parameterizing the scene with moving planes also conveniently allows the pixel locations assigned to each plane to be transformed easily between images or mapped into 3D space using the corresponding homographies.

The energy we define below is minimized in two steps: Starting from a set of superpixels, each segment is first labeled as belonging to one out of a large set of rigidly moving *proposal* planes, while keeping the pixel-to-segment assignment fixed; then, pixels are re-assigned to the best-fitting segment, while keeping the planar geometry and motion of each segment fixed. Note that while the model is based on segments, the aim here is *not* segmentation into semantic objects. Rather, the segments support accurate scene flow estimation. Over-segmentation is deliberately accepted, both to ensure correct depth and flow estimation for non-planar and articulated objects, and to maximize boundary recall, even at the cost of spurious segment boundaries that do not correspond to depth or motion discontinuities.

3.1. Model overview

Our aim is to estimate depth and a 3D scene flow vector for each pixel of the reference frame I_l^0. For now we assume that we have a finite set of possible *rigidly moving* proposal *planes* $\Pi = \{\pi_j\}$ in 3D. We then search for two mappings: A mapping $\mathcal{S} : I_l^0 \to S$, which assigns each pixel $\mathbf{p} \in I_l^0$ to a segment $s \in S$; and a mapping $\mathcal{P} : S \to \Pi$ to assign each segment to a rigidly moving 3D plane $\pi \in \Pi$.

We thus formulate scene flow estimation as minimizing

$$E(\mathcal{P}, \mathcal{S}) = E_D(\mathcal{P}, \mathcal{S}) + \lambda E_R(\mathcal{P}, \mathcal{S}) + \mu E_S(\mathcal{S}). \quad (1)$$

As is common in correspondence problems, a data term E_D ensures consistency of the corresponding appearance between the four views. The regularization term E_R encourages piecewise smooth geometry and motion, and a boundary term E_S additionally assesses the quality of the segmentation. We now define each of the three terms in detail.

3.2. Data term

The data term models the assumption that corresponding points across the four images should be similar in appearance. This amounts to a total of 4 constraints per pixel (two stereo constraints at time steps 0 and 1, and two optical flow constraints for the two left, respectively right images; see Fig. 2, left). Our representation with rigidly moving 3D planes induces homographies, which map pixels from the reference view I_l^0 to the remaining views:

$$\mathbf{H}_r^0(\pi) = (\mathbf{M} - \mathbf{m}\overline{\mathbf{n}}^t)\mathbf{K}^{-1} \quad (2a)$$

$$\mathbf{H}_l^1(\pi) = \mathbf{K}(\mathbf{R} - \mathbf{t}\overline{\mathbf{n}}^t)\mathbf{K}^{-1} \quad (2b)$$

$$\mathbf{H}_r^1(\pi) = (\mathbf{MR} - \big(\mathbf{Mt} + \mathbf{m}\big)\overline{\mathbf{n}}^t\big)\mathbf{K}^{-1} \quad (2c)$$

For convenience, we define $\mathbf{H}_l^0(\pi)$ to be the identity that maps the reference view onto itself, and denote the moving 3D plane of a pixel $\mathbf{p}$ as $\pi_\mathbf{p} = \mathcal{P}\left(\mathcal{S}(\mathbf{p})\right)$. We can thus define optical flow-induced appearance constraints across time as

$$D_i^\mathrm{f} = \sum_{\mathbf{p} \in I_l^0} \rho\big(\mathbf{H}_i^0(\pi_\mathbf{p})\mathbf{p}, \mathbf{H}_i^1(\pi_\mathbf{p})\mathbf{p}\big), \quad i \in \{l, r\}\ , \quad (3)$$

and stereo constraints between the simultaneous views as

$$D_t^\mathrm{s} = \sum_{\mathbf{p} \in I_l^0} \rho\big(\mathbf{H}_l^t(\pi_\mathbf{p})\mathbf{p}, \mathbf{H}_r^t(\pi_\mathbf{p})\mathbf{p}\big), \quad t \in \{0, 1\}\ . \quad (4)$$

The function $\rho(\cdot, \cdot)$ may simply encourage brightness constancy by penalizing brightness changes, *e.g.* through a robust, truncated penalty. Alternative choices include the more robust census transform [28]. The complete data term is given as the sum of the four terms in Eqs. (3) and (4):

$$E_D(\mathcal{P}, \mathcal{S}) = D_0^\mathrm{s} + D_1^\mathrm{s} + D_l^\mathrm{f} + D_r^\mathrm{f}. \quad (5)$$

3.3. Shape and motion regularization

The regularization terms shall encourage piecewise smooth 3D shape, as well as a piecewise smooth 3D motion field. Since each segment is assigned to one rigidly moving plane, smoothness within a segment is always satisfied; we thus only need to consider the segment boundaries. Assume that $\mathbf{p}$ and $\mathbf{q}$ are two adjacent pixels that are assigned to different moving planes $\pi_\mathbf{p} = \mathcal{P}(\mathcal{S}(\mathbf{p}))$ and $\pi_\mathbf{q} = \mathcal{P}(\mathcal{S}(\mathbf{q}))$.

Let us begin with the shape. To assess the regularity of the boundary between two pixels, we consider them to be

tiny square patches of equal area. We further assume that the boundary shared by the two pixels in image space has the (2D) endpoints c^1 and c^2. If we now project each endpoint onto each of the two 3D planes, we obtain the 3D endpoints $\mathbf{c}^1_\mathbf{p}$, $\mathbf{c}^1_\mathbf{q}$, $\mathbf{c}^2_\mathbf{p}$ and $\mathbf{c}^2_\mathbf{q}$ (see Fig. 2, right). Since we have chosen $\mathbf{p}$ and $\mathbf{q}$ to lie on different planes, the pixel boundaries will in general not coincide in 3D space, and our goal is to penalize the boundary distances. To that end we first denote the vectors between the 3D endpoints as $\mathbf{d}_1 = \mathbf{c}^1_\mathbf{p} - \mathbf{c}^1_\mathbf{q}$ and $\mathbf{d}_2 = \mathbf{c}^2_\mathbf{p} - \mathbf{c}^2_\mathbf{q}$. Since we are using planes as primitives, the 3D distance along the boundary is a convex combination of the endpoint distances $\alpha||\mathbf{d}_1|| + (1-\alpha)||\mathbf{d}_2||$. In order to take into account surface curvature as well, we not only evaluate the distance of the endpoints themselves, but also the distance after shifting the endpoints along the respective normals $\mathbf{n}_\mathbf{p}, \mathbf{n}_\mathbf{q}$. Denoting the difference of the normals as $\mathbf{d}_n = \mathbf{n}_\mathbf{p} - \mathbf{n}_\mathbf{q}$, we define a distance function $f_\gamma(\alpha,\beta) = ||\alpha(\mathbf{d}_1 + \gamma\beta\mathbf{d}_n) + (1-\alpha)(\mathbf{d}_2 + \gamma\beta\mathbf{d}_n)||$ (see Fig. 2). The weight γ balances plain boundary distance *vs.* curvature. The shape regularizer is then defined as the integral of $(f_\gamma)^2$ along the boundary (w.r.t. α) and along the normal direction (w.r.t. β), which yields a closed form:

$$E^1_R(\mathcal{P},\mathcal{S}) = \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} w_{\mathbf{p},\mathbf{q}}\psi\left(3\int_0^1\int_{-1}^1 f_\gamma(\alpha,\beta)^2 \mathrm{d}\beta\mathrm{d}\alpha\right) \quad (6)$$

$$= \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} w_{\mathbf{p},\mathbf{q}}\psi\left(||\mathbf{d}_1||^2 + ||\mathbf{d}_2||^2 + \langle\mathbf{d}_1,\mathbf{d}_2\rangle + \gamma^2||\mathbf{d}_n||^2\right).$$

Here, $\mathcal{N}$ are all neighboring pixels of the reference image (8-neighborhood), the length of the edge between the pixels is given by $w_{\mathbf{p},\mathbf{q}}$, and $\psi(\cdot)$ denotes a penalty function. Note that we can sum over all neighboring pixels, since for adjacent pixels on the same segment $f_\gamma \equiv 0$. The arbitrary scaling factor 3 is used for mathematical convenience.

The motion field is regularized in a similar manner, by integrating over the distances between adjacent motion vectors using $\mathbf{d}^m_i = \mathbf{R}_\mathbf{p}\mathbf{c}^i_\mathbf{p} + \mathbf{t}_\mathbf{p} - \mathbf{c}^i_\mathbf{p} - (\mathbf{R}_\mathbf{q}\mathbf{c}^i_\mathbf{q} + \mathbf{t}_\mathbf{q} - \mathbf{c}^i_\mathbf{q})$, as well as the differences between the (rotated) normals $\mathbf{d}^m_n = (\mathbf{R}_\mathbf{p}\mathbf{n}_\mathbf{p} - \mathbf{n}_\mathbf{p}) - (\mathbf{R}_\mathbf{q}\mathbf{n}_\mathbf{q} - \mathbf{n}_\mathbf{q})$, which leads to

$$E^2_R(\mathcal{P},\mathcal{S}) = \quad (7)$$

$$\sum_{(\mathbf{p},\mathbf{q})\in\mathcal{N}} w_{\mathbf{p},\mathbf{q}}\psi\left(||\mathbf{d}^m_1||^2 + ||\mathbf{d}^m_2||^2 + \langle\mathbf{d}^m_1,\mathbf{d}^m_2\rangle + \gamma^2||\mathbf{d}^m_n||^2\right).$$

The regularizer $E_R(\mathcal{P},\mathcal{S})$ is given as the sum of the shape and motion regularizers. To yield the necessary robustness against occasional discontinuities, we use truncated penalties $\psi(y) = \min(\sqrt{y}, \eta)$ (with thresholds η_1, η_2).

Note that the proposed regularizer is not restricted to 3D, since one is free to replace the distances. If 2D regularization in the image plane is desired instead, one can mimic the regularizer of [18] by replacing the 3D distances with disparity differences, differences between 2D optical flow vectors, and changes of the disparity difference over time.

3.4. Segmentation regularization

As the data term and the previous regularization term consider not only the motion and geometry of the moving plane assigned to each segment, but also which segment each pixel is being assigned to, it is necessary to regularize the segmentation further to encourage spatially coherent (though not necessarily compact) segments. We employ a segment regularization term similar in spirit to the approach of [21], which encourages smooth segments whose boundaries coincide with image edges. The energy is defined as

$$E_S(\mathcal{S}) = \sum_{\substack{(\mathbf{p},\mathbf{q})\in\mathcal{N},\\ \mathcal{S}(\mathbf{p})\neq\mathcal{S}(\mathbf{q})}} \exp\left(\tfrac{-a|I^0_l(\mathbf{p}) - I^0_l(\mathbf{q})|}{\sigma_I(\mathbf{p},\mathbf{q}) + \epsilon}\right) \quad (8)$$

$$+ \sum_{\mathbf{p}\in I^0_l} \begin{cases} 0, & \exists\,\mathbf{e}\in\mathcal{E}(s_i) : ||\mathbf{e}-\mathbf{p}||_\infty < N_S \\ \infty, & \text{else.} \end{cases}$$

The first term is a contrast-sensitive pairwise Potts model, which penalizes segment transitions such that transitions that coincide with large image gradients are penalized less. The standard deviation $\sigma_I(\mathbf{p},\mathbf{q})$ is estimated from a 50×50 window around $\mathbf{p},\mathbf{q}$; we set $a = 10$, $\epsilon = 0.01$. As above, $\mathcal{N}$ denotes the 8-neighborhood of each pixel. The second term ensures that each pixel can only be assigned to those segments s_i, for which a seed point $\mathbf{e}\in\mathcal{E}(s_i)$ is less than N_S pixels away (w.r.t. the ℓ_∞ distance); we set $N_S = 25$. The motivation is twofold: First, this prevents segments from getting too large, such that the scene is not overly simplified by the assumed piecewise planarity and piecewise rigid motion. Second, since only a subset of possible segment assignments needs to be considered at each pixel during optimization, a significant speedup is achieved. The set of seed points $\mathcal{E}(s_i)$ for a segment contains the center pixel of the segment s_i in the initial superpixelization, as well as all pixels from a regularly-spaced grid (with spacing N_S) that fell into the respective initial superpixel.

3.5. Approximate inference

We perform inference in our model by approximately minimizing the energy from Eq. (1) w.r.t. the segmentation $\mathcal{S}$ and the rigid motion of each segment, represented as the mapping $\mathcal{P}$. To bootstrap this process, we obtain an initial segmentation $\mathcal{S}$ through a superpixelization of the reference image. To that end we first minimize the segmentation energy E_S from Eq. (8) alone, which amounts to the superpixelization approach of [21]. Seed points $\mathcal{E}$ on a regular grid ensure a sufficiently fine tiling. Strongly non-convex segments are split into (near-)convex pieces.

Next, we update depth and motion by approximately minimizing the energy w.r.t. the segment-to-plane map $\mathcal{P}$, assuming for now that the segmentation $\mathcal{S}$ is fixed to the superpixelization. Since the segmentation regularization E_S does not depend on $\mathcal{P}$, we only need to consider the data

term E_D and the shape and motion regularization $E_R^{1,2}$. Recall that these regularizers can only incur penalties at segment boundaries, since neighboring pixels within a segment will be assigned to the same moving plane, thus incur no regularization penalty. For efficiency, we simplify the energy by computing the penalty $\psi(\cdot)$ in Eqs. (6) and (7) from the endpoints of a segment boundary. Since the length of the actual boundary can be precomputed, this is much more efficient than computing the penalty for each boundary pixel. This approximation is reasonable, since the superpixel segments are near-convex. The optimization is performed over a set of proposal planes with their associated motion (see Sec. 3.6) using fusion moves [11] and QPBO [15].

Finally, we update the segmentation $\mathcal{S}$ assuming a fixed $\mathcal{P}$. Note that the preceding simplification cannot be applied in this case, since the segmentation itself is updated. We thus minimize Eq. (1) directly. This is still reasonably efficient, because the region constraint from Eq. (8) ensures that pixels can only be assigned to a certain segment if they lie within N_S pixels from one of its seed points. Consequently, optimization involves α-expansion (with QPBO) in a local graph of at most $(2N_S-1)^2$ nodes. Although the energy is not submodular, unlabeled nodes rarely occur.

3.6. Proposal generation

To perform inference over the depth and motion of each segment, we require a comprehensive proposal set of 3D planes along with their rigid motions. We can generate these from the output of other scene flow algorithms or by combining the results of stereo and optical flow algorithms (see Sec. 4.2). To that end we fit a 3D plane to each superpixel segment, and estimate its rigid motion from the flow field(s). In either case, fitting must be robust to a potentially large amount of outliers, caused both by inaccurate depth and motion estimates and by superpixels not being aligned with surface or motion boundaries. We address this by robustly minimizing the transfer error: We first generate an initial solution by minimizing the quadratic transfer error using efficient algebraic methods, and then refine the rigidly moving proposal planes by gradient descent on the robust transfer error (Lorentzian penalty). Each locally fitted proposal is considered valid for the closest ≈ 100 segments in its spatial vicinity. Thus, fusion moves can be made efficient using a partial (local) instantiation of the graph.

3.7. Occlusion handling

The data term as defined in Eq. (5) does not contain any form of occlusion handling; every pixel is always assumed visible. Since our scene representation is defined in 3D, it allows for explicit occlusion reasoning. This is particularly interesting in case of scene flow, since we have four views of the scene (2 cameras at 2 time steps). Hence, even if a pixel is occluded in a subset of the views, there may still be a view pair where no occlusion takes place.

We apply occlusion handling to all view pairs for which we formulate a data term (*c.f.* Eq. (5)); for the remainder we assume that we consider one of the view pairs with its data term, D. We apply the well-known principle of using a fixed occlusion penalty θ, if a certain pixel $\mathbf{p}$ (in the reference view) is occluded in at least one view of a pair. Since in case of an occlusion, the corresponding pixel locations are not related in their appearance, we do not apply the usual data penalty from Eqs. (3) or (4). Note that we only have to consider occlusions between pixels in different segments, since a visible 3D plane cannot occlude itself.

To ease understanding, we describe occlusion reasoning directly during a fusion/expansion move of the inference procedure (Sec. 3.5), *i.e.* we solve a binary optimization problem. Assume for now that the segment-to-plane mapping $\mathcal{P}$ is fixed, and we update the per-pixel segmentation $\mathcal{S}$. We address updating the segment-to-plane mapping later. Let $x_\mathbf{p}=0$ denote that a pixel $\mathbf{p}$ remains in its current segment, and $x_\mathbf{p}=1$ indicate that $\mathbf{p}$ will be switched to the candidate segment. We can thus rewrite the data term without occlusion reasoning (Sec. 3.2) as the Boolean function

$$D(\mathbf{x}) = \sum\nolimits_{\mathbf{p}\in I_l^0} \Big(u_\mathbf{p}^0 (1-x_\mathbf{p}) + u_\mathbf{p}^1 x_\mathbf{p} \Big), \qquad (9)$$

where $\mathbf{x}$ is the binary vector of all pixel assignments, $u_\mathbf{p}^0$ is the data penalty for $\mathbf{p}$ being in its current segment, and $u_\mathbf{p}^1$ the data penalty for switching $\mathbf{p}$ to the candidate segment.

We now consider whether a pixel $\mathbf{p}$ is occluded or not, which depends both on its binary segment assignment $x_\mathbf{p}$, and on whether there is any other pixel $\mathbf{q}$ (or possibly multiple pixels) that occludes $\mathbf{p}$. Determining whether $\mathbf{q}$ leads to an occlusion in turn depends on its segment assignment $x_\mathbf{q}$. We call $\mathcal{O}_\mathbf{p}^i$ the set of all pixel-assignment pairs $(\mathbf{q}, j)$ for which pixel $\mathbf{q}$ occludes pixel $\mathbf{p}$ if $x_\mathbf{p}=i$ and $x_\mathbf{q}=j$. Then we replace Eq. (9) with the occlusion-sensitive data term

$$D_\mathrm{O}(\mathbf{x}) = \sum_{\mathbf{p}\in I_l^0} \Big(\theta + \sum_{i=0}^{1} \hat{u}_\mathbf{p}^i [x_\mathbf{p}=i] \prod_{(\mathbf{q},j)\in\mathcal{O}_\mathbf{p}^i} [x_\mathbf{q}\neq j] \Big) \qquad (10)$$

Here, $\hat{u}_\mathbf{p}^i = u_\mathbf{p}^i - \theta$ is the difference of the (unoccluded) data penalty and the occlusion cost θ, and $[\cdot]$ denotes the Iverson bracket. To understand this, let us consider a single pixel $\mathbf{p}$. The summand becomes $\hat{u}_\mathbf{p}^0$, if $x_\mathbf{p}=0$ *and* the product equals 1, which is the case if there is no pixel that could possibly occlude $\mathbf{p}$, or if all possibly occluding pixels $\mathbf{q}$ are assigned a segment $x_\mathbf{q}$ in which they do not lead to an occlusion. The data cost for a pixel $\mathbf{p}$ thus equals θ in case of an occlusion, and otherwise the standard data penalty $u_\mathbf{p}^i$.

We detect potential occlusions using z-buffering. The per-pixel penalty may be a higher-order pseudo-Boolean function ($|\mathcal{O}_\mathbf{p}^i|>1$), depending on the number of possibly occluding pixels. To facilitate applying QPBO, we reduce

these higher-order terms to quadratic ones by introducing auxiliary variables [2].

To update the segment-to-plane mapping $\mathcal{P}$ given a fixed segmentation $\mathcal{S}$, we perform inference at the segment level for efficiency (see Sec. 3.5), *i.e*. by computing penalties for entire segments. The Boolean functions for this case are the same as in Eq. (10), but with the variables $x_{\mathbf{p}}, x_{\mathbf{q}}$ representing segments rather than pixels. A segment is considered occluded if its center is occluded.

4. Experiments

To give an impression of what the proposed model is capable of on realistic data, we first report qualitative results on a street scene from [19], which is recorded from a vehicle as it approaches a roundabout with multiple moving traffic participants. Both the independent object motion, but also complex occlusion patterns pose difficult challenges. Fig. 1 shows the estimated 3D scene flow (*left*), and the results after various processing stages *(right)*. The segment-based scene reconstruction (*Segment*) without occlusion handling already gives fairly plausible results in unoccluded areas, but assigns incorrect depth and motion to regions not visible in the reference image (best seen immediately left of the pedestrian). By adding the segment-based occlusion model (*Segment & Occlusion*), the occlusion regions are properly detected and their motion is extrapolated in a more realistic manner. Finally, the per-pixel refinement (*Per-Pixel & Occlusion*) visibly improves object and occlusion boundaries. To ensure the necessary robustness, we use aggressive truncation values $\eta_1 = \eta_2 = 1$ for the robust penalty, assuming that depth and motion changes exceeding 1 world unit are due to discontinuities. We use $\rho(a, b) = \min(|a-b|, \zeta)$, *i.e*. brightness constancy truncated at $\zeta = 10\%$ of the intensity range, and set $\lambda = 0.1, \mu = 0.1, \theta = 0.03, \gamma = 1$.

4.1. Comparison with 2D scene flow

For comparison with other scene flow methods from the literature [10, 18, 24] we consider the synthetic scene of [10], which consists of two independently rotating hemispheres in front of a plane. Our method performs slightly better than competing ones (Table 1), even though the dataset does not conform to our assumptions: The spheres are not well approximated well by planar segments, and the texture gradients do not coincide with depth/motion boundaries, so the initial over-segmentation is rather arbitrary.

4.2. KITTI dataset

For quantitative evaluation we test our method on two-frame stereo pairs from the KITTI dataset [8]. The dataset provides images (1240×376 pixels) recorded with a calibrated stereo rig on a car, for benchmarking of optical flow and stereo algorithms in the context of automotive applications. Semi-dense ground truth data was acquired by a laser scanner attached to the car. Having stereo pairs from consecutive video frames, the dataset also fulfills the requirements for scene flow estimation (see Fig. 3 for examples). We use the training portion of the dataset for detailed quantitative analysis, and the test portion (non-public ground truth) to compare to the state of the art.

	[18]	[10]	[24]	Ours
RMSE 2D Flow	**0.63**	0.69	0.77	**0.63**
RMSE Disparity	3.8	3.8	10.9	**2.84**
RMSE Scene Flow	1.76	2.51	2.55	**1.73**

Table 1. 2D errors for the *"sphere"* sequence [10].

The KITTI dataset provides a very challenging testbed for today's stereo, optical flow and scene flow algorithms: First, pixel displacements in the data set are large in general, exceeding 150 pixels for stereo and 250 pixels for optical flow. Second, the images exhibit strongly varying lighting conditions and many non-Lambertian surfaces, especially translucent windows and specular glass and metal surfaces. Third, the high speed of the forward motion creates large regions on the image boundaries that move out of the field of view between frames, such that no correspondence can be established. To identify these areas one might use the vehicles' ego-motion, which is not available, however.

Appearance modeling. We address the challenging lighting conditions using the census transform [28] over a 7×7 neighborhood to measure the data fidelity ρ, which has been shown to cope well with complex outdoor lighting [14]. In detail, we scale the Hamming distances by $1/24$ for the census data term, see [28], and set $\lambda = 10\mu, \gamma = 1$, $\kappa = 1.05$. The parameters are fixed for all image sets.

Visibility. To cope with areas that are *out of bounds*, *i.e*. not visible in all four images, we let the stereo and 2D flow algorithms from the proposal generator predict which pixels are out-of-bounds and encourage the scene flow estimate to stay near that prediction. Let V_l^1, V_r^0 and V_r^1 be the predicted binary visibility masks for all but the reference image (*out-of-bounds*: 0, pixel visible: 1), and further let $\Gamma_i^j[\cdot]$ be a binary function that determines whether its argument lies within the boundaries of image I_i^j. To penalize deviations from the predicted visibility, we add an energy term

$$E_{\mathrm{V}}(\mathcal{P}, \mathcal{S}) = \kappa \sum_{\mathbf{p} \in I_l^0} \left| V_r^0(\mathbf{p}) - \Gamma_r^0 \left[\mathbf{H}_r^0(\pi_{\mathbf{p}}) \mathbf{p} \right] \right| + \tag{11}$$
$$\left| V_l^1(\mathbf{p}) - \Gamma_l^1 \left[\mathbf{H}_l^1(\pi_{\mathbf{p}}) \mathbf{p} \right] \right| + \left| V_r^1(\mathbf{p}) - \Gamma_r^1 \left[\mathbf{H}_r^1(\pi_{\mathbf{p}}) \mathbf{p} \right] \right| .$$

Evaluation. In Table 2 we compare the average errors on the KITTI training set. Because of the uncertainty in the LiDAR data, the ground truth is not directly suited to assess sub-pixel accuracy. Therefore, the recommended error metric is to threshold the deviations from ground truth with a range of inlier/outlier thresholds, Z, and count the fraction of outliers for each Z. Note that the benchmark is biased towards methods that focus on the dominant background, be-

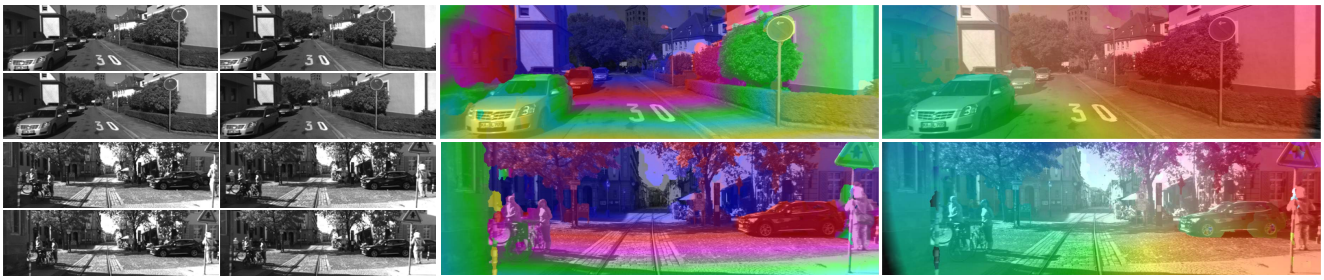

Figure 3. Example results from the KITTI benchmark. *(left)* input images; *(middle)* disparity w.r.t. reference frame; *(right)* flow reprojected to the reference image. We use the color scheme of the benchmark. See text for details.

cause independently moving objects usually have no ground truth points. Nonetheless, we believe that this dataset is better suited for scene flow evaluation than the very limited, synthetic datasets used before [*e.g.*, 10, 22]. Following the KITTI protocol, we distinguish between an evaluation on all pixels (*All*) and only on unoccluded pixels (*Noc*), and report results for error thresholds of 2, 3, 4 and 5 pixels.

As baselines we use our implementations of three methods: L_1-regularized 3D scene flow (*LSF*, [3]); locally rigid 3D scene flow (*Rig*, [22]); and independently derived 2D stereo (semi-global matching [9]) and optical flow (census data term, total generalized variation [25] regularization), indicated by (*S+F*). At time of writing the results of the 2D baseline rank 13^{th} in both stereo and optical flow among published methods in the official KITTI ranking. Our baseline implementations are on par with the published benchmark results, suggesting that they match the state of the art.

We quantitatively evaluate the individual steps of our approach, as well as different variants: First, we report results only based on superpixels (*PRSSeg*) and after per-pixel refinement (*PRSPix*). Second, we compare regularization in the image (*-2D*, with $\eta_1 = \eta_2 = 20$ and $\mu = 1/30$) and 3D regularization (*-3D*, with $\eta_1 = \eta_2 = 2.5$ and $\mu = 0.2$). We use less aggressive truncation thresholds here, since the lighting conditions are more challenging than in Fig. 1. Third, we include results with occlusion modeling (*-O*) and without. Finally, we distinguish the use of various proposal sets: All experiments default to proposals from the stereo and flow-derived $2D$ baseline technique (*S+F*). The suffix (*+R*) denotes that a proposal set composed from locally rigid scene flow (*Rig*, [22]) is additionally used. Finally, we optionally make use of egomotion proposals (*+E*), by estimating the dominant 3D motion using our proposal fitting technique on the 2D baseline output. In contrast to [27] we do not make any hard assumptions about the scene, or prefer epipolar motion in the energy. Rather, epipolar motion is one of several proposal solutions, which are used to minimize an energy that can cope with general, non-epipolar motion.

From the results in Table 2 we first observe that the per-pixel refinement (PRSPix) improves results significantly in all measures, on average by 9% to 15%. 2D and 3D regularization lead to rather similar results, possibly explained by the fact that the evaluation does not have ground truth for 3D scene flow, but only for disparity and 2D optical flow. We thus mostly rely on the 2D regularizer, and note that better 3D benchmarks are needed for quantitative evaluation of 3D scene flow methods. Additional occlusion reasoning (*-O*) improves the results, especially for motion estimates in occluded areas, but performance in the stereo case slightly decreases. While occlusion reasoning has a positive overall effect on the accuracy, the effect is somewhat limited here (unlike Fig. 1) due to the limited amount of independent motion in KITTI. Like any proposal-based optimization technique [*e.g.*, 4, 11], the quality of the proposals is of some importance. Still, already the 2D proposal set from *S+F* alone is sufficient to surpass all our baselines by a large margin, including two recent 3D scene flow techniques [3, 22], on average by 33%. Incorporating additional proposals, the average improvement becomes 38% (*PRSPix-2D+R*) and 44% (*PRSPix-2D+R+E*).

We have evaluated our model on the official KITTI benchmark. On a dual-core Intel i7 machine and for one KITTI scene, our current implementation takes $\approx$ 12s for fitting 2 proposal sets and 2000 segments, $\approx$ 32s for per-segment optimization and $\approx$ 18s for per-pixel optimization. The data term is the bottleneck. At time of writing our method (*PRSPix-2D+R+E*) ranks 1^{st} out of 28 published approaches for optical flow in all measures, and 3^{rd} out of 25 published methods in stereo, while (*PRSPix-2D*) ranked 3^{th} and 5^{th} respectively. Similar performance is only achieved by [27], which can only handle epipolar motion. Our approach, in contrast, can cope with independent object motion (see Fig. 1). Moreover, it strongly outperforms another 3D scene flow technique [7], even on its semi-dense output. Finally, even the best general 2D optical flow method is surpassed. To the best of our knowledge, this is the first scene flow method to outperform optical flow algorithms w.r.t. reprojection error on a benchmark set, and thus realize the advantage stemming from the additional stereo information.

5. Conclusion

We have shown that modeling a dynamic scene with local regions corresponding to rigidly moving planes can lead to compelling results for the task of joint geometry and 3D

	FLOW (*All*)				FLOW (*Noc*)				STEREO(*All*)				STEREO (*Noc*)			
Error threshold Z	2	3	4	5	2	3	4	5	2	3	4	5	2	3	4	5
LSF [3]	21.6	16.9	14.3	12.7	16.0	12.0	10.0	8.8	17.6	12.0	9.0	7.2	16.4	10.8	8.0	6.3
Rig [22]	16.1	12.1	10.1	8.8	10.6	7.3	5.7	4.8	15.0	10.6	8.3	6.8	13.7	9.5	7.2	5.8
2D [9, 25]	18.9	15.0	12.8	11.3	11.0	7.9	6.5	5.7	13.5	9.9	8.0	6.7	12.3	8.9	7.0	5.8
PRSSeg-3D	13.8	10.1	8.2	7.1	8.4	5.6	4.5	3.9	9.4	6.8	5.4	4.6	8.4	6.0	4.8	4.0
PRSPix-3D	12.8	9.3	7.6	6.6	7.2	4.7	3.7	3.2	8.1	5.8	4.6	3.9	7.1	5.0	4.0	3.3
PRSSeg-2D	12.4	9.0	7.3	6.4	7.4	5.0	3.9	3.4	8.9	6.4	5.1	4.3	7.9	5.6	4.4	3.7
PRSPix-2D	11.8	8.5	6.9	6.0	6.9	4.5	3.5	3.0	8.3	5.9	4.7	3.9	7.3	5.1	4.0	3.3
PRSPix-O-2D	11.2	7.7	5.9	5.1	6.8	4.4	3.3	2.8	8.3	5.9	4.7	4.0	7.4	5.2	4.1	3.4
PRSPix-2D+R	10.9	7.6	6.0	5.1	6.3	4.1	3.1	2.7	7.9	5.7	4.5	3.8	6.9	4.8	3.8	3.2
PRSPix-2D+R+E	**10.0**	**6.7**	**5.0**	**4.1**	**5.8**	**3.6**	**2.7**	**2.2**	**7.4**	**5.3**	**4.2**	**3.5**	**6.4**	**4.5**	**3.6**	**3.0**

Table 2. Average error rates for the KITTI training set. Error given as the percentage of erroneous pixels (deviation to ground truth above threshold of Z pixels) in non-occluded areas (*Noc*) and over the full image (*All*).

motion estimation. The proposed model achieves accurate geometry and motion boundaries by refining an initial oversegmentation of the scene, and allows for occlusion reasoning. We show that our method substantially outperforms previous dense scene flow approaches on a challenging data set, and even surpasses dedicated state-of-the-art stereo and optical flow techniques at their respective task. Its main limitation are scenes with strongly non-rigid motion or extreme curvature, where the piecewise planar and rigid approximation does not hold. In practice such scenes are quite rare.

In future work we plan to extend our method to sequences of more than two frames, which we believe our formulation is well-suited for. Another interesting avenue would be to embed object-level semantic image understanding into the segmentation scheme.

References

[1] G. Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *PAMI*, 7(4):384–401, 1985.

[2] A. M. Ali, A. A. Farag, and G. L. Gimel'Farb. Optimizing binary MRFs with higher order cliques. *ECCV 2008*.

[3] T. Basha, Y. Moses, and N. Kiryati. Multi-view scene flow estimation: A view centered variational approach. *CVPR'10*.

[4] M. Bleyer, C. Rother, P. Kohli, D. Scharstein, and S. N. Sinha. Object stereo – Joint stereo matching and object segmentation. *CVPR 2011*.

[5] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *ECCV 2004*.

[6] R. L. Carceroni and K. N. Kutulakos. Multi-view scene capture by surfel sampling: From video streams to non-rigid 3D motion, shape and reflectance. *IJCV*, 49:175–214, 2002.

[7] J. Cech, J. Sanchez-Riera, and R. P. Horaud. Scene flow estimation by growing correspondence seeds. *CVPR 2011*.

[8] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? *CVPR 2012*.

[9] H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *PAMI*, 30(2):328–341, 2008.

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

[11] V. Lempitsky, S. Roth, and C. Rother. FusionFlow: Discrete-continuous optimization for optical flow estimation. *CVPR 2008*.

[12] T. Nir, A. Bruckstein, and R. Kimmel. Over-parameterized variational optical flow. *IJCV*, 76(2):205–216, 2008.

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

[14] R. Ranftl, S. Gehrig, T. Pock, and H. Bischof. Pushing the limits of stereo using variational stereo estimation. *IV 2012*.

[15] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof duality. *CVPR'07*.

[16] D. Sun, E. B. Sudderth, and M. J. Black. Layered image motion with explicit occlusions, temporal consistency, and depth ordering. *NIPS 2010*.

[17] M. Unger, M. Werlberger, T. Pock, and H. Bischof. Joint motion estimation and segmentation of complex scenes with label costs and occlusion modeling. *CVPR 2012*.

[18] L. Valgaerts, A. Bruhn, H. Zimmer, J. Weickert, C. Stoll, and C. Theobalt. Joint estimation of motion, structure and geometry from stereo sequences. *ECCV 2010*.

[19] T. Vaudrey, C. Rabe, R. Klette, and J. Milburn. Differences between stereo and motion behaviour on synthetic and real-world stereo sequences. *IVCNZ 2008*.

[20] S. Vedula, S. Baker, R. Collins, T. Kanade, and P. Rander. Three-dimensional scene flow. *CVPR 1999*.

[21] O. Veksler, Y. Boykov, and P. Mehrani. Superpixels and supervoxels in an energy optimization framework. *ECCV 2010*.

[22] C. Vogel, K. Schindler, and S. Roth. 3D scene flow estimation with a rigid motion prior. *ICCV 2011*.

[23] J. Wang and E. Adelson. Representing moving images with layers. *IEEE TIP*, 3:625–638, 1994.

[24] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, and D. Cremers. Efficient dense scene flow from sparse or dense stereo data. *ECCV 2008*.

[25] M. Werlberger. *Convex Approaches for High Performance Video Processing*. PhD thesis, TU Graz, 2012.

[26] K. Yamaguchi, T. Hazan, D. McAllester, and R. Urtasun. Continuous Markov random fields for robust stereo estimation. *ECCV 2012*.

[27] K. Yamaguchi, D. McAllester, and R. Urtasun. Robust monocular epipolar flow estimation. *CVPR 2013*.

[28] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. *ECCV 1994*.