SphereFlow: 6 DoF Scene Flow from RGB-D Pairs

Michael Hornáček[1,*], Andrew Fitzgibbon[2], Carsten Rother[3]
[1]TU Vienna [2]Microsoft Research Cambridge [3]TU Dresden

Abstract

We take a new approach to computing dense scene flow between a pair of consecutive RGB-D frames. We exploit the availability of depth data by seeking correspondences with respect to patches specified not as the pixels inside square windows, but as the 3D points that are the inliers of spheres in world space. Our primary contribution is to show that by reasoning in terms of such patches under 6 DoF rigid body motions in 3D, we succeed in obtaining compelling results at displacements large and small without relying on either of two simplifying assumptions that pervade much of the earlier literature: brightness constancy or local surface planarity. As a consequence of our approach, our output is a dense field of 3D rigid body motions, in contrast to the 3D translations that are the norm in scene flow. Reasoning in our manner additionally allows us to carry out occlusion handling using a 6 DoF consistency check for the flow computed in both directions and a patchwise silhouette check to help reason about alignments in occlusion areas, and to promote smoothness of the flow fields using an intuitive local rigidity prior. We carry out our optimization in two steps, obtaining a first correspondence field using an adaptation of PatchMatch, and subsequently using α-expansion to jointly handle occlusions and perform regularization. We show attractive flow results on challenging synthetic and real-world scenes that push the practical limits of the aforementioned assumptions.

1. Introduction

The growing consumer-level availability of RGB-D data—in particular since the introduction of the inexpensive Microsoft Kinect camera—has made solving computer vision problems by jointly exploring cues in color and depth an increasingly practical pursuit. We present a substantially new way of computing the dense 3D motion field between a pair of consecutive RGB-D frames of a (perhaps non-rigidly) moving scene, making neither of the traditional assumptions of brightness constancy or local surface planarity. The assumption that corresponding pixels have the same color is perhaps the most common simplifying assumption in scene flow, and breaks down as displacements become large. An alternative it to aggregate intensity information over patches of pixels. Traditionally, however, patch-based methods have relied on motion models that assume local surface planarity, which imposes limits on the size of such patches over geometry that is not planar.

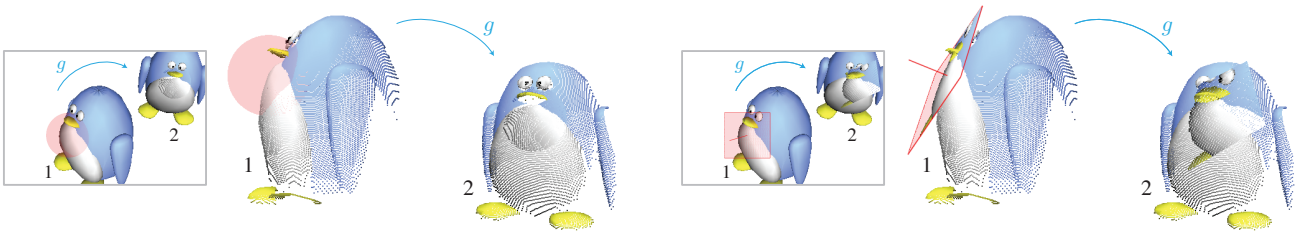
In stark contrast to previous patch-based scene flow techniques, ours exploits the availability of 3D points in RGB-D data by reasoning in terms of so-called '3D point patches'. Our **main contributions** derive from how we reason about patches. By identifying the points constituting our patches as the inliers of *spheres* and relating such patches using 6 DoF 3D rigid body motions, we are able to: (i) overcome the local surface planarity assumption, allowing for larger patches over non-planar surfaces than are possible with traditional motion models; (ii) introduce a 6 DoF consistency check for the flow recovered in both directions; (iii) introduce a patchwise silhouette check to help reason about alignments in occlusion areas; and (iv) introduce an intuitive local rigidity prior to promote smoothness. Finally, a consequence of our approach is that (v) our output is a dense field of 6 DoF 3D rigid body motions, rather than one of 3D translation vectors as is the norm in scene flow.

1.1. Related Work

The term 'scene flow' is due to Vedula *et al*. [26], who introduced it as a dense field of 3D translation vectors for each point in the scene. Traditionally, scene flow has referred to techniques that take RGB images as input and recover 3D structure in addition to 3D motion. For rigid scenes, sparse 3D structure and 6 DoF motion can be obtained using structure from motion (cf. Pollefeys *et al*. [19]).

RGB scene flow methods include Huguet and Devernay [15], Min and Sohn [18], Zhang and Kambhamettu [32], Basha *et al*. [4], Čech *et al*. [25], and Vogel *et al*. [27]. Basha *et al*. use a 3D point cloud representation to directly model the desired 3D unknowns, allowing smoothness assumptions to be imposed directly on the scene flow and

*Michael Hornáček is funded by Microsoft Research through its European Ph.D. scholarship programme.



(a) Our 3D point patches.

(b) Homography induced by the plane.

Figure 1. Our 3D point patches contrasted with the geometry of homography induced by the plane (cf. Hartley and Zisserman [12]). We use the same rigid body motion g—recovered using our approach—for both examples. The local planarity assumption breaks down on non-planar surfaces for large enough patch size. Note also for the overlaid matched points in (a) that because penguin 2 is illuminated frontally but penguin 1 is illuminated from the side, the brightness constancy assumption likewise does not hold at this displacement.

structure. Čech *et al*. compute scene flow across sequences of stereo pairs by growing correspondence seeds. Vogel *et al*. carry out regularization by encouraging a locally rigid 3D motion field using a rigid motion prior, avoiding systematic biases of 2D isotropic regularization.

Pons *et al*. [20] introduce a variational framework for multiple-view motion-stereo estimation that works directly in world space, evolving a 3D vector field to register the input images captured at different times. Carceroni and Kutulakos [8] model the scene as a set of surfels, with each surfel described by shape, reflectance, bump map, and affine motion. They recover surfel parameters by maximizing photoconsistency, but require knowledge of relative camera parameters and of the illumination scenario. Devernay *et al*. [9] likewise proceed by tracking surfels. In both cases, surfels imply the local surface planarity assumption. Vogel *et al*. [28] assign rigidly moving planes to image segments.

Spies *et al*. [24] introduce the range flow constraint and recover 3D motion using a depth and intensity information. They performed sparse motion estimation, which was subsequently regularized to unestimated regions. Letouzey *et al*. [17] claim novelty for using RGB-D data obtained from a Kinect. They use sparse feature points to help guide the motion field estimation for wide displacements and use dense normal flow for short ones. Wedel *et al*. [29] explicitly decouple the position and velocity estimation steps and estimate dense velocities using a variational approach while reporting frame rates of 5 fps on standard consumer hardware. Hadfield and Bowden [11] use a particle filter while assuming brightness constancy. Herbst *et al*. [13] compute RGB-D scene flow using a variational technique, which they apply to rigid motion segmentation. Quiroga *et al*. [21] use a variational technique combining local and global constraints to compute RGB-D scene.

2. Algorithm

We begin by obtaining a dense 6 DoF correspondence field from the first frame to the second and vice versa, using a new variant of the PatchMatch algorithm building on the work of Barnes *et al*. [2, 3], Bleyer *et al*. [5], and Hornáček *et al*. [14]. We detail this step in Section 2.2. In Section 2.1, we present our 3D point patches and the matching cost we aim to minimize, from which the majority of our contributions derive. In Section 2.3, we detail the second step of our algorithm, optimized using α-expansion, which refines the correspondence fields by handling occlusions and promoting 6 DoF smoothness.

2.1. 3D Point Patches

We carry out our correspondence search by reasoning in terms of so-called '3D point patches' under 6 DoF 3D rigid body motions as in Hornáček *et al*. [14], but extend the matching cost presented in their formulation to include the influence of 2D image gradients recoverable from the available RGB. Gradients allow for matching salient texture features without relying on brightness constancy, while our 3D point patch formulation allows for better alignment of such gradients over non-planar surfaces than is possible using traditional motion models that assume local surface planarity (cf. Figure 1). Additionally, our formulation allows us to compare depth between correspondences in a manner that borrows from the ICP literature (cf. Rusinkiewicz and Levoy [22]), which would also not be possible with traditional motion models. Recognizing that object boundaries encoded in depth can be noisy or poorly aligned with RGB, we additionally integrate an optional adaptive support weighting scheme in our matching cost. Finally, we overcome a shortcoming of the formulation in [14]—that tied the number of inlier points constituting a 3D point patch to its corresponding sphere's depth (cf. Figure 2)—enabling us to ensure a more uniform matching quality across the scene.

Formal Definition. Let $g = (\mathtt{R}, \mathbf{t}) \in SE(3)$ denote a 6 DoF rigid body motion in 3D, where $\mathtt{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$. Given one of the two input views, let the *3D point patch* $\mathcal{S}_\mathbf{x}$ denote the subset of the set of 3D points $\mathcal{P} \subset \mathbb{R}^3$ encoded in the depth map and situated within a radius $r_\mathbf{x}$ of the point $\mathbf{P}_\mathbf{x} = Z_\mathbf{x} \cdot \mathtt{K}^{-1}(\mathbf{x}^\top, 1)^\top \in \mathbb{R}^3$, where $Z_\mathbf{x}$ is the depth encoded at the pixel $\mathbf{x} = (x, y)^\top$ and $\mathtt{K}$ is the 3×3

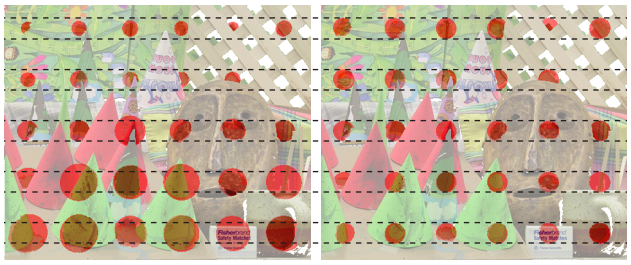


Figure 2. Apparent spatial extent of 3D point patches. Left: fixed sphere radius r renders apparent spatial extent a function of sphere depth. Right: proposed per-pixel radius $r_{\mathbf{x}}$, obtained as a function of the depth $Z_{\mathbf{x}}$ of $\mathbf{P}_{\mathbf{x}}$. In this manner, spheres can be expected to contain a more uniform number of inlier points than for fixed r, thereby providing more uniform matching quality.

camera calibration matrix (cf. Hartley and Zisserman [12]). Our goal is to assign a rigid body motion $g_{\mathbf{x}}$ to each valid pixel $\mathbf{x}$, mapping $\mathcal{S}_{\mathbf{x}}$ in the source view to its analogue in the destination view. A pixel $\mathbf{x}$ is deemed *valid* only if a depth value is encoded at $\mathbf{x}$ in the input depth map.

Radius $r_{\mathbf{x}}$. It is easy to show that the distance in world space between two points both situated at depth Z and projecting to neighboring pixels in image space is given by Z/f,[1] where f is the camera's focal length in units of pixels. Accordingly, the number of points constituting a 3D point patch given a fixed radius r can on average be expected to decrease as depth increases (cf. Figure 2), and with it so too can confidence in the strength of any matching score based on such points. In order to alleviate this problem of reasoning in terms of a fixed radius r, we proceed instead by assigning a tailored radius $r_{\mathbf{x}}$ for each pixel $\mathbf{x}$:

$$r_{\mathbf{x}} = r_{\text{pix}} \cdot Z_{\mathbf{x}} / f, \tag{1}$$

where r_{pix} is a fixed radius *in pixels*. In this manner, each sphere appears to have equal size from the viewpoint of the camera, and thus the spheres can be expected to contain a more uniform number of inlier points than for fixed r.

Matching Cost. Let I, I' and G, G' be the source and destination color and gradient images, respectively, and let π, π' denote projection into the source and destination views. We compute gradient similarity by projection and interpolation, promoting sub-pixel accuracy with respect to salient texture edges in image space:

$$C^{\text{gr}}_{\mathbf{x}}(g) = \sum_{\mathbf{P} \in \mathcal{S}_{\mathbf{x}}} w(\mathbf{x}, \pi(\mathbf{P})) \cdot \| G(\pi(\mathbf{P})) - G'(\pi'(g(\mathbf{P}))) \|_2^2 , \tag{2}$$

where $w(\mathbf{x}, \mathbf{x}') = \exp\left(-\|(I(\mathbf{x}) - I(\mathbf{x}'))\|_2 / \gamma\right)$ implements a form of adaptive support weighting (cf. Yoon and Kweon [31]), which is valuable when object boundaries in the depth map are noisy or poorly aligned with RGB. We compute L2 color distance for adaptive support weighting in the CIE L*a*b color space. Let $NN_{\mathcal{S}}(\mathbf{P})$ denote the nearest neighbor point to $\mathbf{P}$ in the set $\mathcal{S} \subset \mathbb{R}^3$. We compute 3D point similarity by

$$C^{\text{pt}}_{\mathbf{x}}(g) = \sum_{\mathbf{P} \in \mathcal{S}_{\mathbf{x}}} w(\mathbf{x}, \pi(\mathbf{P})) \cdot \| g(\mathbf{P}) - NN_{\mathcal{P}'}(g(\mathbf{P})) \|_2^2 , \tag{3}$$

where $\mathcal{P}'$ denotes the set of points encoded in the depth map of the destination view. Reasoning in terms of nearest neighbors in 3D for point similarity allows for handling shot noise and invalid pixels without special treatment, which would not be possible by projection and interpolation. Moreover, it allows for a natural delineation of boundaries at depth discontinuities, insofar as object boundaries encoded in the input depth maps are of reasonable quality. We compute the final matching cost according to

$$C_{\mathbf{x}}(g) = \begin{cases} C^{\text{pt}}_{\mathbf{x}}(g) + \alpha \cdot C^{\text{gr}}_{\mathbf{x}}(g) & \text{if } \mathbf{x} \text{ is valid} \\ \infty & \text{otherwise} \end{cases} , \tag{4}$$

where α is a fixed weight.

2.2. Dense 6 DoF Matching via PatchMatch

We turn to PatchMatch (cf. Barnes *et al.* [2, 3]) to carry out our dense 6 DoF matching, building primarily upon the PatchMatch variant introduced in Hornáček *et al.* [14] for depth super resolution, and upon that introduced in Bleyer *et al.* [5] for stereo. Our goal is to assign a rigid body motion $g_{\mathbf{x}}$ to each valid pixel $\mathbf{x}$, mapping the 3D point patch $\mathcal{S}_{\mathbf{x}}$ in the source view to its analogue in the destination view. We begin with a semi-random initialization step, assigning a first guess to each valid pixel. Next, for i iterations, we visit each $\mathbf{x}$, carrying out (i) spatial propagation, (ii) j additional semi-random initializations, (iii) k refinements, and (iv) view propagation. In each of the steps (i-iv), a candidate rigid body motion is adopted at $\mathbf{x}$ if doing so yields equal or lesser matching cost. For the first of the two views, we visit its pixels in scanline order, upper left to lower right for even iterations, lower right to upper left for odd. A contribution of ours—applicable to stereo as well as to scene flow—is to promote convergence via view propagation by traversing the pixels of the second view *in the opposite order*, *in parallel* with the first view. We describe the individual steps in greater detail below.

Semi-Random Initialization. For each valid pixel $\mathbf{x}$ in the source view, we randomly pick a point $\mathbf{P}_{\mathbf{x}'}$ in the destination view within a maximum search radius v of $\mathbf{P}_{\mathbf{x}}$, giving a translation vector $\mathbf{P}_{\mathbf{x}'} - \mathbf{P}_{\mathbf{x}}$ (3 DoF). Additionally, we obtain candidate translations by matching SURF features in image space. We obtain the remaining 3 DoF by computing the rotation minimizing arc length between the surface normal vector at $\mathbf{P}_{\mathbf{x}}$ and that at $\mathbf{P}_{\mathbf{x}'}$ (2 DoF), and choosing a random around-normal angular perturbation (1 DoF).

[1] We provide a derivation in the supplementary material.

Spatial Propagation. Given a traversal from upper left to lower right, we consider at $\mathbf{x}$ the rigid body motions $g_{\mathbf{x}_\mathrm{n}}, g_{\mathbf{x}_\mathrm{w}}$, where $\mathbf{x}_\mathrm{n} = (x, y-1)^\top, \mathbf{x}_\mathrm{w} = (x-1, y)^\top$, and adopt a motion if doing so yields equal or lesser matching cost. Analogously, for a traversal from lower right to upper left, we consider $g_{\mathbf{x}_\mathrm{s}}, g_{\mathbf{x}_\mathrm{e}}$, where $\mathbf{x}_\mathrm{s} = (x, y+1)^\top, \mathbf{x}_\mathrm{e} = (x+1, y)^\top$. The geometric rationale behind spatial propagation in our 6 DoF setting can be understood by observing that if two objects are related by a rigid body motion, then any corresponding pair of 3D point patches is related (modulo noise or sampling) by precisely the same motion.

Refinement. We aim of to improve upon $g_\mathbf{x}$ by gently perturbing the motion. We build our candidates as in the semi-random initialization step, but try different combinations of changing the translation by hopping to a neighboring point in the destination view (3 DoF), altering the rotation to reflect a different destination normal (2 DoF), or modifying the around-normal rotation (1 DoF). With each try, we adopt the resulting candidate motion if doing so gives equal or lesser matching cost, and progressively reduce the allowed range of deviation from the current best.

View Propagation. As a last step when visiting a pixel $\mathbf{x}$, we project $g_\mathbf{x}(\mathbf{P}_\mathbf{x})$ to the nearest integer pixel $\mathbf{x}'$ in the destination view, where we evaluate the inverse $g_\mathbf{x}^{-1}$ of $g_\mathbf{x}$ provided that $\mathbf{x}'$ is valid. We adopt $g_\mathbf{x}^{-1}$ at $\mathbf{x}'$ if doing so gives equal or lesser matching cost. Since we carry out PatchMatch in parallel in both directions while traversing pixels in opposite order, by the time a pixel is reached in one view the most recent match available from the other has already been propagated. In contrast, Bleyer *et al.* [5] treat views sequentially, with the effect of carrying out their form of view propagation *after* a full iteration is completed.

2.3. Occlusion Handling and Regularization

The dense correspondence search algorithm from Section 2.2 is reasonable only over points visible in both views, and can be expected to fail in areas where occlusions arise. We introduce a 6 DoF consistency check in the aim of distinguishing good matches from bad. In a first occlusion handling step, we assign to each valid pixel $\mathbf{x}$ that failed the check for $g_\mathbf{x}$ the motion $g_{\mathbf{x}'}$, such that $\mathbf{P}_{\mathbf{x}'}$ is the nearest neighbor point to $\mathbf{P}_\mathbf{x}$ corresponding to the subset of pixels that passed, drawn from the same view. Next, we carry out regularization of the motion field, which we reduce to a labeling problem optimized using α-expansion (cf. Boykov *et al.* [6]) over unary and pairwise potentials. Recognizing that over pixels that failed the consistency check, our unary potentials cannot rely on any criterion derived from our matching cost, we introduce a silhouette check to promote at least patchwise edge alignment from the viewpoint of the camera. We introduce an intuitive local rigidity prior as our pairwise potential. In contrast to Section 2.2, we carry out all steps in this section sequentially: first for the first view, then for the second.

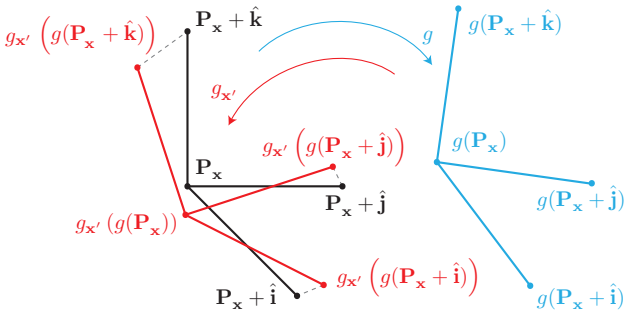


Figure 3. 6 DoF consistency check for g at $\mathbf{x}$. We project $g(\mathbf{P}_\mathbf{x})$ to the nearest integer pixel $\mathbf{x}'$ in the destination view, where we obtain the assigned motion $g_{\mathbf{x}'}$. We transform a unit tripod centered on the point $\mathbf{P}_\mathbf{x}$ (black) forward by g (blue) and then backward by $g_{\mathbf{x}'}$ (red). The check fails if any one of the distances in world space indicated by the three dashed lines exceeds a threshold.

Consistency Check. For a motion g under consideration at a pixel $\mathbf{x}$—by analogy to the left-right consistency check over disparity in stereo (cf. e.g. Bleyer *et al.* [5])—we project $g(\mathbf{P}_\mathbf{x})$ to the nearest integer pixel $\mathbf{x}'$ in the destination view and fail our check if $\mathbf{x}'$ is not valid or if the distance in image space between $\mathbf{x}$ and the projection of $g_{\mathbf{x}'}(g(\mathbf{P}_\mathbf{x}))$ back into the source view exceeds 1 pixel. However, such a check is by itself unsatisfactory: the distance in world space corresponding to a pixel displacement in image space grows as depth increases, and such a check ignores the rotational components of $g, g_{\mathbf{x}'}$. Accordingly, we additionally fail the check if $\exists \mathbf{v} \in \{\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}\}$ such that

$$\left\| (\mathbf{P}_\mathbf{x} + \mathbf{v}) - g_{\mathbf{x}'}\left(g(\mathbf{P}_\mathbf{x} + \mathbf{v})\right) \right\|_2 > \delta, \tag{5}$$

where $\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}$ are the standard (unit) basis vectors for $\mathbb{R}^3$. We set δ to Z_med/f (cf. Section 2.1), where Z_med is the median depth across both views, however this does not preclude alternative schemes for choosing a threshold. An illustration of the geometry of (5) is provided in Figure 3. Finally, we fail the check if $|\mathcal{S}_\mathbf{x}| < n$ or $|\mathcal{S}_{\mathbf{x}'}| < n$ in order to ensure a minimum matching support.

Initial Labeling. Each valid pixel $\mathbf{x}$ that passed the consistency check for $g_\mathbf{x}$ is assigned its own unique label l. For each valid pixel $\mathbf{x}$ that failed the consistency check, we instead assign to $\mathbf{x}$ the label assigned to the pixel $\mathbf{x}'$ such that $\mathbf{P}_{\mathbf{x}'} = NN_{\tilde{\mathcal{P}}}(\mathbf{P}_\mathbf{x})$, where $\tilde{\mathcal{P}}$ denotes the set of points corresponding to the set $\tilde{\mathcal{X}}$ of valid pixels that *passed* the consistency check for the view under consideration.

Silhouette Check. Pixels occluded in the destination view are likely to fail the consistency check, as our matching cost function makes sense only over patches visible in both views. For such pixels, we want to at least promote patchwise alignment of depth edges. The silhouette check

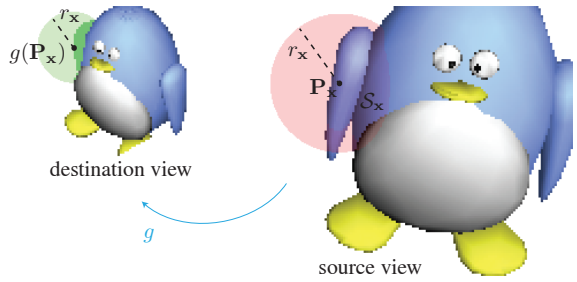


Figure 4. Silhouette check for g at $\mathbf{x}$. The check is passed if the points $g(\mathcal{S}_{\mathbf{x}})$ all project into the pixel mask in the destination view corresponding to the inlier points of the sphere of the same radius $r_{\mathbf{x}}$, but centered on $g(\mathbf{P}_{\mathbf{x}})$ (emphasized in green). Note that $\mathcal{S}_{\mathbf{x}}$ is partly self-occluded in the further penguin.

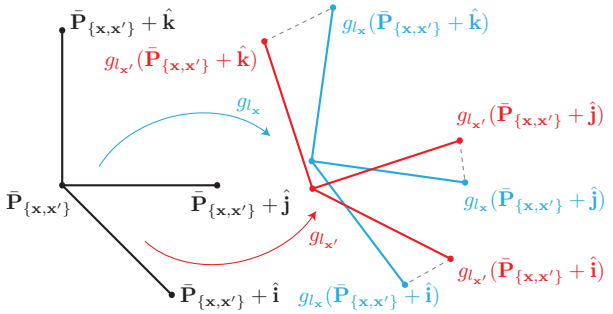


Figure 5. Local rigidity prior. Given a pixel $\mathbf{x}$ and a neighboring pixel $\mathbf{x}'$, $\{\mathbf{x},\mathbf{x}'\} \in \mathcal{N}$, and the rigid body motions $g_{l_{\mathbf{x}}}, g_{l_{\mathbf{x}'}}$ corresponding to the labels $l_{\mathbf{x}}, l_{\mathbf{x}'}$, the prior returns the sum of squares of the distances in world space indicated by the three dashed lines.

is passed at $\mathbf{x}$ for g if all points $g(\mathcal{S}_{\mathbf{x}})$ project into the pixel mask corresponding to the points in the opposite view that are inliers of the sphere of radius $r_{\mathbf{x}}$ centered on $g(\mathbf{P}_{\mathbf{x}})$, as illustrated in Figure 4. Optionally, this mask can be dilated morphologically in order for the check to be more permissive towards sampling artifacts or noisy object boundaries.

Regularization. We formulate our regularization as an energy minimization problem, taking the form:

$$E(\mathbf{l}) = \sum_{\mathbf{x}\in\mathcal{X}} D_{\mathbf{x}}(l_{\mathbf{x}}) + \sum_{\{\mathbf{x},\mathbf{x}'\}\in\mathcal{N}} V_{\{\mathbf{x},\mathbf{x}'\}}(l_{\mathbf{x}}, l_{\mathbf{x}'}), \quad (6)$$

where $\mathcal{X}$ denotes the set of pixels in the image and $\mathcal{N}$ the set of its 4-connected pixel neighbors, and where $D_{\mathbf{x}}(l_{\mathbf{x}})$ denotes the unary potential at $\mathbf{x}$ and $V_{\{\mathbf{x},\mathbf{x}'\}}(l_{\mathbf{x}}, l_{\mathbf{x}'})$ the pairwise potential between $\mathbf{x}, \mathbf{x}'$. For pixels $\mathbf{x}$ that are not valid, we set $D_{\mathbf{x}}(l_{\mathbf{x}}) = 0$. Otherwise, if the consistency check was *passed* at $\mathbf{x}$ for the motion $g_{\mathbf{x}}$ assigned in our dense matching stage, $D_{\mathbf{x}}(l_{\mathbf{x}})$ takes the form $D^{\mathrm{p}}_{\mathbf{x}}(l_{\mathbf{x}})$:

$$D^{\mathrm{p}}_{\mathbf{x}}(l_{\mathbf{x}}) = \begin{cases} 0 & \text{if } g_{l_{\mathbf{x}}} \text{ passes consistency check at } \mathbf{x} \\ \rho & \text{otherwise} \end{cases}, \quad (7)$$

where ρ reflects the trust we lend to the assigned motions that satisfied the consistency check, recognizing that they were the strongest matches that PatchMatch succeeded in finding. In future work, we plan to address the merits of setting $D^{\mathrm{p}}_{\mathbf{x}}(l_{\mathbf{x}})$ to our matching cost when the check is passed; here, we opt instead for an approximation with the advantage of speed. If the consistency check was *failed* at $\mathbf{x}$ for $g_{\mathbf{x}}$, $D_{\mathbf{x}}(l_{\mathbf{x}})$ instead takes the form $D^{\mathrm{f}}_{\mathbf{x}}(l_{\mathbf{x}})$:

$$D^{\mathrm{f}}_{\mathbf{x}}(l_{\mathbf{x}}) = \begin{cases} 0 & \text{if } g_{l_{\mathbf{x}}} \text{ passes silhouette check at } \mathbf{x} \\ \kappa & \text{otherwise} \end{cases}, \quad (8)$$

where κ weighs the influence of the silhouette check against the pairwise potentials. Our pairwise potentials reduce to 3D point SSD analogously to (3), but are computed over the axes of a unit tripod transformed according to the motions $g_{l_{\mathbf{x}}}, g_{l_{\mathbf{x}'}}$:

$$V_{\{\mathbf{x},\mathbf{x}'\}}(l_{\mathbf{x}}, l_{\mathbf{x}'}) = \beta_{\{\mathbf{x},\mathbf{x}'\}} \cdot \sum_{\mathbf{v}\in\{\hat{\mathbf{i}},\hat{\mathbf{j}},\hat{\mathbf{k}}\}} \left\| g_{l_{\mathbf{x}}}(\bar{\mathbf{P}}_{\{\mathbf{x},\mathbf{x}'\}} + \mathbf{v}) - g_{l_{\mathbf{x}'}}(\bar{\mathbf{P}}_{\{\mathbf{x},\mathbf{x}'\}} + \mathbf{v}) \right\|_2^2 . \quad (9)$$

We set $\beta_{\{\mathbf{x},\mathbf{x}'\}}$ to a fixed weight β if both $\mathbf{x}, \mathbf{x}'$ are valid and $\|\mathbf{P}_{\mathbf{x}} - \mathbf{P}_{\mathbf{x}'}\|_2 \leq r_{\mathrm{pix}} \cdot Z_{\mathrm{med}}/f$ (cf. Section 2.1), to 0 otherwise. This has the effect of regularizing only over pixels whose corresponding points could both be inliers of a sphere at depth Z_{med}. Our manner of proceeding does not preclude alternative schemes for choosing a maximum radius. We set $\bar{\mathbf{P}}_{\{\mathbf{x},\mathbf{x}'\}} = (\mathbf{P}_{\mathbf{x}} + \mathbf{P}_{\mathbf{x}'})/2$, noting that the energy formulation in (6) implies that $V_{\{\mathbf{x},\mathbf{x}'\}} = V_{\{\mathbf{x}',\mathbf{x}\}}$, and that the local effect of the motions $g_{l_{\mathbf{x}}}, g_{l_{\mathbf{x}'}}$ on $\bar{\mathbf{P}}_{\{\mathbf{x},\mathbf{x}'\}}$ can be expected to approximate their effect on $\mathbf{P}_{\mathbf{x}}, \mathbf{P}_{\mathbf{x}'}$ when the two points are spatial neighbors in 3D belonging to the same object, which is precisely the target scenario. We illustrate the geometry of our local rigidity prior in Figure 5.

We minimize our energy via the α-expansion algorithm of Boykov *et al*. [6], using QPBO (cf. Lempitsky *et al*. [16]) to compute the expansions. Labels l are drawn at random (without replacement) from the set of pixels for which the consistency check was satisfied in the initial labeling.

3. Evaluation

There exists at present no benchmark tailored to evaluating RGB-D scene flow. In Section 3.1, we accordingly give a quantitative evaluation following the example of Huguet and Devernay [15] by using the color images and ground truth disparity maps available with frames 2 and 6 of the Middlebury Cones, Teddy, and Venus (cf. Scharstein and Szeliski [23]) stereo data sets, respectively, to compare against the known ground truth motion. In Section 3.2, we present qualitative results for the Middlebury data, and for challenging synthetic and real-world (Kinect) data sets.

3.1. Quantitative Evaluation

The 3D scene motion for the static Cones, Teddy, and Venus data sets is due entirely to the motion of the camera

and is purely translational in the X-direction of the camera coordinate frame, in the magnitude of the chosen baseline. While that motion in 3D is simple, the matching problem is nevertheless confounded by occlusions and geometry of varying complexity. We compare against Huguet and Devernay [15] and Basha *et al*. [4]—who use only RGB images as input—in end point error (RMS-OF) and average angular error (AAE) over the 2D optical flow vectors obtained by projecting the output 3D displacements to image space. Following the example of Hadfield and Bowden [11],[2] we additionally give disparity change error (RMS-Vz) for the RGB-D scene flow techniques, namely for Hadfield and Bowden [11], Quiroga *et al*. [21], and our method. For reference, we compute numbers for the RGB optical flow techniques of Brox and Malik [7] and Xu *et al*. [30]. Our results placed our method as the top performer among scene flow algorithms considered in our quantitative evaluation. Numbers and additional explanation are provided in Table 1.

3.2. Qualitative Evaluation

We visualize the recovered correspondence fields by projecting the 3D displacements to 2D optical flow vectors, and coloring those vectors in the conventional manner (cf. Baker *et al*. [1]). In Figure 6, we show our intermediate and final results, contrasted with ground truth colorings. In Figures 7, 8, and 9, we show analogous results for large displacement Kinect data sets of varying complexity, and compare against the results of the RGB-D scene flow techniques of Hadfield and Bowden [11], Herbst *et al*. [13], and Quiroga *et al*. [21]. In Figure 10 we visualize our results—in a manner akin to Vedula *et al*. [26]—by flowing all 3D points in both frames for our final results on the data set in Figure 9 to intermediate points in time, demonstrating the visual credibility of our recovered motions. Finally, in Figure 11 we provide our results for four pairs of the penguins data set, showing outstanding flow results on complex non-planar geometry at large displacements.

3.3. Algorithm Parameters

Radius r_{pix} was set to 15. For PatchMatch, j was set to 3, k to 5. Number of iterations i was 2 for Kinect data, 1 otherwise. Search radius v was set to comfortably exceed the maximum displacement across the data sets. Since $C_{\mathbf{x}}^{\text{pt}}$ is a function of point depth, the weight α between $C_{\mathbf{x}}^{\text{gr}}$ and $C_{\mathbf{x}}^{\text{gr}}$ in (4) was set by effectively first scaling the two terms by (approximately) their maximum possible value, respectively, and then applying a relative weight. We approximated the maximum for $C_{\mathbf{x}}^{\text{pt}}$ by $(Z_{\text{med}}/f)^2$ (cf. Section 2.1), where Z_{med} is the median depth across both views. The relative weight we gave to $C_{\mathbf{x}}^{\text{gr}}$ after scaling was 100. We set γ for adaptive support weighting to 10 like Bleyer *et al*. [5]. For regularization, κ was fixed to 1, and ρ and β were set to 10000 and 5 for Kinect data (giving a large degree of trust to the output of PatchMatch, with gentle regularization), to 1 and 10000 otherwise (promoting a heavily regularized solution). Minimum sphere inlier count n was 10. Pixel masks for the silhouette check were dilated by 5 pixels for Kinect data (owing to poor edge quality in Kinect depth maps), 1 otherwise. We considered 25 labels per view.

[2] Additional details on evaluation methodology are provided in the dissertation of Hadfield [10].

4. Conclusion

We presented a technique for computing dense 6 DoF scene flow between a pair of consecutive RGB-D frames. Rather than rely on brightness constancy or local surface planarity as in most previous work, our main contribution is to reason instead in terms of patches of 3D points identified as inliers of spheres, which we match under 6 DoF 3D rigid body motions. We showed compelling results on the Middlebury Cones, Teddy, and Venus data sets as well as on challenging synthetic and real-world scenes.

References

[1] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 2011. 6

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *SIGGRAPH*, 2009. 2, 3

[3] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized PatchMatch correspondence algorithm. In *ECCV*, 2010. 2, 3

[4] T. Basha, Y. Moses, and N. Kiryati. Multi-view scene flow estimation: A view centered variational approach. *IJCV*, 2013. 1, 6, 8

[5] M. Bleyer, C. Rhemann, and C. Rother. PatchMatch Stereo - Stereo matching with slanted support windows. In *BMVC*, 2011. 2, 3, 4, 6

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 2001. 4, 5

[7] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *PAMI*, 2011. 6, 8

[8] 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*, 2002. 2

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

[10] S. Hadfield. *The estimation and use of 3D information, for natural human action recognition*. PhD thesis, University of Surrey, 2013. 6

[11] S. Hadfield and R. Bowden. Scene particles: Unregularized particle based scene flow estimation. *PAMI*, 2013. 2, 6, 8

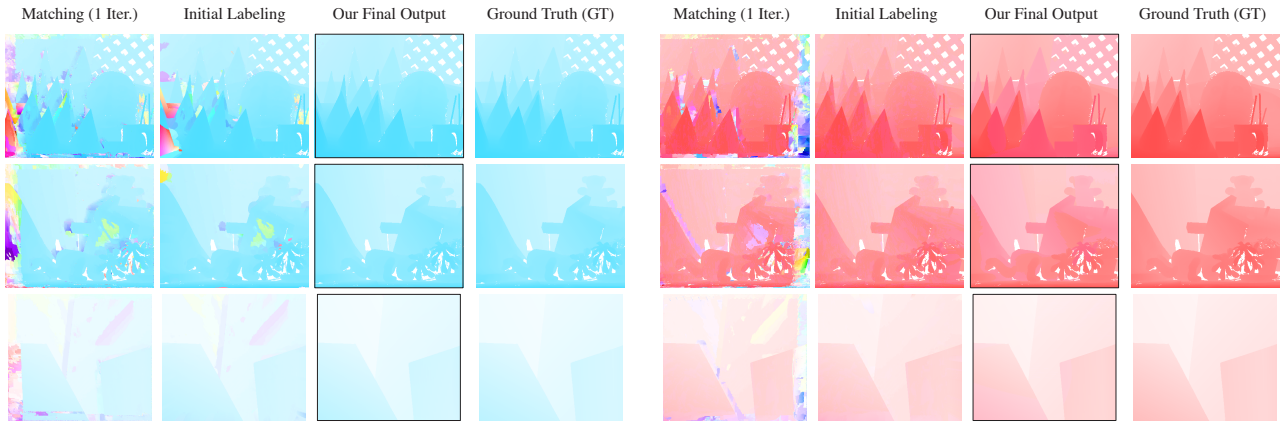


Figure 6. Middlebury Cones, Teddy, and Venus. 2D optical flow coloring by projection of 3D displacements to image space, for the left (blue) and right (red) views. Numbers in Table 1 correspond to the left view. Figure best viewed magnified in the electronic version.

Figure 7. Large displacement motion (Kinect), in large part towards the camera. Our algorithm and that of Quiroga *et al*. [21] do not fail on the forward motion as does that of Herbst *et al*. [13]; our output does not suffer from the haze artefacts present in that of Quiroga *et al*.

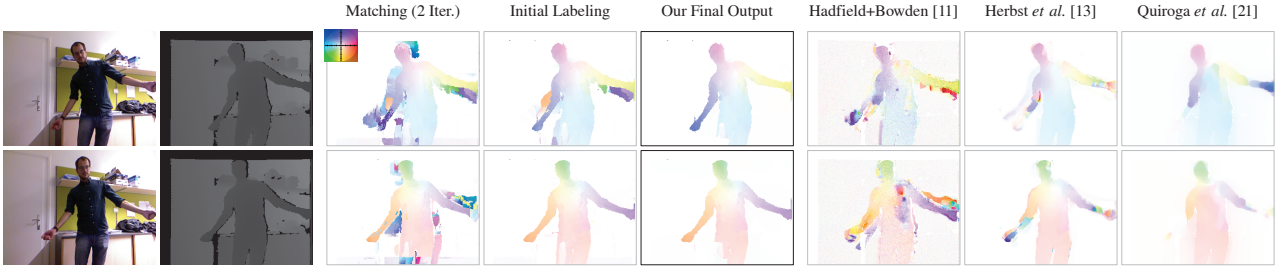


Figure 8. Large displacement motion (Kinect). We best capture the non-rigid motion of the arms while appropriately filling the pixels of the occluded background, again avoiding the haze artefacts present in the output of Quiroga *et al*. [21].

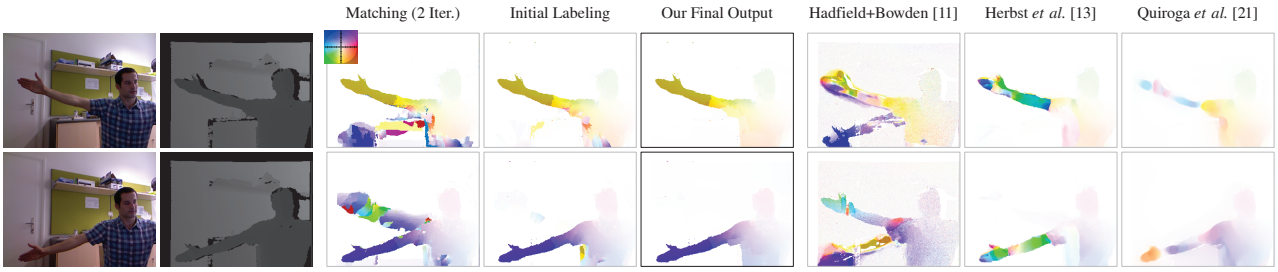


Figure 9. Large displacement motion (Kinect). All three of our competitors fail altogether on the motion of the arm. We visualize our results in Figure 10, demonstrating the visual credibility of our recovered motions.

Figure 10. Visualization of our final results for the Kinect data set in Figure 9. We flow all points encoded in the two input frames to a common point in time, in a manner akin to Vedula *et al*. [26]. Visualization shown at a novel viewpoint.

	Cones			Teddy			Venus		
	RMS-OF	RMS-Vz	AAE	RMS-OF	RMS-Vz	AAE	RMS-OF	RMS-Vz	AAE
Brox and Malik [7] (RGB only)	2.83	1.75†	0.39	3.20	0.47†	0.39	0.72	0.14†	1.28
Xu *et al*. [30] (RGB only)	1.66	1.15†	0.21	1.7	0.5†	0.28	0.3	0.22†	1.43
Basha *et al*. [4] (RGB only)	0.58	N/A	**0.39**	0.57	N/A	1.01	**0.16**	N/A	1.58
Huguet and Devernay [15] (RGB only)	1.10	N/A	0.69	1.25	N/A	0.51	0.31	N/A	0.98
Hadfield and Bowden (RGB + GT Depth) [11]	1.24	0.06	1.01	0.83	0.03	0.83	0.36	0.02	1.03
Quiroga *et al*. (RGB + GT Depth) [21]	0.57	0.05	0.42	0.69	0.04	0.71	0.31	**0.00**	1.26
Our Method (RGB + GT Depth)	**0.54**	**0.02**	0.52	**0.35**	**0.01**	**0.15**	0.26	0.02	**0.53**

Table 1. Quantitative evaluation on RMS-OF (end point error), RMS-Vz (disparity change error), and AAE (average angular error). The topmost two methods are RGB optical flow algorithms; the next two are RGB scene flow algorithms that compute 3D translational flow and depth jointly. The remaining three are RGB-D scene flow techniques. Results were computed over all valid pixels, here meaning that GT disparity is nonzero and the pixel is marked as unoccluded in the Middlebury GT occlusion map. † indicates that RMS-Vz was computed by estimating 3D translational flow by interpolating depth encoded at the start and end points given its 2D flow vector. Accordingly, for the two optical flow techniques, we also ignored pixels at which the recovered 2D flow pointed to pixels with GT disparity of 0, since no end point depth could be interpolated. For Hadfield and Bowden, we additionally deemed pixels for which no flow was recovered as invalid.

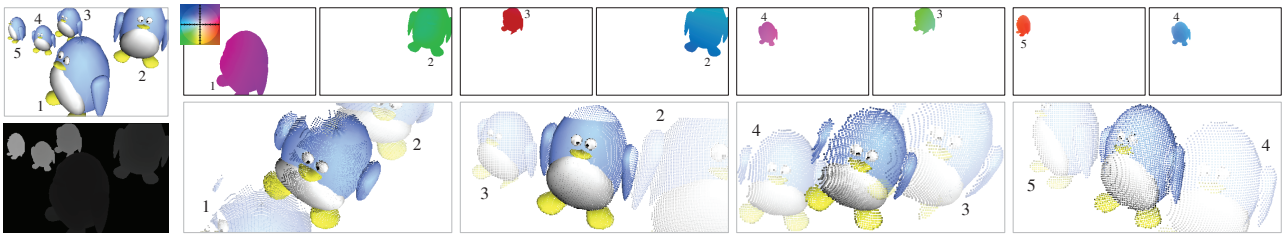


Figure 11. Large displacement synthetic data set from Hornáček *et al*. [14] for which assumptions of brightness constancy and local surface planarity (for discriminative patch size) fail pronouncedly. We run our algorithm on consecutive penguin pairs and visualize our results at a novel viewpoint by flowing all points, respectively, to an intermediate point in time. Original points shown with transparency for reference.

[12] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004. 2, 3

[13] E. Herbst, X. Ren, and D. Fox. RGB-D flow: Dense 3-D motion estimation using color and depth. In *ICRA*, 2013. 2, 6, 7

[14] M. Hornáček, C. Rhemann, M. Gelautz, and C. Rother. Depth super resolution by rigid body self-similarity in 3D. In *CVPR*, 2013. 2, 3, 8

[15] F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *ICCV*, 2007. 1, 5, 6, 8

[16] V. Lempitsky, C. Rother, S. Roth, and A. Blake. Fusion moves for markov random field optimization. *PAMI*, 2010. 5

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

[18] D. Min and K. Sohn. Edge-preserving simultaneous joint motion-disparity estimation. In *ICPR*, 2006. 1

[19] M. Pollefeys, R. Koch, and L. V. Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. *IJCV*, 1999. 1

[20] J.-P. Pons, R. Keriven, and O. Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *IJCV*, 2007. 2

[21] J. Quiroga, F. Devernay, and J. L. Crowley. Local/Global Scene Flow Estimation. In *ICIP*, 2013. 2, 6, 7, 8

[22] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *3DIM*, 2001. 2

[23] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *CVPR*, 2003. 5

[24] H. Spies, B. Jähne, and J. L. Barron. Range flow estimation. *CVIU*, 2002. 2

[25] J. Čech, J. Sanchez-Riera, and R. Horau. Scene flow estimation by growing correspondence seeds. In *CVPR*, 2011. 1

[26] S. Vedula, S. Baker, P. Rander, R. T. Collins, and T. Kanade. Three-dimensional scene flow. *PAMI*, 2005. 1, 6, 7

[27] C. Vogel, K. Schindler, and S. Roth. 3D scene flow estimation with a rigid motion prior. In *ICCV*, 2011. 1

[28] C. Vogel, K. Schindler, and S. Roth. Piecewise rigid scene flow. In *ICCV*, 2013. 2

[29] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, and D. Cremers. Efficient dense scene flow from sparse or dense stereo data. In *ECCV*, 2008. 2

[30] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *PAMI*, 2012. 6, 8

[31] K. J. Yoon and I. S. Kweon. Locally adaptive support-weight approach for visual correspondence search. In *CVPR*, 2005. 3

[32] Y. Zhang and C. Kambhamettu. On 3D scene flow and structure estimation. In *CVPR*, 2001. 1