

# Topology-Aware Non-Rigid Point Cloud Registration

Konstantinos Zampogiannis, *Student Member, IEEE*, Cornelia Fermüller, *Member, IEEE*, and Yiannis Aloimonos, *Member, IEEE*

**Abstract**—In this paper, we introduce a non-rigid registration pipeline for pairs of unorganized point clouds that may be topologically different. Standard warp field estimation algorithms, even under robust, discontinuity-preserving regularization, tend to produce erratic motion estimates on boundaries associated with ‘close-to-open’ topology changes. We overcome this limitation by exploiting backward motion: in the opposite motion direction, a ‘close-to-open’ event becomes ‘open-to-close’, which is by default handled correctly. At the core of our approach lies a general, topology-agnostic warp field estimation algorithm, similar to those employed in recently introduced dynamic reconstruction systems from RGB-D input. We improve motion estimation on boundaries associated with topology changes in an efficient post-processing phase. Based on both forward and (inverted) backward warp hypotheses, we explicitly detect regions of the deformed geometry that undergo topological changes by means of local deformation criteria and broadly classify them as ‘contacts’ or ‘separations’. Subsequently, the two motion hypotheses are seamlessly blended on a local basis, according to the type and proximity of detected events. Our method achieves state-of-the-art motion estimation accuracy on the MPI Sintel dataset. Experiments on a custom dataset with topological event annotations demonstrate the effectiveness of our pipeline in estimating motion on event boundaries, as well as promising performance in explicit topological event detection.

**Index Terms**—non-rigid registration, warp field, dense motion estimation, surface deformation, dynamic topology

## 1 INTRODUCTION

MOTION estimation in 3D is a problem of great importance in computer vision, robotics, and computer graphics, playing a central role in a wide range of applications that include 3D scene reconstruction/modeling, human and object pose tracking, robot localization, augmented reality, human-computer interfaces and deformable shape manipulation. The advent of affordable, commercial depth sensors has caused significant research effort on 3D motion estimation from 3D input, leading to the development of RGB-D algorithms for fast visual odometry [1], [2], efficient and accurate scene flow estimation [3], [4], as well as notable SLAM systems for both static [5], [6] and dynamic [7], [8] environments.

Given the availability of 3D input, dense non-rigid registration is the most general motion estimation problem and it is particularly challenging. In its general form, the problem can be described as computing a motion field, densely supported on the surface of a 3D shape, that deforms the latter in order to geometrically align it to another, fixed “template” shape. This process of non-rigid 3D registration shares fundamental similarities with 2D image registration, known in the computer vision community as optical flow estimation: both problems pose similar challenges in deriving formulations that lead to accurate alignment while encoding reasonable prior constraints (regularization) to overcome ill-posedness.

A classical problem variant that is closely related to 3D non-rigid registration is that of RGB-D scene flow. Given a pair of images, scene flow refers to the per-pixel 3D motion of observed points in space from the first frame to the second; optical flow refers to the per-pixel 2D projected motion. There have been a number of successful recent works on scene flow estimation from RGB-D frame pairs, following both classical (variational) [3], [4], [9], [10], [11] and deep learning [12] frameworks. While of great relevance to a number of motion reasoning tasks, RGB-D scene flow targets a specific instance of dense 3D motion estimation, as it inherently registers pairs of 2.5D geometries (depth maps). This hinders its application in scenarios that require alignment of arbitrary 3D geometries, such as *model-to-frame* registration for dynamic reconstruction or *model-to-model* shape deformation.

Recently introduced dynamic reconstruction pipelines from RGB-D input [7], [8], [13], [14] solve a more general problem by implementing warp field optimization algorithms for their *model-to-frame* registration step. Despite adopting different approaches for their model representations and surface fusion steps, they all rely on similar, point cloud based formulations for non-rigid registration. Scenes with dynamic topology are a challenging case for dynamic reconstruction systems: [7] and [8] make no provisions at all for these cases, while [13] and [14] deal with registration errors that occur because of dynamic topology at a subsequent stage, by discarding problematic regions and reinitializing model tracking. The fully volumetric approaches of [15] and [16] do not use point representations for registration, directly aligning Signed Distance Fields (SDFs) [17] instead. While they intrinsically handle topological changes, significant scalability limitations are introduced by relying on

- K. Zampogiannis (kzampog@cs.umd.edu) and Y. Aloimonos (yiannis@cs.umd.edu) are with the Department of Computer Science, University of Maryland, College Park, MD, 20742.
- C. Fermüller (fer@umiacs.umd.edu) is with the University of Maryland Institute for Advanced Computer Studies (UMIACS), University of Maryland, College Park, MD, 20742.

volumetric representations. To the best of our knowledge, there exists no non-rigid *point cloud* registration algorithm producing warp fields that are error-free on motion boundaries induced by dynamic scene topology.

We note that, throughout our discussion, we use the term ‘point cloud’ to refer to a geometry representation by a *discrete* point set sample of the underlying surface, as opposed to a volumetric 3D image. Our broad definition does not preclude additional per-point attributes. Therefore, oriented point clouds (point sets equipped with per-point normals) and surfel clouds (oriented point clouds with per-point radii) both fall within what we refer to simply as point cloud based representations.

**Contributions.** In this paper, we introduce a complete pipeline for the non-rigid registration of unorganized, oriented 3D point cloud pairs, which explicitly detects topology changes between the input point sets and produces piecewise-smooth warp fields that respect motion boundaries that result from these events. At the core of our approach lies a general warp field estimation algorithm (Section 3.3), inspired by those employed in recent dynamic reconstruction systems from RGB-D input. We improve motion estimation on motion boundaries associated with topology changes in an efficient post-processing phase (Section 3.4) that exploits the different properties of warp fields that are estimated in different directions (i.e. forward and backward) with respect to different types of topological events (i.e. ‘contact’ or ‘separation’, Section 3.2). After explicitly detecting regions of topology change events by means of simple, intuitive tests of local deformation, our method blends the forward and inverted backward motion hypotheses on a local basis, based on the type and proximity of detected events, ensuring smooth, seamless hypothesis transitions on the deformed surface. This stage makes no assumptions about the underlying registration engine and can be easily adapted for integration into existing pipelines. The implementation of our warp field estimation module (without the topology event handling) is openly available as part of our point cloud processing library [18]. Furthermore, the ability to detect and classify motion boundaries associated with dynamic topology is a byproduct of our pipeline that may be useful in tasks beyond geometric registration.

After discussing related work, we present our proposed method in detail in Section 3. In Section 4, we provide an extensive series of quantitative and qualitative experiments for the evaluation of our approach in terms of both registration accuracy and topological event handling.

## 2 RELATED WORK

**RGB-D scene flow estimation.** The term ‘scene flow’ was introduced in [19] to refer to “the three-dimensional motion field of points in the world, just as optical flow is the two-dimensional motion field of points in an image.” Since then, significant research focus has shifted towards scene flow estimation from RGB-D input. The formulation of [9] couples an  $L^1$ -norm data term derived from the *optical flow* and *range flow* [20] constraints with weighted TV regularization. In [10], the authors follow a similar variational approach but use a rigid motion parameterization of the flow field, computing 6DoF per-pixel transformations and enforcing a

local rigidity prior on the solution. A 6DoF local parameterization is also used in [21], which introduces a correspondence search mechanism that relies on 3D spheres rather than image plane patches, and effectively handles large displacements. In [22], a probabilistic approach for joint segmentation and motion estimation method is proposed; a depth-based segmentation is used for motion estimation, which is in turn regularized based on the mean rigid motion of each layer. A joint segmentation and scene flow estimation method is also presented in [11], which assumes that the scene movement can be described by a small number of latent rigid motions. Starting with a spatial  $k$ -means clustering for the motion label initialization, the algorithm iterates between motion estimation and segmentation (soft labeling), merging labels in the process. In [3], the first real-time RGB-D variational scene flow method is introduced, achieving state-of-the-art accuracy. An efficient joint odometry and piecewise-rigid scene flow estimation method is proposed in [4], where the scene is segmented into ‘static’ and ‘moving’ geometric clusters, from which odometry and independent non-rigid motions are computed.

As mentioned in our introduction, scene flow solves a somewhat restricted problem in the context of dense 3D registration, as the support of the computed motion field is image bound.

**Dynamic scene reconstruction.** General non-rigid 3D registration algorithms have been developed in the context of online reconstruction of dynamic scenes from RGB-D input. Most of them are formulated within a non-rigid Iterative Closest Point (ICP) framework, similar to the one introduced in [23], with the goal of registering a point cloud representation of the scene model to the current frame, while there also exist purely volumetric approaches [15] that align Signed Distance Field (SDF) geometry representations. DynamicFusion [7] was the first system to achieve high quality, real-time dense reconstructions from RGB-D input. While it performs volumetric (SDF) fusion [17], its warp field estimation algorithm is based on oriented point cloud renderings of the model geometry. The estimated warp field is defined on a sparse ‘Embedded Deformation’ (ED) graph [24], with a 6DoF transformation attached to each node, and its evaluation on arbitrary points is performed via interpolation. The registration objective consists of a point-to-plane ICP cost, coupled with an ‘As-Rigid-As-Possible’ (ARAP) [25], hierarchically defined regularization term, both under robust loss functions. The non-rigid tracker of VolumeDeform [8] does not rely on an ED graph and estimates individual 6DoF transformations for every source geometry point. Its cost function consists of a dense point-to-plane cost, a sparse point-to-point term derived from SIFT [26] correspondences, and an ARAP prior based on a ‘flat’ neighborhood graph, with all terms being quadratic. Fusion4D [13] combines the input of multiple range cameras for the task of dynamic reconstruction, using an ED warp field parameterization and following a similar registration objective formulation that additionally includes a ‘visual hull’ term. CoFusion [27] and MaskFusion [28] segment, using semantic and motion cues, and reconstruct multiple moving objects in real-time, assuming that every object moves rigidly. SurfelWarp [14] is a purely point (surfel) cloud based approach that also relies on an ED motion field representation and uses the

same registration costs as DynamicFusion, but under the quadratic loss function. On the other end of the spectrum, KillingFusion [15] and SobolevFusion [16] are purely volumetric approaches that rely on direct SDF-to-SDF alignment [29] via variational minimization under novel regularizers that enforce the motion field to be isometric and preserve level set geometry.

All of the above systems produce results of remarkable quality, especially given their real-time budget. However, with the exceptions of [13], [14], [15], and [16], they cannot handle scenes with dynamic topology, with the ‘close-to-open’ case (‘separation’, in our terminology) being particularly problematic. According to our introductory discussion, [13] and [14] deal with these cases essentially by discarding affected regions, while the volumetric registration approaches of [15] and [16] are inherently immune to these events. Our proposed method is the first to tackle dynamic topology within the context of *motion estimation* and within a scalable *point-based* representation framework.

### 3 OUR APPROACH

#### 3.1 Problem statement

Given a pair of unorganized 3D point sets, our goal is to estimate a warping function that non-rigidly deforms the first point cloud (source geometry) towards the second one (target) in a piece-wise smooth manner.

Let  $S = \{x_i^s\} \subset \mathbb{R}^3$  and  $D = \{x_i^d\} \subset \mathbb{R}^3$  be the source and target geometry point sets, respectively, and  $\mathcal{W} : \mathbb{R}^3 \mapsto \mathbb{R}^3$  be a warping function. In our non-rigid alignment setting,  $\mathcal{W}$  is required to have the following properties:

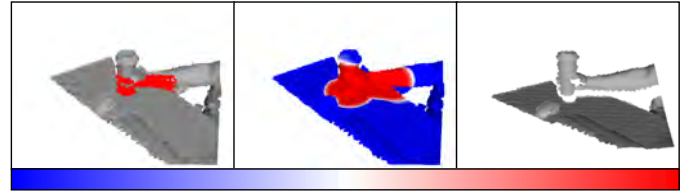
- The image of point set  $S$  via  $\mathcal{W}$ ,  $\mathcal{W}[S]$ , should be aligned as close to the target geometry  $D$  as possible. Typically, this is formulated as the minimization of the sum of residuals between points in  $\mathcal{W}[S]$  and their corresponding points (e.g., nearest neighbors) in  $D$ .
- Local transformations of neighboring points that lie on the same moving surface in  $S$  should be similar; i.e.  $\mathcal{W}$  should be *smooth*. At the same time, motion discontinuities should be preserved: neighboring points in  $S$  that lie on independently moving surfaces should be allowed to have different local transformations. This combined prior is known as *piecewise-smoothness*.

In a typical registration objective minimization formulation, the first property is expressed by the sum of registration residuals (e.g., point-to-point and/or point-to-plane distances) over corresponding point pairs in the objective (*data term*), while the second one renders the otherwise under-constrained problem well-posed by introducing terms that penalize differences in local transformations of neighboring points (*regularization term*).

The loss function used to model the regularization penalty terms, plays an important role in the behavior of the warping function in motion boundary regions. For example, it is well known from the optical flow literature that quadratic regularization tends to oversmooth motion boundaries. On the other hand, robust loss functions (e.g.,  $L^1$ -norm approximations for the penalty terms) are more effective in producing piecewise-smooth motion fields that preserve discontinuities.



(a) Left, middle: color frames (source and target) captured by an RGB-D camera. Right: warped source frame by the result of a standard non-rigid registration algorithm.



(b) Results of our proposed method. Left: detected (red) topological event regions (Section 3.4.1). Middle: blending weight  $w_b^i \in [0, 1]$  for the inverted backward hypothesis using a ‘blue-to-red’ colormap (blue: forward, red: inverted backward) shown at the bottom (Section 3.4.2). Right: source frame transformed by our topology-aware warp field.

Fig. 1. Non-rigid registration under a ‘close-to-open’ topology change.

In this work, we focus on estimating warp fields that respect motion boundaries resulting from changes in scene *topology*. Our notion of topology is directly derived from object-level connectivity: a change in scene topology can occur either when two or more separate objects come into contact or when two or more initially connected objects separate. We note that we use the term ‘object’ simply to refer to independently moving scene surface regions, without attaching to them any semantic meaning or assuming prior knowledge thereof.

In the following, we show that simply adopting a robust loss function for regularization still produces visible warping artifacts in motion discontinuity regions that result from scene topology changes, and we present a complete registration pipeline that effectively and efficiently solves this problem.

#### 3.2 Motivation and overview of our approach

Motion estimation errors on motion boundaries typically manifest as oversmoothing of the warp field because of excessive regularization and can be suppressed by eliminating regularization penalty terms for points in  $S$  that lie on different sides of the discontinuity. However, without any knowledge about  $S$  and its motion (e.g., some form of segmentation into independently moving objects), we cannot obtain a “correct” regularization graph a priori. Instead, the common choice is to use a  $k$ -NN graph of points in  $S$  to define the regularization terms. It is easy to see that this choice is particularly problematic in cases where connected objects in  $S$  move apart in  $D$ , as  $k$ -NN regularization over  $S$  will introduce penalty terms that relate points that lie on different objects, resulting in some amount of motion field oversmoothing over the separation boundary.

Such a challenging scenario that involves object ‘separations’ is depicted in Fig. 1a, where the general, topology-agnostic warp field estimation algorithm described in Section 3.3 was used to non-rigidly align two RGB-D frames.

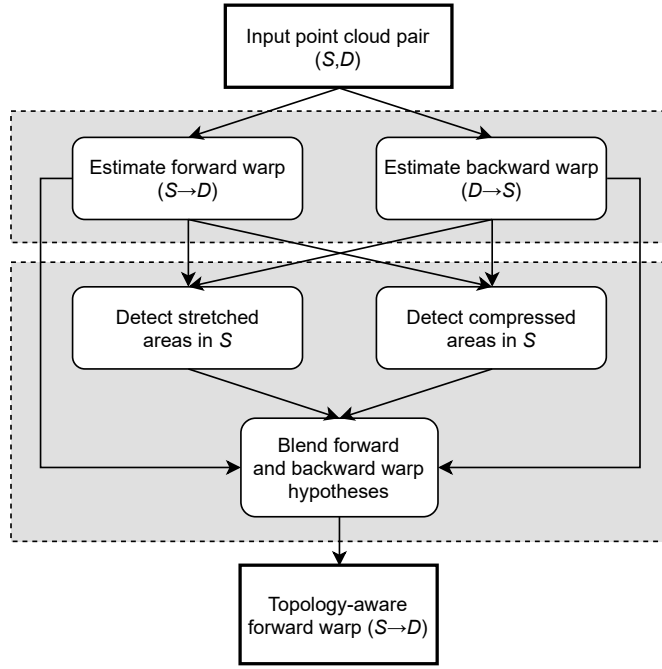


Fig. 2. Overview of our topology-aware non-rigid registration pipeline.

Despite the fact that the algorithm’s regularization term is formulated based on the robust, discontinuity-preserving Huber- $L^1$  loss function (see Section 4.1 for parameter details), the registration result (warped source geometry) shows visible artifacts near the object separation areas. Quadratic regularization is known to induce even more excessive smoothing on motion boundaries. Since quadratic and  $L^1$ -norm approximation regularization types are the most commonly used ones in the literature, most current non-rigid registration algorithms are expected to exhibit a very similar behavior on these types of motion boundaries.

At the same time, it is clear that any change in topology between  $S$  and  $D$  will be directly reflected on the (different) nearest-neighbor graph structures of the source and target geometries. We exploit this fact in the following way. Consider the case where two or more objects that are connected in  $S$  become separate in  $D$ . As discussed above, estimating the warp field  $\mathcal{W}_S^f$  that aligns  $S$  to  $D$  using the source geometry’s  $k$ -NN graph to define the regularization penalties is expected to result in some amount of oversmoothing over the motion boundary of the separation. However, in the *backward* motion direction (from  $D$  to  $S$ ), the same topology change manifests as a connection of separate objects. Estimating the backward warp field  $\mathcal{W}_D^b$  that aligns  $D$  to  $S$  using the target ( $D$ ) geometry’s  $k$ -NN graph to define the regularization penalties should not exhibit any oversmoothing over the connection boundary, because the corresponding  $k$ -NN graph edges that would define regularization terms over the discontinuity are not there in the first place. Inverting the warping function  $\mathcal{W}_D^b$  yields another *forward* warp field hypothesis,  $\mathcal{W}_S^b$ , that will be free of oversmoothing over separation motion boundaries. Of course, the latter, being derived from  $\mathcal{W}_D^b$ , may suffer from oversmoothing over motion boundaries that correspond to object separations in the backward motion direction (from

### Algorithm 1 $\mathcal{W} = \text{NONRIGIDICP}(D, S, \mathcal{W}_0)$

---

```

1:  $\mathcal{W} \leftarrow \mathcal{W}_0$ 
2: repeat
3:    $S' \leftarrow \mathcal{W}[S]$ 
4:    $\mathcal{C} \leftarrow \text{FINDCORRESPONDENCES}(D, S')$ 
5:    $\mathcal{W}_{\text{iter}} \leftarrow \text{OPTIMIZEWARPFIELD}(D, S', \mathcal{C})$ 
6:    $\mathcal{W} \leftarrow \mathcal{W}_{\text{iter}} \circ \mathcal{W}$ 
7: until  $\mathcal{W}_{\text{iter}}$  is close to the identity warp
  
```

---

$D$  to  $S$ ), or, equivalently, to objects coming into contact from  $S$  to  $D$ . These cases are expected to be handled correctly in the first place by the standard forward warp,  $\mathcal{W}_S^f$ .

Based on the above observations, the standard *forward* warp  $\mathcal{W}_S^f$  is expected to exhibit good behavior over *contact* boundaries, but to oversmooth *separation* boundaries. On the other hand, the *inverted backward* warp  $\mathcal{W}_S^b$  is expected to behave the opposite way, preserving *separation* discontinuities, but possibly blurring motion estimates in *contact* areas. Our proposed registration pipeline builds upon this idea by first detecting regions in  $S$  that are likely to be contact or separation boundaries, and then locally blending the warp hypotheses  $\mathcal{W}_S^f$  and  $\mathcal{W}_S^b$  accordingly in a seamless manner. The final result is a piecewise-smooth warp field that aligns  $S$  to  $D$  and respects motion boundaries because of changes in scene topology.

An overview of our approach is provided in Fig. 2. The (topology-agnostic) warp field estimation algorithm used to obtain the initial forward and backward warp hypotheses is described in detail in Section 3.3. Our topology event detection mechanism, as well as our local hypothesis blending approach, are presented in Section 3.4.

### 3.3 Warp field estimation

We implement our warp field estimation algorithm within a non-rigid Iterative Closest Point (ICP) framework [23], similarly to the non-rigid trackers used in the recently introduced dynamic reconstruction pipelines of [7], [8], and [14].

Given the source and target geometries  $S$  and  $D$ , represented as oriented point clouds, as well as an initial estimate  $\mathcal{W}_0$  of the unknown warp field  $\mathcal{W}$  (usually taken as the identity warp), the algorithm iteratively refines the latter until convergence has been reached. At the top level, the process iterates between a point correspondence search step between the warped source  $\mathcal{W}[S]$  (according to the current  $\mathcal{W}$  estimate) and  $D$ , and a warp field optimization step that updates  $\mathcal{W}$  given the established point correspondences (Algorithm 1). The two algorithm phases are presented in detail in the following.

#### 3.3.1 Correspondence association

Our framework supports two complementary types of point correspondences between the (warped) source and the target geometries: *dense* correspondences that are established based on spatial point proximity, and *sparse* correspondences that result from keypoint matching. Each individual correspondence is represented as a pair of point indices, whose first component indexes a point in  $S$  and

its second one a point in  $D$ :  $\mathcal{C} = \{\mathcal{C}_{\text{dense}}, \mathcal{C}_{\text{sparse}}\}$ , where  $\mathcal{C}_{\text{dense}}, \mathcal{C}_{\text{sparse}} \subseteq \{1, \dots, |S|\} \times \{1, \dots, |D|\}$ .

We support two mechanisms to establish dense correspondences. By default, we assume that both  $S$  and  $D$  are arbitrary, unorganized point sets and we establish dense correspondences by finding the nearest-neighbor in  $D$ , in terms of Euclidean distance, of each point in  $\mathcal{W}[S]$ , with the search being performed efficiently by parallel  $k$ d-tree queries. For certain applications that only require frame-to-frame (2.5D-to-2.5D) or model-to-frame (3D-to-2.5D) registration, we can further speed up the process by obtaining *projective* correspondences. This amounts to projecting  $S$  and  $D$  onto the target frame image and extracting correspondences based on points that are projected to the same pixel. This is the mechanism adopted in most real-time reconstruction pipelines [5], [7], [8].

In many common situations, dense geometric/depth correspondences alone are not enough to disambiguate the underlying motion. For example, tracking points on flat surfaces that lack geometric texture and slide parallel to each other may exhibit drift. Establishing robust keypoint correspondences between the source and target geometries can effectively mitigate this problem. We assume that our input geometries are equipped with sparse interest points; our sparse correspondences are established by the interest point descriptor matches between  $S$  and  $D$ . In our implementation, we focus on input geometries that are either RGB-D frames or 3D reconstructions from RGB-D input. The availability of regular images along with (registered) geometry allows us to adopt SIFT keypoint [26] (lifted to 3D) matches for our sparse correspondences.

To make optimization more stable, we discard correspondence candidates from the above mechanisms that do not meet some basic proximity and local similarity criteria. Let  $\{n_i^s\}$ ,  $\{n_i^d\}$ , and  $\{c_i^s\}$ ,  $\{c_i^d\}$  be the surface normals and colors (e.g., RGB value 3-vectors) of the source and target geometries, indexed in the same way as their support points in  $S$  and  $D$ . A correspondence candidate  $(i, j) \in \{1, \dots, |S|\} \times \{1, \dots, |D|\}$  is considered valid and used in the optimization if all of the following hold:

- $\|x_i^s - x_j^d\|_2 < \theta_d$
- $\arccos(n_i^s \cdot n_j^d) < \theta_n$
- $\|c_i^s - c_j^d\|_2 < \theta_c$

In the above,  $\theta_d$  is a point distance threshold,  $\theta_n$  is a normal angle threshold, and  $\theta_c$  is a color “distance” threshold.

### 3.3.2 Warp field optimization

Given a set of dense and sparse point correspondences, we shall now describe our warp field optimization step. Modeling the warp field using locally affine [23] or locally rigid [7] transformations provides better motion estimation results than adopting a simple translational local model, due to more effective regularization. In our approach, we adopt a locally rigid (6DoF) model.

Instead of computing a unique rigid transformation for each point in  $S$ , we use the more efficient embedded deformation graph representation [24] for the warp field  $\mathcal{W}$ , similarly to [7] and [13]. Let  $\mathcal{G} = \{(g_i, \sigma_i, T_i)\}$  be the set of virtual deformation nodes, where  $g_i \in \mathbb{R}^3$  is the position of the  $i$ th node,  $\sigma_i$  is a radius parameter that controls the  $i$ th node’s area of effect, and  $T_i \in SE(3)$  is the 6DoF rigid

transformation attached to the  $i$ th node. The deformation node positions are obtained by downsampling the source geometry  $S$  by means of a voxel grid of bin size  $r_b$ . This allows us to use a uniform radius parameter,  $\sigma_{\text{def}}$ , for all deformation nodes, so that  $\sigma_i = \sigma_{\text{def}}$ , for  $i = 1, \dots, |\mathcal{G}|$ . A reasonable choice that ensures sufficient area of effect overlap among neighboring nodes is  $\sigma_{\text{def}} = r_b/2$ . As in [8], each local transformation  $T_i$  is parameterized during optimization by a 6D vector  $\theta_i$  of 3 Euler angles and 3 translational offsets. The effect of the warp field  $\mathcal{W}$ , represented by  $\mathcal{G}$ , on a point  $x \in \mathbb{R}^3$  is given by interpolating the local node deformations in the neighborhood of  $x$ . Let  $\mathcal{N}(x) \subseteq \{1, \dots, |\mathcal{G}|\}$  be the indices of the  $k$ -nearest neighbors of  $x$  in  $\mathcal{G}$ . The local transformation parameter vector at  $x$  is given by:

$$\theta(x) \equiv \frac{\sum_{i \in \mathcal{N}(x)} w_i(x) \theta_i}{\sum_{i \in \mathcal{N}(x)} w_i(x)}, \quad (1)$$

where  $w_i(x) = \exp(-\|x - g_i\|_2^2 / (2\sigma_i^2))$ . The image of  $x$  via  $\mathcal{W}$  is then:

$$\mathcal{W}(x) \equiv \text{Rot}(\theta(x))x + \text{Trans}(\theta(x)), \quad (2)$$

where  $\text{Rot}(\theta)$  and  $\text{Trans}(\theta)$  extract the rotation matrix and translation vector from our 6D parameterization.

We note that the above 6D parameterization is only used within optimization (line 5 of Algorithm 1) and that both the estimated *incremental* warp  $\mathcal{W}_{\text{iter}}$  and the final composite estimate  $\mathcal{W}$  have their node transformations  $T_i$  expressed in terms of  $SE(3)$  transformation matrices. The fact that we continuously warp  $S$  and compute  $\mathcal{W}_{\text{iter}}$  starting from the identity warp, combined with the smoothness prior imposed on the warp field (shown below), allows us to overcome any problems associated with Euler angle parameterizations of rotation.

Our registration objective, as a function of the unknown warp field  $\mathcal{W}$ , which in the context of Algorithm 1 is the incremental warp  $\mathcal{W}_{\text{iter}}$ , and the point correspondences  $\mathcal{C}$  between  $S$  and  $D$ , which are fixed for this step, is formulated as:

$$E(D, S, \mathcal{C}, \mathcal{W}) = E_{\text{data}}(D, S, \mathcal{C}, \mathcal{W}) + \lambda_{\text{stiff}} E_{\text{stiff}}(\mathcal{W}). \quad (3)$$

Our data term,  $E_{\text{data}}(D, S, \mathcal{C}, \mathcal{W})$ , is a weighted sum of a point-to-plane and a point-to-point ICP cost:

$$E_{\text{data}}(D, S, \mathcal{C}, \mathcal{W}) = E_{\text{plane}}(D, S, \mathcal{C}, \mathcal{W}) + \lambda_{\text{point}} E_{\text{point}}(D, S, \mathcal{C}, \mathcal{W}). \quad (4)$$

Pure point-to-plane metric optimization generally converges faster and to better solutions than pure point-to-point [30], and is the standard trend in recent rigid [5], [6] and non-rigid [7], [8], [14] registration pipelines. However, as discussed in Section 3.3.1, integrating a point-to-point term for robust point matches into the registration cost can effectively disambiguate motion estimation in cases where surfaces that lack geometric texture slide parallel to each other. Similarly to [8], we use our dense geometric correspondences  $\mathcal{C}_{\text{dense}}$  to define our point-to-plane cost and

our sparse keypoint correspondences  $\mathcal{C}_{\text{sparse}}$  for our point-to-point cost:

$$E_{\text{plane}}(D, S, \mathcal{C}, \mathcal{W}) = \sum_{(i,j) \in \mathcal{C}_{\text{dense}}} \left( n_j^d \left( \mathcal{W}(x_i^s) - x_j^d \right) \right)^2, \quad (5)$$

$$E_{\text{point}}(D, S, \mathcal{C}, \mathcal{W}) = \sum_{(i,j) \in \mathcal{C}_{\text{sparse}}} \left\| \mathcal{W}(x_i^s) - x_j^d \right\|_2^2. \quad (6)$$

Our regularization term  $E_{\text{stiff}}(\mathcal{W})$  directly penalizes differences in transformation parameters of neighboring virtual nodes of  $\mathcal{G}$  under the robust Huber- $L^1$  loss function. If  $\mathcal{N}(i) \subseteq \{1, \dots, |\mathcal{G}|\}$  is the set of indices of the  $k$ -nearest neighbors of  $g_i$  in  $\mathcal{G}$ , our regularization term is formulated as:

$$E_{\text{stiff}}(\mathcal{W}) = \sum_{i=1}^{|\mathcal{G}|} \sum_{j \in \mathcal{N}(i)} w_{ij} \psi_{\delta}(\theta_i - \theta_j), \quad (7)$$

where  $w_{ij} = \exp\left(-\|g_i - g_j\|^2 / (2\sigma_{\text{def}}^2)\right)$  weights the pairwise penalties based on node distance, and  $\psi_{\delta}(\Delta\theta)$  denotes the sum of the Huber loss function values over the 6 parameter residual components. Parameter  $\delta$  controls the point at which the loss function behavior switches from quadratic (squared  $L^2$ -norm) to absolute-linear ( $L^1$ -norm). Since  $L^1$ -norm regularization is known to better preserve solution discontinuities, we use a small value of  $\delta = 10^{-4}$ . Given our locally rigid motion model, this regularization scheme enforces an ‘As-Rigid-As-Possible’ [25] prior to the estimated warp field.

The registration objective of equation (3) is non-linear in the  $6|\mathcal{G}|$  unknowns. We minimize  $E(D, S, \mathcal{C}, \mathcal{W})$  by performing a small number of Gauss-Newton iterations. As in [13], we handle non-quadratic terms using the square-rooting technique of [31]. At every step, we linearize  $E$  around the current solution  $\theta \in \mathbb{R}^{6|\mathcal{G}|}$  (vector concatenation of all node transformation parameters  $\theta_i$ ) and obtain a solution increment  $\hat{\theta}$  by solving the system of normal equations  $J^T J \hat{\theta} = J^T r$ , where  $J$  is the Jacobian matrix of the residual terms in  $E$  and  $r$  is the vector of (negative) residual values. We solve this sparse system iteratively, using the Conjugate Gradient algorithm with a diagonal preconditioner.

### 3.4 Handling topology changes

In our post-processing phase for handling topology change motion boundaries, we first explicitly detect likely *contact* and *separation* regions in the source geometry, and proceed by appropriately blending our (default) *forward* and *inverted backward* warp hypotheses in a local yet seamless manner. It follows from the discussion in Section 3.2 that, as far as topology changes are concerned, the forward warp is only problematic in separation areas. However, instead of only focusing on and amending separations, it is beneficial to also explicitly consider contact events. As will become clear in the following, considering both types of events and treating them symmetrically robustifies their detection and allows for a less biased hypothesis blending scheme.

Using the same notation as in Section 3.2, let  $\mathcal{W}_S^f$  be the warp field that aligns  $S$  to  $D$  and  $\mathcal{W}_D^b$  the backward warp that aligns  $D$  to  $S$ , both computed by Algorithm 1. We will be using the ‘ $S$ ’ subscript for forward ( $S \rightarrow D$ ) motion

#### Algorithm 2 EXTRACTTOPOLOGYEVENTS

---

**Input:**  $S$ ,  $\text{STRETCH}_S^f$ ,  $\text{STRETCH}_S^b$ ,  $\text{COMPRESS}_S^f$ ,  $\text{COMPRESS}_S^b$   
**Output:** CON, SEP

---

```

1: CON  $\leftarrow \emptyset$ 
2: SEP  $\leftarrow \emptyset$ 
3: for  $i = 1, \dots, |S|$  do
4:   stretch  $\leftarrow \max\{\text{STRETCH}_S^f(i), \text{STRETCH}_S^b(i)\}$ 
5:   compress  $\leftarrow \max\{\text{COMPRESS}_S^f(i), \text{COMPRESS}_S^b(i)\}$ 
6:   if stretch  $> \tau$  and stretch  $> \alpha \cdot \text{compress}$  then
7:     SEP  $\leftarrow \text{SEP} \cup \{x_i^s\}$ 
8:   end if
9:   if compress  $> \tau$  and compress  $> \alpha \cdot \text{stretch}$  then
10:    CON  $\leftarrow \text{CON} \cup \{x_i^s\}$ 
11:   end if
12: end for
```

---

entities and the ‘ $D$ ’ subscript for backward ( $D \rightarrow S$ ) ones. For the needs of the following discussion, we will consider these motion fields to be represented by the *per-point* local rigid transformations of their support geometries, so that:

$$\mathcal{W}_S^f = \{T_{S_i}^f, i = 1, \dots, |S|\}, \quad (8)$$

$$\mathcal{W}_D^b = \{T_{D_i}^b, i = 1, \dots, |D|\}. \quad (9)$$

To invert  $\mathcal{W}_D^b$  in order to obtain an alternative forward warp, we appropriately rebase its inverse local transformations on  $S$ . To that end, we first compute the target geometry’s image  $\mathcal{W}_D^b[D]$  (which should be closely aligned to  $S$ ) and, to each point  $x_i^s \in S$ , we assign the transformation  $T_{D_j}^{b-1}$ , where  $j$  is the index of the nearest neighbor of  $x_i^s$  in the point set  $\mathcal{W}_D^b[D]$ . Of course, we assume that the latter is indexed in the same way as  $D$ . The inverted backward warp is then represented as:

$$\mathcal{W}_S^b = \{T_{S_i}^b, i = 1, \dots, |S|\}, \quad (10)$$

where  $T_{S_i}^b = T_{D_j}^{b-1}$  and  $j$  is the nearest neighbor index of  $x_i^s$  in  $\mathcal{W}_D^b[D]$ . Analogously, we obtain an alternative backward warp, by inverting our forward hypothesis:

$$\mathcal{W}_D^f = \{T_{D_i}^f, i = 1, \dots, |D|\}, \quad (11)$$

where  $T_{D_i}^f = T_{S_j}^{f-1}$  and  $j$  is the nearest neighbor index of  $x_i^d$  in  $\mathcal{W}_S^f[S]$ . To summarize, we have two forward motion ( $S \rightarrow D$ ) warp hypotheses ( $\mathcal{W}_S^f$  and  $\mathcal{W}_S^b$ ) and two backward motion ( $D \rightarrow S$ ) ones ( $\mathcal{W}_D^b$  and  $\mathcal{W}_D^f$ ).

#### 3.4.1 Detecting topology change events

We detect topology change regions in  $S$  based on how our warp estimates affect local neighborhoods of the source geometry.

Naturally, we expect that if  $x_i^s \in S$  is close to a separation boundary, its distance to some of its neighbors in  $S$  should increase after applying the correct warp to  $S$ . We shall refer to this effect as neighborhood *stretching*. The dual case of a contact event manifests exactly the same way in the backward motion direction (stretching of neighborhoods of  $D$ ), in which the event is perceived as a separation. In



the following, we will use a local measure of stretch over points in  $S$  to detect separation areas, and map the same measure over  $D$  in the backward direction onto  $S$  to obtain a dual measure of “compression” that will allow us to detect contacts.

We quantify the above intuition by defining a local “stretch” operator for point  $x_i \in X \subseteq \mathbb{R}^3$  under the warp  $\mathcal{W}$  as the maximum ratio of the distance to its neighbors before and after applying  $\mathcal{W}$ :

$$\text{Stretch}(i, X, \mathcal{W}) \equiv \max_{j \in \mathcal{N}(i)} \frac{\|\mathcal{W}(x_i) - \mathcal{W}(x_j)\|_2}{\|x_i - x_j\|_2}, \quad (12)$$

where  $\mathcal{N}(i) \subseteq \{1, \dots, |X|\}$  indexes the neighbors of  $x_i$  in  $X$  that lie within  $\rho_s$  distance from it. The choice of the neighborhood radius value  $\rho_s$  depends on the scale and resolution of the input geometries. For close-range point clouds acquired with Kinect-like cameras, we use  $\rho_s = 1.5\text{cm}$ .

To each point in  $S$ , we associate one stretch value for each of our two forward warp hypotheses, according to definition (12). For  $i = 1, \dots, |S|$ , we have:

$$\text{STRETCH}_S^f(i) = \text{Stretch}(i, S, \mathcal{W}_S^f), \text{ and} \quad (13)$$

$$\text{STRETCH}_S^b(i) = \text{Stretch}(i, S, \mathcal{W}_S^b). \quad (14)$$

We also compute the local stretch of the target geometry  $D$  under each of the two backward warps:

$$\text{STRETCH}_D^f(i) = \text{Stretch}(i, D, \mathcal{W}_D^f), \text{ and} \quad (15)$$

$$\text{STRETCH}_D^b(i) = \text{Stretch}(i, D, \mathcal{W}_D^b), \quad (16)$$

which we subsequently map onto  $S$ , interpreting them as a compression measure (contact indicator), according to:

$$\text{COMPRESS}_S^f(i) = \text{STRETCH}_D^f(\text{NN}(\mathcal{W}_S^f(x_i^s), D)), \quad (17)$$

$$\text{COMPRESS}_S^b(i) = \text{STRETCH}_D^b(\text{NN}(\mathcal{W}_S^b(x_i^s), D)), \quad (18)$$

where  $\text{NN}(x, X) \in \{1, \dots, |X|\}$  is the index of the nearest neighbor of point  $x$  in point set  $X$ .

Using the above point-wise stretch/compress values on  $S$ , we extract subsets of the source geometry that are likely to lie on topology change motion boundaries. Let  $\text{SEP}, \text{CON} \subseteq S$  be the sets of candidate separation and contact boundary points, respectively. According to the above discussion, points on a separation boundary are expected to have high stretch scores, while points on a contact boundary should exhibit high local compression. To decide whether a point in  $S$  is a boundary candidate, we perform two symmetric tests per case that rely on two threshold values, an absolute score threshold  $\tau$ , and a relative (ratio) threshold  $\alpha$ . A point of  $S$  is a member of  $\text{SEP}$  ( $\text{CON}$ ) if and only if the maximum of its two stretch (compress) scores is greater than  $\tau$  and also greater than  $\alpha$  times its maximum compress (stretch) score. The process is summarized in Algorithm 2. A sample output is shown in Fig. 1b (left), marked in red; note that  $\text{CON} = \emptyset$  in this case.

As we will show in Section 4.3, the above procedure is very effective at detecting and classifying topology changes, but, because of the continuous nature of our local stretch/compression measures and depending on the selected threshold values, it may produce “false positives”

### Algorithm 3 LOCALHYPOTHESISBLENDING

**Input:**  $S, \mathcal{W}_S^f, \mathcal{W}_S^b, \text{CON} = \{c_i\}, \text{SEP} = \{s_i\}, \rho_e$

**Output:**  $\mathcal{W}_S = \{T_{Si}, i = 1, \dots, |S|\}$

```

1:  $\sigma \leftarrow \rho_e/3$ 
2: for  $i = 1, \dots, |S|$  do
3:    $\mathcal{N} \leftarrow \text{RADIUSSEARCH}(x_i^s, \text{CON}, \rho_e)$ 
4:    $w_f \leftarrow 1$ 
5:   for  $j \in \mathcal{N}$  do
6:      $w_f \leftarrow w_f + \exp(-\|x_i^s - c_j\|^2 / (2\sigma^2))$ 
7:   end for
8:    $\mathcal{N} \leftarrow \text{RADIUSSEARCH}(x_i^s, \text{SEP}, \rho_e)$ 
9:    $w_b \leftarrow 0$ 
10:  for  $j \in \mathcal{N}$  do
11:     $w_b \leftarrow w_b + \exp(-\|x_i^s - s_j\|^2 / (2\sigma^2))$ 
12:  end for
13:   $w \leftarrow w_f + w_b$ 
14:   $w_f \leftarrow w_f/w$ 
15:   $w_b \leftarrow w_b/w$ 
16:   $T_{Si} \leftarrow \text{SE3}(w_f T_{Si}^f + w_b T_{Si}^b)$ 
17: end for
```

(e.g., in areas of actual deforming surface stretching or compression but constant topology). However, under the assumption that our two forward warp hypotheses behave similarly in the false positive areas and, as will become clear in the next section, this does not affect our final warp estimate.

#### 3.4.2 Local hypothesis blending

Our blending scheme produces a topology-aware warp field  $\mathcal{W}_S$  by combining the forward warp hypotheses  $\mathcal{W}_S^f$  and  $\mathcal{W}_S^b$  on a per-point basis. Our objective is to assign a higher weight to  $\mathcal{W}_S^b$  (inverted backward warp) near separation areas, and ensure that  $\mathcal{W}_S^f$  (forward warp) has a stronger weight near contact areas. At the same time, it is desirable that point weights vary smoothly on  $S$ , so that our warp blending does not introduce seam artifacts on  $\mathcal{W}_S[S]$  due to differences in our original warp hypotheses.

We achieve the above by attaching a smoothly decaying kernel on each of our detected event points in  $\text{CON}$  and  $\text{SEP}$  and locally computing the weight for each event class. Assuming a maximum radius of effect  $\rho_e$  (free parameter) for our event points, we model the influence of each event with an RBF kernel of bandwidth  $\sigma = \rho_e/3$ . The weights  $w_f^i$  and  $w_b^i$  of  $\mathcal{W}_S^f$  and  $\mathcal{W}_S^b$  for the source point  $x_i^s$  are computed by accumulating influences of the event points in  $\text{CON} = \{c_i\}$  and  $\text{SEP} = \{s_i\}$  respectively that are within a  $\rho_e$ -radius from  $x_i^s$ :

$$w_f^i = \frac{1}{Z} \left( 1 + \sum_{j \in \mathcal{N}_C(i)} \exp(-\|x_i^s - c_j\|^2 / (2\sigma^2)) \right), \text{ and} \quad (19)$$

$$w_b^i = \frac{1}{Z} \left( \sum_{j \in \mathcal{N}_S(i)} \exp(-\|x_i^s - s_j\|^2 / (2\sigma^2)) \right), \quad (20)$$

where  $Z$  is a normalizing constant ensuring that  $w_f^i + w_b^i = 1$ , and  $\mathcal{N}_C(i), \mathcal{N}_S(i)$  index the  $\rho_e$ -radius neighbors of  $x_i^s$  in  $\text{CON}$  and  $\text{SEP}$  respectively. In the absence of any topology

TABLE 1  
Evaluation on the MPI Sintel Dataset

Sequence	Frame-wise EPE over entire sequence						Frame-wise AE over entire sequence					
	PD-Flow		F-Warp		FB-Warp		PD-Flow		F-Warp		FB-Warp	
	Median	Mean	Median	Mean	Median	Mean	Median	Mean	Median	Mean	Median	Mean
alley_1	0.774	0.964	0.485	0.519	<b>0.485</b>	<b>0.516</b>	<b>7.005</b>	<b>7.319</b>	8.256	8.379	8.285	8.361
ambush_5	12.651	21.172	1.714	7.017	<b>1.639</b>	<b>6.411</b>	30.435	29.505	5.558	6.920	<b>5.262</b>	<b>6.386</b>
ambush_6	25.467	28.981	<b>4.409</b>	<b>7.535</b>	5.106	8.028	37.813	39.470	7.165	7.520	<b>7.083</b>	<b>7.457</b>
ambush_7	1.861	4.969	0.633	0.925	<b>0.584</b>	<b>0.891</b>	11.517	27.146	11.377	10.885	<b>11.270</b>	<b>10.763</b>
bamboo_1	1.104	1.221	<b>0.677</b>	<b>0.712</b>	0.677	0.713	8.159	8.560	<b>5.517</b>	<b>5.705</b>	5.517	5.713
bamboo_2	0.889	2.628	0.687	1.027	<b>0.686</b>	<b>1.014</b>	<b>7.134</b>	11.924	8.315	8.729	8.357	<b>8.669</b>
bandage_1	1.683	2.160	0.973	0.957	<b>0.936</b>	<b>0.935</b>	15.201	15.244	10.111	9.829	<b>9.862</b>	<b>9.681</b>
bandage_2	0.769	1.062	<b>0.329</b>	0.334	0.330	<b>0.331</b>	9.035	10.754	6.525	6.554	<b>6.523</b>	<b>6.542</b>
shaman_2	<b>0.286</b>	0.317	0.294	0.284	0.292	<b>0.284</b>	<b>5.571</b>	<b>5.643</b>	7.801	7.838	7.779	7.831
shaman_3	0.485	0.552	<b>0.178</b>	<b>0.291</b>	0.179	0.298	8.125	8.115	3.155	<b>4.559</b>	<b>3.153</b>	4.604
sleeping_1	0.502	<b>0.514</b>	0.149	0.541	<b>0.149</b>	0.541	5.939	6.110	1.954	<b>6.487</b>	<b>1.954</b>	6.488
sleeping_2	<b>0.146</b>	<b>0.143</b>	0.170	0.171	0.170	0.171	<b>2.508</b>	<b>2.520</b>	3.352	3.335	3.352	3.335
<b>Overall</b>	0.701	4.122	0.488	1.379	<b>0.487</b>	<b>1.336</b>	7.899	13.009	6.826	7.213	<b>6.815</b>	<b>7.136</b>

event influence (e.g., for  $x_i^s$  at least  $\rho_e$  from any event point), the above defaults to  $w_f^i = 1$  and  $w_b^i = 0$ , giving full weight to the standard forward hypothesis  $\mathcal{W}_S^f$ . The local transformation of our final, topology-aware warp field estimate  $\mathcal{W}_S = \{T_{S_i}, i = 1, \dots, |S|\}$  at location  $x_i^s$  is then given by:

$$T_{S_i} = \text{SE3} \left( w_f^i T_{S_i}^f + w_b^i T_{S_i}^b \right), \quad (21)$$

where  $\text{SE3}(\cdot)$  converts the linear blend of the two transformation matrices back to a valid  $SE(3)$  transformation matrix. The complete blending process is summarized in Algorithm 3 (see also equations (8) and (10)). A visualization of the blending weights is given in Fig. 1b (middle), where each source geometry point is colored according to its inverted backward warp hypothesis weight  $w_b^i$ .

We note that, for source points close to topology events, one of  $w_f^i$  and  $w_b^i$  will dominate the other, effectively rendering the blending of (21) a binary selection. As we move farther from topology events, it is possible that the two weights assume comparable values (e.g., at points lying between two events of different type). For our blended output (21) to be seamless and error-free in that case, it is expected that  $\mathcal{W}_S^f$  and  $\mathcal{W}_S^b$  do not differ significantly in areas that are “far enough” from event points. This highlights the importance of parameter  $\rho_e$ , which should be of adequate magnitude to cover event regions; we have found that values  $\rho_e \geq 3r_b$  work well in practice, where  $r_b$  is the resolution of our virtual deformation graph (Section 3.3.2).

As a concluding remark, we observe that the most costly operation of our post-processing phase is the calculation, based on radius-neighborhoods, of the stretch/compress values of Section 3.4.1. Algorithm 3 also performs radius queries on point sets CON and SEP, but the latter are typically very small in size. In our experience, the overall running time of the entire phase is significantly smaller than a single warp field estimation. Furthermore, in the case of RGB-D input, image structure can be easily exploited in order to accelerate the extraction of point neighborhoods.

## 4 EXPERIMENTS

We conduct three sets of experiments for the evaluation of our registration pipeline. The first one is performed

on a public optical flow evaluation dataset and examines our algorithm’s motion estimation accuracy, both with and without the topology handling phase (Section 4.2). For the second one, we use a custom dataset with topology event annotations and evaluate our event detection performance, as well as our estimated warp field quality in the presence of separation events (Section 4.3). In our third set, we provide qualitative results in long-term model-to-frame registration scenarios on public RGB-D sequences that feature dynamic scene topology (Section 4.4).

### 4.1 General setup details

The input to all series of experiments is RGB-D data, either synthetic (first set) or captured by a Kinect-like RGB-D camera (second and third sets).

**Point cloud generation.** RGB-D frames are converted to point clouds equipped with surface normal and color information, as well as a sparse set of interest points derived from SIFT features. In all cases, the full resolution of the input depth map is used, which is  $1024 \times 436$  for the synthetic sequences and  $640 \times 480$  for the camera data. We use a fixed maximum depth of 5m for all sequences in the first set, and vary the cut-off value in the range of 0.8m to 2.0m for the camera data, depending on the sequence. For normal estimation, we use  $k$ -NN neighborhoods with  $k = 30$  in our first set of experiments, and  $\rho$ -radius neighborhoods with  $\rho = 1.5\text{cm}$  in our second and third sets. SIFT keypoints are extracted from the RGB images and lifted to 3D, discarding the ones that lie on depth boundaries.

**Warp field estimation.** In the correspondence association step (Section 3.3.1) of our non-rigid ICP algorithm, we set the maximum correspondence distance to  $\theta_d = 15\text{cm}$  for our first set of experiments and  $\theta_d = 5\text{cm}$  for the other sets, while we use common values  $\theta_n = 15^\circ$  and  $\theta_c = 0.4$  for the maximum normal angle and color difference (colors are RGB triplets in  $[0, 1]^3$ ). The embedded deformation graph  $\mathcal{G}$  for our warp field parameterization (Section 3.3.2) has a resolution of  $r_b = 2.5\text{cm}$ , with each node’s area of effect being controlled by  $\sigma_{\text{def}} = r_b/2$ . To evaluate the local transformation for each point in the source geometry (equation (1)), we use its 4 nearest neighbors in  $\mathcal{G}$ . The point-to-point



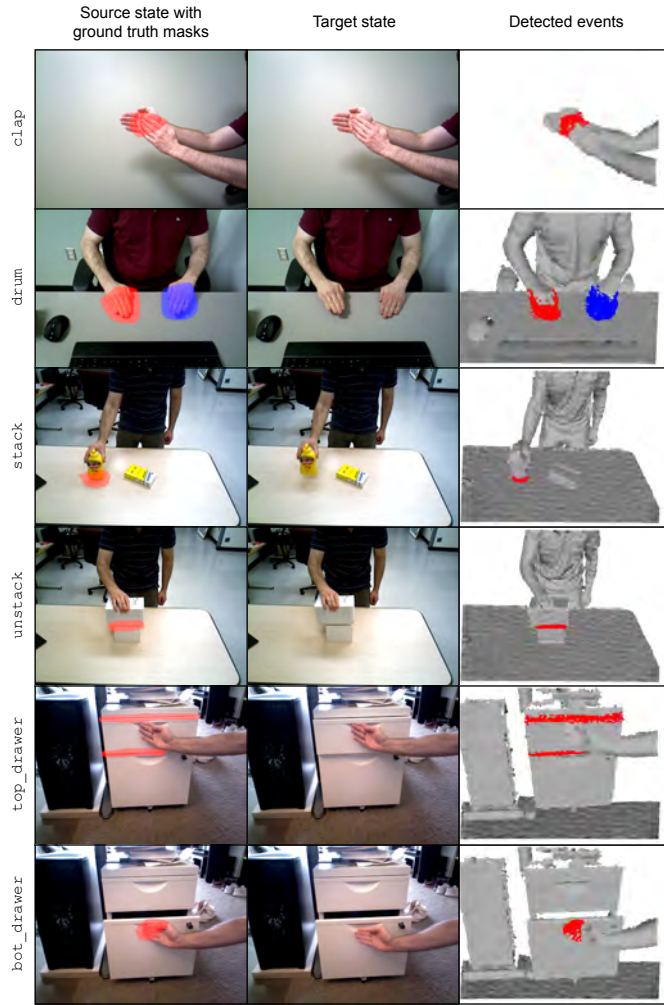


Fig. 3. Topological event detections on our dataset. Each row corresponds to a different sequence. First column: source color frame with event mask overlays (blue for contact, red for separation). Second column: target color frame. Third column: our topological event detections overlaid on the source geometry (blue for contact, red for separation).

weight in our data term (4) is set to  $\lambda_{\text{point}} = 2$ . To favor  $L^1$ -norm behavior by our regularization term (7), we use a small Huber loss parameter value of  $\delta = 10^{-4}$ , while we set the term's weight to  $\lambda_{\text{stiff}} = 200$  (equation (3)). Regularization topology is given by the 6 nearest neighbor nodes of  $\mathcal{G}$ . We perform a maximum of 10 top-level ICP iterations, while the process typically converges in less. Within each optimization step, we perform a maximum of 5 Gauss-Newton iterations.

**Dynamic topology handling.** In all experiments, local stretch is computed on neighborhoods of radius  $\rho_s = 1.5\text{cm}$  (Section 3.4.1). To detect and classify topology change events, we use an absolute score threshold of  $\tau = 2.2$  and a relative ratio of  $\alpha = 1.5$ . For our blending step (Section 3.4.2), we assume that every detected event has a radius of effect equal to  $\rho_e = 7.5\text{cm}$ .

## 4.2 Motion estimation accuracy evaluation

Due to the lack of publicly available datasets with ground truth dense 3D motion, we perform our accuracy assessments on MPI Sintel [32], a synthetic optical flow evaluation dataset. The dataset contains multiple sequences of (typically) 50 frames that capture motions ranging from slow,

almost rigid to very large, highly non-rigid ones. In addition to ground truth optical flow, metric ground truth depth and camera intrinsics are provided, which we use to emulate RGB-D input.

We base our evaluation on two classical optical flow performance measures, the endpoint error (EPE) and the angular error (AE) [33]. If  $\tilde{f} = (\tilde{u}, \tilde{v})$  is an optical flow estimate at a given pixel whose ground truth value is  $f = (u, v)$ , EPE is computed as:

$$e_{\text{EPE}}(\tilde{f}, f) = \|\tilde{f} - f\|_2. \quad (22)$$

The angular error AE is defined as the angle between the 3D space-time vectors  $h(\tilde{f}) = (\tilde{u}, \tilde{v}, 1)$  and  $h(f) = (u, v, 1)$ , as:

$$e_{\text{AE}}(\tilde{f}, f) = \arccos \frac{h(\tilde{f})^\top h(f)}{\|h(\tilde{f})\|_2 \|h(f)\|_2}, \quad (23)$$

effectively enabling evaluation at pixels of zero flow. We convert 3D motion estimates to optical flow by first warping the source points in 3D and then computing the 2D point/pixel displacements as differences of the projected endpoints onto the image plane.

We evaluate and compare three methods: PD-Flow [3], a state-of-the-art scene flow algorithm, F-Warp, our general warp field estimation algorithm defined in Section 3.3 (without the topology change handling phase), and FB-Warp, our complete topology-aware warp field estimation pipeline. We run the three algorithms on the entire duration of 12 Sintel sequences and compute the average EPE and AE values per consecutive frame pair. We report the median and mean frame-level average errors over each sequence in Table 1. Median values are not easily affected by extreme values, often providing a better picture of how accurate estimation is “half of the time”. We also report overall mean and median error values for each method, computed over the total number of frames from all sequences.

Our FB-Warp method overall achieves the highest accuracy in terms of both error metrics, followed closely by our baseline, F-Warp. PD-Flow is very accurate in estimating slow motions (e.g., in the `sleeping_2` sequence), but falls behind in most cases, producing particularly large errors in sequences that contain very fast motions, such as `ambush_5` and `ambush_6`. We also refer the reader to the Sintel-based evaluation of MC-Flow [11] (Table 2 of that paper), another state-of-the-art scene flow algorithm with significantly better performance than PD-Flow in estimating large motions. We were unable to evaluate MC-Flow ourselves, because its implementation has not been released. While 6 of the Sintel sequences in our experiments and the ones in [11] are common, there are significant differences between our evaluation setups: 1) their reported EPE and AE values are computed on one specific frame pair per sequence, not over whole sequences as in here, 2) they downsample their input to half its original resolution per dimension (to  $512 \times 218$ ), whereas we use full resolution images, and 3) they consider only non-occluded pixels, while we compute errors on all valid ones. The above prohibit reaching definitive conclusions. However, the fact that we adopt an arguably more difficult evaluation strategy (at *full resolution*, which means smaller pixel size for EPE interpretation, and evaluating over *whole sequences*) and still obtain comparable EPE and

TABLE 2  
Topology Change Event Detection

Sequence	Number of events		Ground truth to detected mapping GTToDET			Detected to ground truth mapping DETToGT		
	Ground truth	Detected	Matched fraction %	Mean overlap	Mean delay	Matched fraction %	Mean overlap	Mean delay
clap	11	32	100.00	0.832	1.000	78.12	0.633	0.800
drum	21	63	100.00	0.763	-0.190	76.19	0.616	-0.042
stack	4	15	100.00	0.820	-0.250	40.00	0.713	0.000
unstack	4	20	100.00	0.689	-0.250	40.00	0.527	0.000
separate	5	17	100.00	0.661	-0.400	70.59	0.527	-0.583
top_drawer	4	11	100.00	0.951	0.000	54.55	0.819	-0.667
bot_drawer	3	12	100.00	0.882	0.000	66.67	0.757	-0.375
<b>Overall:</b>	52	170	100.00	0.788	0.058	66.47	0.630	0.035

AE absolute values to the ones reported in [11] (averages of 1.203 and 6.559, respectively), leads us to argue that FB-Warp compares favorably to MC-Flow.

### 4.3 Dynamic topology event handling

To evaluate the performance of the topology-handling phase of our pipeline, we collected, using a Kinect-like camera, an RGB-D dataset of 7 sequences that contain changes in scene topology. The regions of visible topology changes were manually annotated on the color image as binary masks, drawn in freehand mode, and classified as either contacts or separations. Our collected sequences capture the following diverse set of actions:

- Hand-clapping (clap sequence)
- Fast hand-drumming on a desk (drum sequence)
- Two pick-and-place actions on objects lying on a flat surface or on top of each other (stack and unstack sequences)
- A separation of two touching objects using both hands (separate sequence)
- Two drawer opening action sequences (top\_drawer and bot\_drawer)

The annotated ground truth events capture all visible instances of hand-hand, hand-object, and object-object interaction. Snapshots of our sequences are shown in Fig. 3 (first two columns).

Our evaluation of this phase is twofold. First, we assess how well our detected event points in CON and SEP (Section 3.4.1) relate to the ground truth events. Second, we qualitatively and quantitatively evaluate our topology-aware motion estimate (FB-Warp), including comparisons with our baseline algorithm (F-Warp), as well as two scene flow estimation methods.

#### 4.3.1 Topology event detection

We uniformly represent topology change events as triplets  $e_i = (l_i, t_i, X_i)$ , where  $l_i \in \{0, 1\}$  is a binary label indicating contact or separation,  $t_i \in \mathbb{N}$  is the time (frame index) of the event, and  $X_i \subset \mathbb{R}^3$  is the subset of points in the source geometry that lie very close to event motion boundaries. We use the superscript ‘gt’ to denote ground truth event entities, and ‘det’ to denote the ones associated with detections by our algorithm. Let  $\mathcal{E}^{\text{gt}} = \{e_i^{\text{gt}}\}$  and  $\mathcal{E}^{\text{det}} = \{e_i^{\text{det}}\}$  denote the sets of ground truth and detected events, respectively. Given

an annotated sequence, these are populated according to the following:

- Labels and timestamps for events in  $\mathcal{E}^{\text{gt}}$  come directly from the annotation data. Ground truth event point clouds  $X_i^{\text{gt}}$  are obtained by the image annotation binary masks, which directly mask regions of the  $t_i$ -th frame’s point cloud (input color and depth maps are registered).
- Detected events  $\mathcal{E}^{\text{det}}$  are derived from the per-frame outputs of Algorithm 2. At time  $k$  (frame pair index), we interpret the *connected components*, in the Euclidean sense, of point sets CON and SEP as separate, meaningful events. We insert all triplets  $(0, k, \text{CON}_i)$  and  $(1, k, \text{SEP}_i)$  into  $\mathcal{E}^{\text{det}}$ , where  $\{\text{CON}_i\}$  and  $\{\text{SEP}_i\}$  denote the respective connected component sets. We use a distance threshold of 2cm for the Euclidean segmentation; to avoid noisy detections, we discard components that contain less than 75 points.

Adopting a 3D point set based representation for the spatial extent of topology events enables reasoning about event similarity in terms of metric distances.

Our assessments on spatial overlap of events will be based on the ‘ $\rho$ -overlap’ metric, defined for a pair of point clouds  $X_1$  and  $X_2$  as:

$$\text{OVERLAP}_\rho(X_1, X_2) \equiv \frac{|S_\rho^{X_2}(X_1)| + |S_\rho^{X_1}(X_2)|}{|X_1| + |X_2|}, \quad (24)$$

where  $S_\rho^B(A) \subseteq A$  contains exactly the points in  $A$  that lie within distance  $\rho$  from their nearest neighbor in set  $B$ . Clearly,  $\text{OVERLAP}_\rho(X_1, X_2) \in [0, 1]$ . It is easy to verify that this metric is simply the *intersection-over-union* ratio for the sets  $X_1 \cup S_\rho^{X_1}(X_2)$  and  $X_2 \cup S_\rho^{X_2}(X_1)$ . We use a radius value of  $\rho = 3\text{cm}$ .

We derive a many-to-many matching between sets  $\mathcal{E}^{\text{gt}}$  and  $\mathcal{E}^{\text{det}}$  by associating events in the two sets that have a significant spatiotemporal overlap. A ground truth event  $e_i^{\text{gt}}$  is matched to a detected event  $e_j^{\text{det}}$  if and only if they both belong to the same class (contact or separation), their timestamps are very close, and they share a substantial spatial overlap. If  $\mathcal{M} = \{(i, j)\}$  is the set of event matches, then  $\mathcal{M}$  contains all pairs  $(i, j)$ , for  $i = 1, \dots, |\mathcal{E}^{\text{gt}}|$  and  $j = 1, \dots, |\mathcal{E}^{\text{det}}|$ , that satisfy all three conditions:

- $l_i^{\text{gt}} = l_j^{\text{det}}$
- $|t_i^{\text{gt}} - t_j^{\text{det}}| \leq 2$
- $\text{OVERLAP}_\rho(X_i^{\text{gt}}, X_j^{\text{det}}) \geq 0.2$

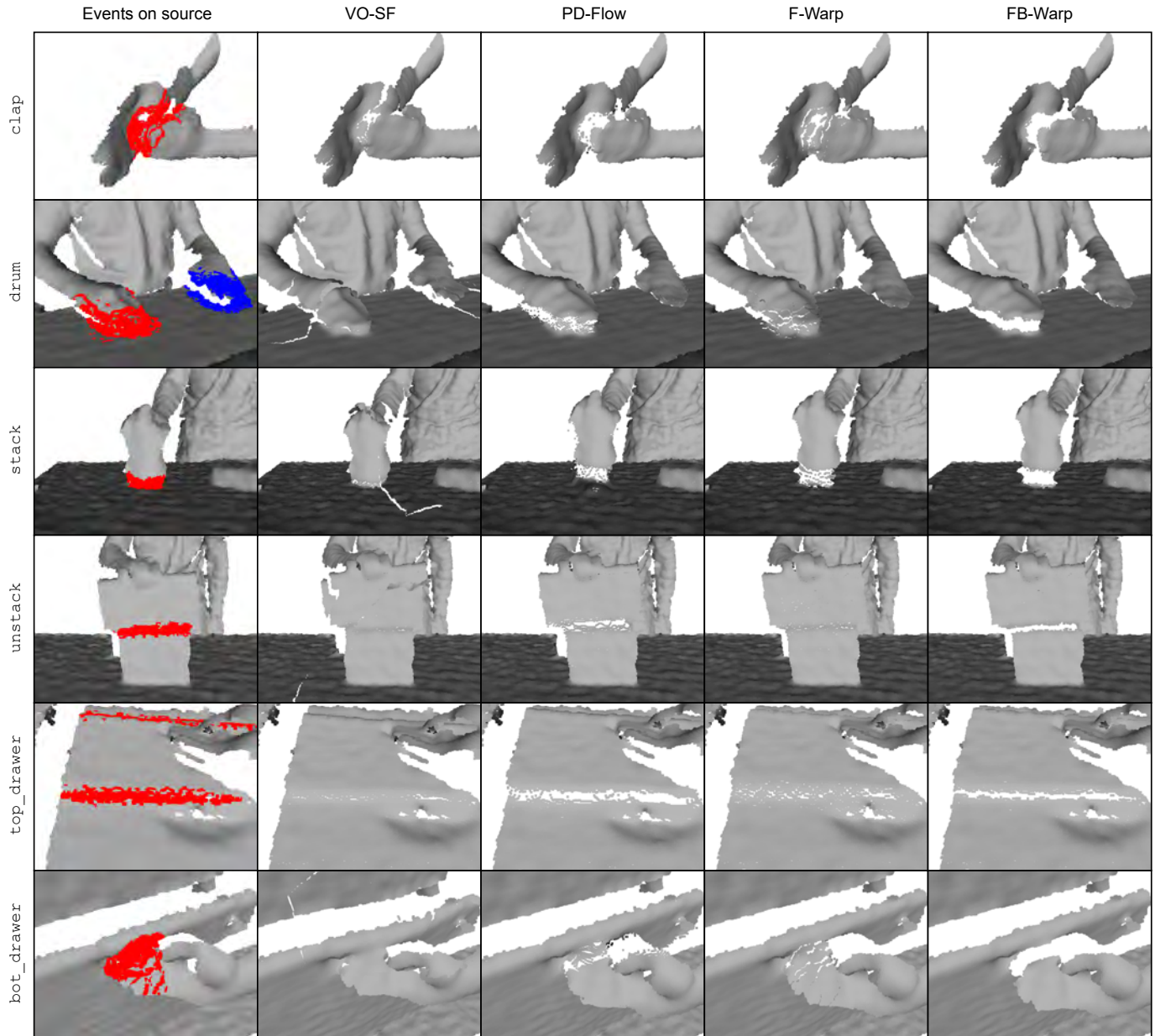


Fig. 4. Warping results on our dynamic topology dataset, with focus given to separation events. Each row corresponds to a different sequence. First column: our topological event detections overlaid on the source geometry (same as last column of Fig. 3, but rendered from a different viewpoint). Columns 2-5: warped source geometry under VO-SF, PD-Flow, F-Warp (our topology-agnostic baseline algorithm), and FB-Warp (our proposed approach).

Based on the set of all valid spatiotemporal matches  $\mathcal{M}$ , we derive two interesting event mappings: one that maps each ground truth event to a single detected one, and an ‘inverse’ one that maps each detected event to a ground truth one. Both our ‘ground truth to detected’ and ‘detected to ground truth’ mappings associate an event in the first set with its match in the second one that maximizes overlap:

$$\text{GTToDET}(i) = \arg \max_{j: (i,j) \in \mathcal{M}} \text{OVERLAP}_\rho(X_i^{\text{gt}}, X_j^{\text{det}}), \quad (25)$$

$$\text{DETToGT}(j) = \arg \max_{i: (i,j) \in \mathcal{M}} \text{OVERLAP}_\rho(X_i^{\text{gt}}, X_j^{\text{det}}). \quad (26)$$

Of course, the above are only defined for events (ground truth/detected) that have valid matches, i.e.  $(i, j) \in \mathcal{M}$ .

We present our detection results on our custom dataset in Table 2. On average, our pipeline extracts three times more events than the annotated ones (columns 2-3). This is normal, as topology changes may manifest gradually in continuous video sequences, while our annotation process treats them as being instantaneous. In columns 4-6 and 7-9, we evaluate each of the mappings GTToDET and DETToGT in terms of event coverage (fraction of  $\mathcal{E}^{\text{gt}}$  and  $\mathcal{E}^{\text{det}}$ , respectively, that was matched), average spatial overlap, and average detection delay (signed difference  $t_i^{\text{gt}} - t_j^{\text{det}}$ ). All ground truth events are covered by our detections (column 4), while an average of 66.47% of the detected events have a valid ground truth match (column 7). These coverage fractions directly correspond to *recall* and *precision*, yielding an F-score of 0.8. At the same time, the spatial overlap of



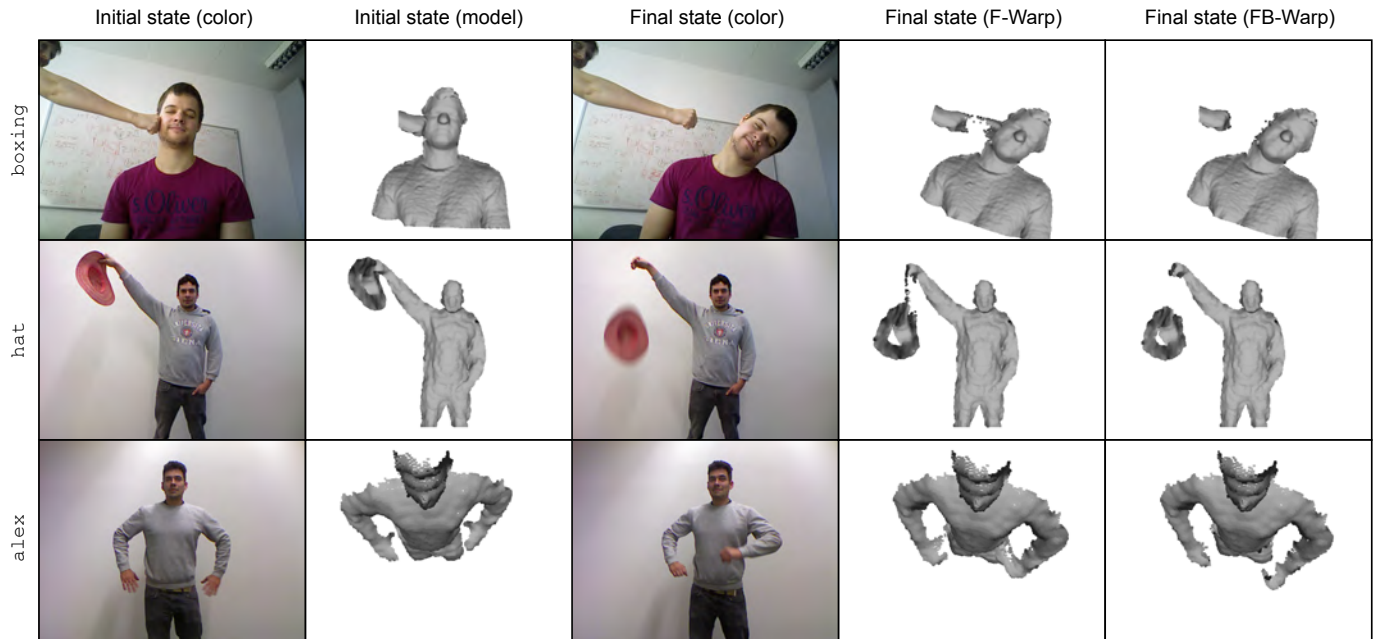


Fig. 5. Long-term model-to-frame registration. The *boxing* sequence (first row) is from [8]. The *hat* and *alex* sequences (second and third rows) are from [15]. First column: color image at initial scene state. Second column: scene model in its initial state. Third column: color image at final scene state. Fourth and fifth columns: scene model in its final state after continuous model-to-frame deformation by our F-Warp (baseline) and FB-Warp (proposed) algorithms, respectively. The models for the *alex* sequence are rendered from a different viewpoint to better visualize the artifacts introduced by F-Warp (see Section 4.4).

matched events is high (almost 80% on average for the covered ground truth events) and within the error margins of our freehand annotation, while average detection delays are very small. A small number of our detections (mostly separations) are depicted in the third column of Fig. 3, with the respective ground truth events shown in the first column.

We note that, in our context of non-rigid registration, high recall is more important than high accuracy, because missing a topology change event is very likely to result in motion estimation errors. At the same time, as discussed in the concluding remarks of Section 3.4.1, a small number of false positive detections have essentially no impact on motion estimation under reasonable assumptions. Therefore, our topology change detection mechanism has desirable properties from a motion estimation perspective, while, at the same time, being able to detect contacts and separations with reasonable accuracy.

#### 4.3.2 Registration under dynamic topology

We now evaluate our registration accuracy in the presence of dynamic topology. As discussed before, object separation events tend to induce warp field artifacts when not accounted for, while standard ‘forward’ warp estimates properly handle contacts. Therefore, we focus our assessments on areas of separation events. For each ground truth separation event, we compute the average point-to-point distance between points in the *warped* source geometry and their nearest neighbors in the target frame. More specifically, the source geometry is given by the RGB-D frame associated with the annotated separation event, while the target one is given by its next frame in the sequence. In order to avoid obscuring the differences between the two estimates in the areas of interest, instead of averaging over the whole source

TABLE 3  
Registration Under Close-To-Open Topology

Sequence	Registration error (in mm)			
	VO-SF	PD-Flow	F-Warp	FB-Warp
clap	4.652	1.155	1.326	<b>1.135</b>
drum	2.678	<b>1.006</b>	1.608	1.207
stack	3.209	1.565	1.996	<b>1.452</b>
unstack	6.457	2.371	3.138	<b>2.192</b>
separate	2.249	<b>1.735</b>	2.418	1.985
top_drawer	1.995	1.383	2.305	<b>1.325</b>
bot_drawer	5.680	1.773	2.380	<b>1.222</b>
<b>Overall:</b>	3.846	1.570	2.167	<b>1.503</b>

(event) frame, we only consider points within our ground truth annotation masks, which provide reasonable approximations of the true motion boundaries. Furthermore, we discard occluded warped points using a simple depth test against the target geometry’s depth map, with a tolerance threshold of  $\Delta z_{occ} = 1\text{cm}$ .

In our comparisons, we include four different motion field estimation algorithms: VO-SF [4], PD-Flow [3], F-Warp (our baseline), and FB-Warp (our proposed method). We report per-sequence average registration errors (in mm) over separation areas in Table 3. FB-Warp is more accurate in most sequences, achieving an average error reduction of about 30% over our F-Warp baseline, with PD-Flow being a very close second. VO-SF produces significantly less accurate results, because of the coarse pre-segmentation step on which its piecewise-rigid model is based. We also provide qualitative registration results for a subset of our ground truth separation events in Fig. 4. VO-SF introduces seam artifacts to the warped geometry, as a result of its pre-segmentation step. As the latter is highly unlikely to align

with separation boundaries, the algorithm does not preserve motion discontinuities. F-Warp, as expected, significantly oversmooths separation motion boundaries. FB-Warp and PD-Flow exhibit the best performance, with the former producing appreciably cleaner surface separations.

#### 4.4 Long-term model-to-frame registration

In our last set of experiments, we qualitatively evaluate the behavior of our proposed approach in long-term non-rigid model-to-frame tracking scenarios. As input, we use some of the public RGB-D videos that accompany the dynamic reconstruction systems [8] (*boxing* sequence) and [15] (*hat* and *alex* sequences) and exhibit dynamic scene topology. We focus our attention on sequence intervals that capture topological events. In particular, we initialize and fix a scene model that captures the scene state shortly before the topological events manifest and then continuously warp it towards each subsequent RGB-D frame. Every time step involves registering the scene model (in its deformed state that aligns it to the previous frame) to the current frame. Other than model-to-frame alignment, we perform no other updates on the scene model throughout the interval duration (e.g., no points are added, removed, or fused with more recent observations).

We show the results of our continuous model-to-frame deformation during our time intervals of interest in Fig. 5, where we compare the final state scene models produced by our F-Warp baseline and our FB-Warp proposed method. The final states correspond to time points well after the topological events completely manifested. In the *boxing* sequence, we focus on the hand-head separation event. We initialize our scene model using the partial reconstruction provided by the dataset (fusion of the first 101 frames), in which the hand is in contact with the head, and track the model for the 51 subsequent frames. In the *hat* sequence, we focus on the hand-hat separation when the actor drops the hat and track our initial model for 16 frames. In the *alex* sequence, we focus on a 27 frame interval towards the end of the sequence, where the actor first touches his torso with both hands (2 contacts) and then lifts them (2 separations). For both *hat* and *alex*, the scene model was initialized from a single RGB-D frame at the beginning of our interval of interest.

As can be seen in the last columns of Fig. 5, FB-Warp properly preserves motion discontinuities and introduces virtually no visible artifacts in the warped model geometry over the separation boundaries. At the same time, the oversmoothing behavior of F-Warp is clearly visible, as objects tend to remain connected and there are no clean separations. Standard non-rigid registration modules used in current dynamic reconstruction systems are expected to demonstrate a behavior similar to our F-Warp baseline (e.g., [7]) or potentially introduce even more severe warping artifacts if they rely on quadratic loss regularization (e.g., [8], [13], [14]). On that note, although we have performed no relevant experiments, we believe that our proposed registration pipeline would significantly simplify building dynamic reconstruction systems, as it effectively suppresses the need to perform laborious regularization graph maintenance (initialization and updates), as in [7] and [8], or handle

dynamic topology in a post-registration stage that discards problematic regions and reinitializes model tracking, as in [13] and [14].

## 5 CONCLUSIONS

We presented a complete pipeline for the non-rigid registration of arbitrary, unorganized point clouds that may be topologically different. Building upon a general warp field estimation algorithm, we introduced an efficient topology event handling post-processing phase that detects and classifies object contact and separation events, and, by exploiting the different qualities of forward and backward motion estimates with respect to different event types, locally selects the most appropriate one, in a seamless manner. We evaluated the motion estimation accuracy of our method on the MPI Sintel dataset, achieving state-of-the-art performance. Our evaluation on a custom dataset with sequences of highly dynamic scene topology demonstrated the success of our method in estimating motion on topological event boundaries, and showed promising performance in event detection. To the best of our knowledge, this is the first approach to handle dynamic topology in the context of raw point cloud registration. Furthermore, we openly release the implementation of our baseline warp field estimation algorithm as part of our point cloud processing library [18].

In this work, we focused on improving dense motion estimation on separation boundaries by reasoning about two specific types of dynamic topology: ‘open-to-close’ and ‘close-to-open’. There exist, however, object interactions that induce different types of topological changes, which our method is not equipped to handle. One such interesting example is the case of an object *sliding* on its supporting surface. In this case, while our deformation criteria might give us some hints regarding the problematic areas, our inverted backward estimate is expected to share similar oversmoothing properties as a standard, forward warp field. We are currently investigating insights that would allow us to efficiently tackle those situations, ideally without attacking the more general and (possibly) much harder problem of joint motion estimation and motion segmentation.

## ACKNOWLEDGMENTS

The support of Northrop Grumman Mission Systems University Research Program, of ONR under grant award N00014-17-1-2622, and the support of the National Science Foundation under grants BCS 1824198 and CNS 1544787 are greatly acknowledged.

## REFERENCES

- [1] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, “Visual odometry and mapping for autonomous flight using an rgb-d camera,” in *Robotics Research*. Springer, 2017, pp. 235–252.
- [2] C. Kerl, J. Sturm, and D. Cremers, “Robust odometry estimation for rgb-d cameras,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3748–3754.
- [3] M. Jaimez, M. Souiai, J. Gonzalez-Jimenez, and D. Cremers, “A primal-dual framework for real-time dense rgb-d scene flow,” in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 98–104.

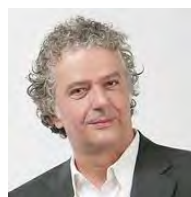
- [4] M. Jaimez, C. Kerl, J. Gonzalez-Jimenez, and D. Cremers, "Fast odometry and scene flow from rgb-d cameras based on geometric clustering," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on.* IEEE, 2017, pp. 3992–3999.
- [5] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on.* IEEE, 2011, pp. 127–136.
- [6] T. Whelan, S. Leutenegger, R. S. Moreno, B. Glocker, and A. Davison, "Elasticfusion: Dense slam without a pose graph," in *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.
- [7] R. A. Newcombe, D. Fox, and S. M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 343–352.
- [8] M. Innmann, M. Zollhöfer, M. Nießner, C. Theobalt, and M. Stamminger, "Volumedeform: Real-time volumetric non-rigid reconstruction," in *European Conference on Computer Vision.* Springer, 2016, pp. 362–379.
- [9] E. Herbst, X. Ren, and D. Fox, "Rgb-d flow: Dense 3-d motion estimation using color and depth," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on.* IEEE, 2013, pp. 2276–2282.
- [10] J. Quiroga, T. Brox, F. Devernay, and J. Crowley, "Dense semi-rigid scene flow estimation from rgbd images," in *European Conference on Computer Vision.* Springer, 2014, pp. 567–582.
- [11] 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 *3D Vision (3DV), 2015 International Conference on.* IEEE, 2015, pp. 64–72.
- [12] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4040–4048.
- [13] M. Dou, S. Khamis, Y. Degtyarev, P. Davidson, S. R. Fanello, A. Kowdle, S. O. Escolano, C. Rhemann, D. Kim, J. Taylor, P. Kohli, V. Tankovich, and S. Izadi, "Fusion4d: Real-time performance capture of challenging scenes," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 114:1–114:13, Jul. 2016.
- [14] W. Gao and R. Tedrake, "Surfelwarp: Efficient non-volumetric single view dynamic reconstruction," in *Robotics: Science and Systems*, 2018.
- [15] M. Slavcheva, M. Baust, D. Cremers, and S. Ilic, "Killingfusion: Non-rigid 3d reconstruction without correspondences," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 3, no. 4, 2017, p. 7.
- [16] M. Slavcheva, M. Baust, and S. Ilic, "Sobolevfusion: 3d reconstruction of scenes undergoing free non-rigid motion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2646–2655.
- [17] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques.* ACM, 1996, pp. 303–312.
- [18] K. Zampogiannis, C. Fermüller, and Y. Aloimonos, "Cilantro: A lean, versatile, and efficient library for point cloud data processing," in *Proceedings of the 26th ACM International Conference on Multimedia*, ser. MM '18. New York, NY, USA: ACM, 2018, pp. 1364–1367.
- [19] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 722–729.
- [20] H. Spies, B. Jähne, and J. L. Barron, "Range flow estimation," *Computer Vision and Image Understanding*, vol. 85, no. 3, pp. 209–231, 2002.
- [21] M. Hornacek, A. Fitzgibbon, and C. Rother, "Sphereflow: 6 dof scene flow from rgb-d pairs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3526–3533.
- [22] D. Sun, E. B. Sudderth, and H. Pfister, "Layered rgbd scene flow estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 548–556.
- [23] B. Amberg, S. Romdhani, and T. Vetter, "Optimal step nonrigid icp algorithms for surface registration," in *Computer Vision and Pattern Recognition*, 2007. CVPR'07. IEEE Conference on. IEEE, 2007, pp. 1–8.
- [24] R. W. Sumner, J. Schmid, and M. Pauly, "Embedded deformation for shape manipulation," in *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3. ACM, 2007, p. 80.
- [25] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Symposium on Geometry processing*, vol. 4, 2007, p. 30.
- [26] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [27] M. Rünz and L. Agapito, "Co-fusion: Real-time segmentation, tracking and fusion of multiple objects," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 4471–4478.
- [28] M. Rünz, M. Buffier, and L. Agapito, "Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR).* IEEE, 2018, pp. 10–20.
- [29] M. Slavcheva, W. Kehl, N. Navab, and S. Ilic, "Sdf-2-sdf registration for real-time 3d reconstruction from rgb-d data," *International Journal of Computer Vision*, vol. 126, no. 6, pp. 615–636, Jun 2018.
- [30] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on.* IEEE, 2001, pp. 145–152.
- [31] C. Engels, H. Stewénius, and D. Nistér, "Bundle adjustment rules," *Photogrammetric computer vision*, vol. 2, no. 2006, 2006.
- [32] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625.
- [33] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, 2011.



**Konstantinos Zampogiannis** is a PhD candidate in Computer Science at the University of Maryland, College Park. He is working in 3D vision and action representations under the supervision of Prof. Yiannis Aloimonos and Dr. Cornelia Fermüller. He completed his undergraduate degree at the National Technical University of Athens, Greece, under the supervision of Prof. Petros Maragos.



**Cornelia Fermüller** is a Research Scientist at the University of Maryland Institute for Advanced Computer Studies. She holds a Ph.D. from the Vienna University of Technology, Austria (1993) and an M.S. from the Graz University of Technology (1989), both in Applied Mathematics. Her research interest has been to understand principles of active vision systems and develop biological-inspired methods, especially in the area of motion. Her recent work has focused on human action interpretation and the development of event-based motion algorithms.



**Yiannis Aloimonos** is Professor of Computational Vision and Intelligence at the Department of Computer Science, University of Maryland, College Park, and the Director of the Computer Vision Laboratory at the Institute for Advanced Computer Studies (UMIACS). He is also affiliated with the Institute for Systems Research and the Neural and Cognitive Science Program. He was born in Sparta, Greece and studied Mathematics in Athens and Computer Science at the University of Rochester, NY (PhD 1990). He is interested in Active Perception and the modeling of vision as an active, dynamic process for real time robotic systems. For the past five years he has been working on bridging signals and symbols, specifically on the relationship of vision to reasoning, action and language.