Dense 3D displacement vector fields for point cloud-based landslide monitoring

Journal Article

Author(s):
Gojcic, Zan; Schmid, Lorenz; Wieser, Andreas

Publication date:
2021-12

Permanent link:
https://doi.org/10.3929/ethz-b-000507673

Rights / license:
Creative Commons Attribution 4.0 International

Originally published in:
Landslides 18(12), https://doi.org/10.1007/s10346-021-01761-y

Landslides (2021) 18:3821–3832
DOI 10.1007/s10346-021-01761-y
Received: 27 April 2021
Accepted: 27 August 2021
Published online: 21 September 2021

Zan Gojcic · Lorenz Schmid · Andreas Wieser

Dense 3D displacement vector fields for point cloud-based landslide monitoring

Abstract We propose a novel fully automated deformation analysis pipeline capable of estimating real 3D displacement vectors from point cloud data. Different from the traditional methods that establish displacements based on the proximity in the Euclidean space, our approach estimates dense 3D displacement vector fields by searching for corresponding points across the epochs in the space of 3D local feature descriptors. Due to this formulation, our method is also sensitive to motion and deformations that occur parallel to the underlying surface. By enabling efficient parallel processing, the proposed method can be applied to point clouds of arbitrary size. We compare our approach to the traditional methods on point cloud data of two landslides and show that while the traditional methods often underestimate the displacements, our method correctly estimates full 3D displacement vectors.

Keywords Deformation analysis · Point clouds · Deep learning · 3D displacement vector field

Introduction

The surface motion of landslides and rockfalls can be derived from pointwise or areal monitoring observations. In the case of pointwise monitoring, the displacements of a limited number of carefully selected, signalized points are measured with high accuracy using, e.g., total stations (TS) (Frukacz et al. 2017; Amaral et al. 2020) or GNSS sensors (Malet et al. 2002; Glabsch et al. 2009). These displacements then need to be generalized in order to derive the changes within the whole area of interest. On the other hand, areal monitoring techniques such as terrestrial radar interferometry (Caduff et al. 2015), spaceborne InSAR (Ye et al. 2004), terrestrial or UAV-borne photogrammetry (Niethammer et al. 2012; Kenner et al. 2018), and laser scanning (Barbarella and Fiani 2013; Huang et al. 2019), enable a nearly continuous spatial sampling, which alleviates the need for generalization and enables detecting changes at unexpected locations (Wunderlich et al. 2016). However, while areal techniques offer advantages in terms of coverage, the subsequent data analysis is still challenging and represents the main bottleneck for their widespread adoption.

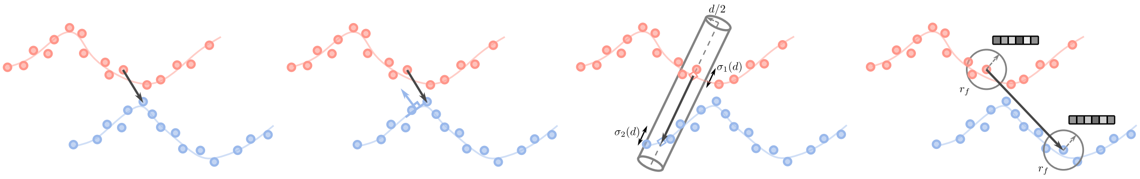
Herein, we focus on areal geomonitoring using point cloud data. Specifically, we consider a setting in which two point clouds (1) are acquired at different times, (2) cover the same underlying surface, of which at least a part moves or deforms between the acquisitions, and (3) are registered, i.e., aligned to a common reference frame. Hereinafter we denote these point clouds as source (first epoch) and target (second epoch). Furthermore, we assume without loss of generality that the point clouds are obtained by laser scanning. The registration typically necessitates prior knowledge of the stable areas and can represent a significant challenge when considering geomonitoring point clouds (Wujanz et al. 2013). However, data-driven registration algorithms capable of both identifying the stable areas and estimating the registration parameters have been proposed in the past (Wujanz et al. 2013; Friedli and Wieser 2016).

One of the main challenges in point cloud-based deformation analysis is related to the measurement principles of the sensors, which typically acquire raw data in a sensor-fixed frame, e.g., in a regular angular grid centered at the scanner in the case of terrestrial laser scanning (TLS), rather than in a frame connected to the monitored surfaces. As a consequence, if the sensor moves between the epochs, if different sensors are used at different epochs, or if at least parts of the surfaces deform between the measurements, the same points are either not measured or cannot be readily identified within the point clouds (Wunderlich et al. 2016; Gojcic et al. 2020). This implies that some sort of geometric or radiometric modeling and matching (Neuner et al. 2016; Holst et al. 2017) is needed for the deformation analysis. In the past, several methods for point cloud-based deformation analysis were proposed (Cignoni et al. 1998; Lane et al. 2003; Lindenbergh and Pfeifer 2005; Teza et al. 2007; Monserrat and Crosetto 2008; Lague et al. 2013). We focus our review on the ones most commonly used in geomonitoring, which are also implemented in standard open-source software packages, e.g., CloudCompare.[1] These include: cloud-to-cloud (C2C), cloud-to-mesh (C2M) (Cignoni et al. 1998), and multiscale model-to-model cloud (M3C2) (Lague et al. 2013) comparison. *C2C* is the simplest and the most efficient method of computing displacements between two point clouds. The displacements are computed as the Euclidean distances between the individual points of the source point cloud and their respective nearest neighbor (NN) in the target point cloud (Fig. 1a). In its basic form, *C2C* therefore does not require any modeling of the local surface (e.g., triangulation or plane fitting).

In *C2M* comparison, the displacements are computed as the shortest Euclidean distances between each point of the source point cloud and its closest facet or edge (if the orthogonal projection of the point does not fall on any of the facets) in the triangulated target point cloud (Fig. 1b). Generating a mesh surface from point clouds of natural scenes, with low resolution and high surface roughness, is a complex task. As a result, the triangulated surfaces typically comprise a lot of holes and artifacts (e.g., spikes or self-intersections), which result in spurious displacement estimates.

The *M3C2* comparison quantifies the displacements using interest points, i.e., subsampled points of the source point cloud. The processing can be broken down into three steps for each interest point. First, two sub-clouds are obtained by intersecting the source and target point cloud with a cylinder of diameter *d* and main axis aligned with

[1] http://www.cloudcompare.org/.



(a) C2C (b) C2M (c) M3C2 (d) F2S3

Fig. 1 Schematic overview of the commonly used methods for point cloud-based deformation analysis. The target point cloud (blue) is shifted to the right, relative to the source point cloud (red). The traditional methods (C2C, C2M, and M3C2) struggle to estimate the correct displacements in the case of motion that is parallel to the surface. Due to establishing the correspondences in the feature rather than Euclidean space, F2S3 is sensitive to both parallel and orthogonal motion relative to the surface. Black arrows depict the inferred displacement vectors, while light blue and red lines depict the (unobserved) underlying surface

the estimated normal vector (Fig. 1c). Second, the points of each subcloud are projected on the cylinder axis. This results in two distributions of points along the normal direction. Finally, these distributions are used to (1) compute the local point cloud roughness (in terms of standard deviations σ_1 and σ_2 in source and target point cloud), (2) compute the displacement as the distance between the mean values of the distributions (diamond symbols in Fig. 1c), and assess the statistical significance of the displacement.

These traditional point cloud comparison methods at least implicitly yield 3D displacement vectors (e.g., between the NNs or between the mean values on the cylinder axis), but these vectors just represent the distance between the surfaces not the actual displacement of points on the surfaces. Therefore, the derived vectors do not represent the real 3D displacements and the results are typically interpreted and visualized in terms of their magnitudes only. Moreover, they represent primarily the deformations orthogonal to the surfaces and fail to reflect the actual displacements, or significantly underestimate them, in case of motion/deformation parallel to the surface (e.g., debris sliding along the surface) or more complex motion (Holst et al. 2017; Gojcic et al. 2020).

A parallel line of works has tried to address this problem by either combining the point clouds with the RGB images (Wagner 2016), or by converting the points clouds to hillshade images (Holst et al. 2021). Both approaches then estimate 3D displacement vectors based on the corresponding points obtained using the 2D image features. However, these methods start by converting 3D data to 2D and treat the third component with less priority. Furthermore, they were only evaluated on a single dataset using hyperparameters that were optimized for the specific study areas.

Recently, Gojcic et al. (2019b, 2020) proposed Feature to Feature Supervoxel-based Spatial Smoothing (F2S3), a novel deep learning framework for point cloud-based deformation analysis that calculates *real* 3D displacement vectors. The main idea of F2S3 is to establish corresponding points across the epochs based on the proximity in a feature space, spanned by local feature descriptors, rather than by proximity in the Euclidean space (Fig. 1d). Local feature descriptors describe the geometric information[2] of the local neighborhood (e.g., sphere with radius r_f) around the interest points in the form of high dimensional vectors and are commonly used for point cloud registration already (Gojcic et al. 2019a; Huang et al. 2020).

By establishing the corresponding points in the feature space, F2S3 is also sensitive to displacements along the surface and thus yields full 3D displacement vectors. We could show that F2S3 outperforms the traditional methods on real-world geomonitoring datasets when the hyper-parameters are chosen appropriately. However, the only implementation of F2S3 that is currently available (Gojcic et al. 2020) is computationally very complex, not fully automated, and requires in-depth knowledge of the algorithm and the deformation process for the appropriate choice of the hyper-parameters.

In this work, we take a step further and embed the F2S3 workflow into a fully automated deformation analysis pipeline, which can easily be applied to point clouds of arbitrary size. To this end, we (1) propose an automated tiling procedure, which divides the point clouds into smaller tiles that can efficiently be processed in parallel, and (2) replace all user-selected hyperparameters by values directly derived from the input point clouds. Furthermore, we (3) optimize the run time and memory complexity by using a more efficient local feature descriptor (Poiesi and Boscaini 2021). As a result, even very large point clouds with several tens of millions of points can be processed on a standalone computer within a couple of hours.

We demonstrate our approach using two case studies of actual landslides and compare the results to the state-of-the-art methods. Although the primary motivation for the underlying research is 3D deformation analysis, the method proposed herein can also be used for solving or mitigating other point cloud processing challenges, e.g., fine registration or segmentation of scenes into deforming/moving and stable parts.

Methodology

In this section, we describe the individual modules of the proposed point cloud-based deformation analysis method, which is shown schematically in Fig. 2. The core of the method is the F2S3 framework (Gojcic et al. 2020), which can be roughly divided into two steps. First, an initial displacement vector field is estimated by determining the pointwise correspondences in the feature space ("Estimating the initial displacement vector field" section). This step is based on the assumption that the same points are measured

[2] Radiometric and spectral information (e.g., intensity, color) could be incorporated in the local feature descriptors, but is usually omitted because it is much more specific to each sensor, to environmental conditions etc., and thus much harder to generalize.

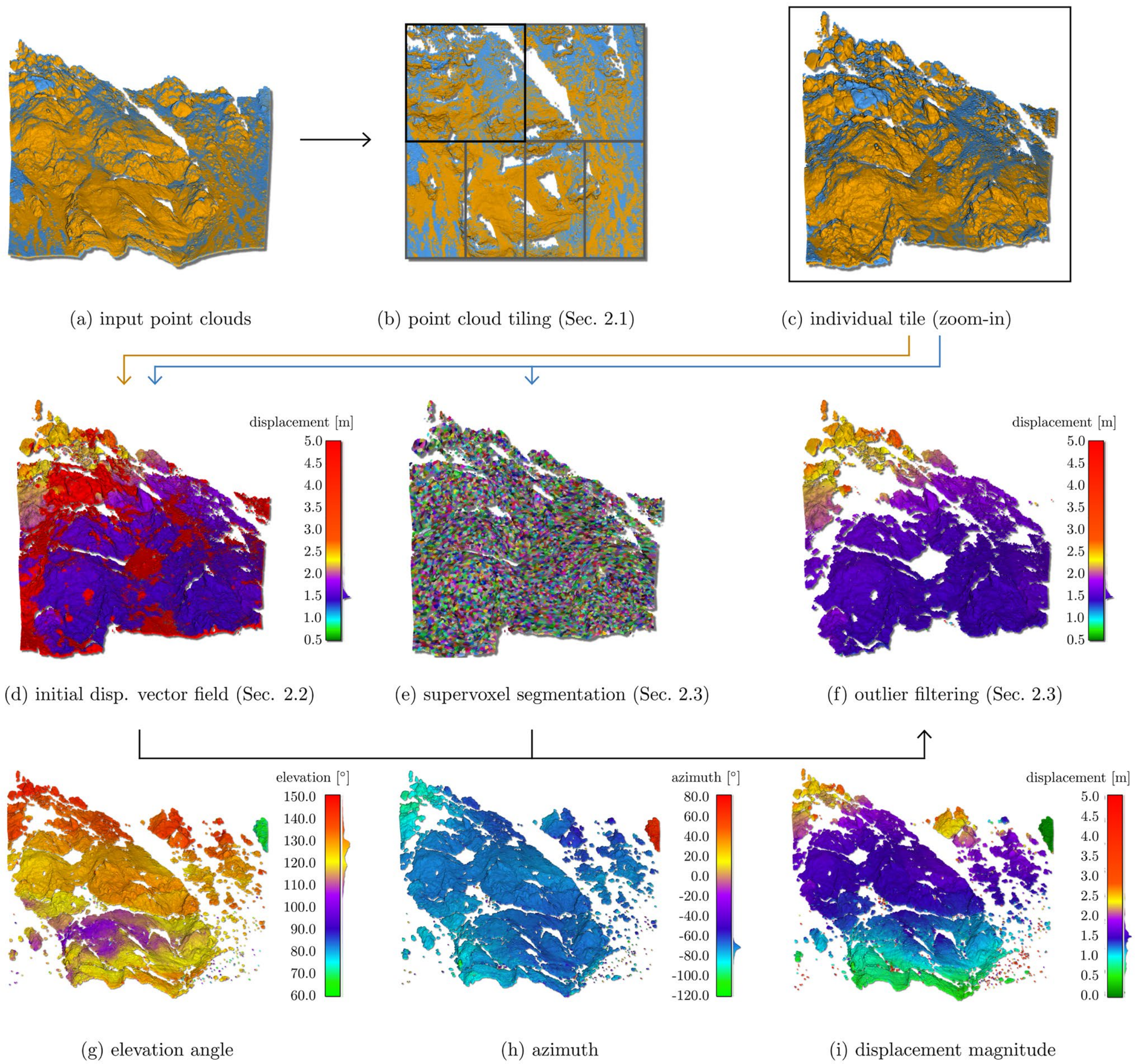


Fig. 2 Workflow of the proposed point cloud-based deformation analysis method. We assume that the input point clouds (orange, blue) are registered (**a**) and start by splitting them into tiles that can be processed in parallel (**b**). The subsequent operations (**d**–**f**) are performed independently for each individual tile (**c**). The output are 3D displacement vectors estimated for all points for which a reliable correspondence was established in the feature space (**g**–**i**)

in both epochs and that the actually corresponding points are closest in the feature space. This is only approximately valid in practice, e.g., due to the low sampling resolution, motion, occlusions, and areas without sufficiently salient features. As a result, the initial displacement vector field is noisy and contains false correspondences. The second step is therefore dedicated to the filtering and smoothing of the initial displacement vector field ("Filtering and smoothing the initial vector field" section). To this end, the points of the source point cloud are first combined into geometrically coherent local patches, which are assumed to move as *nearly* rigid bodies. Finally, the noisy displacement vectors, which do not agree with the dominant rigid motion component within each local patch, are filtered out. The second step of F2S3 is motivated by the observation that the actual displacement vector fields are not random and in fact show high local regularity (e.g., large stones move as rigid bodies and nearby areas without discontinuities often move similarly).

We complement the F2S3 workflow with an automated tiling procedure ("Point cloud tiling" section) and replace the user-selected parameters with the ones estimated directly from the data. The relation for each individual parameter is described in the respective subsections. The resulting fully automated procedure takes two point clouds as input and returns the 3D displacement

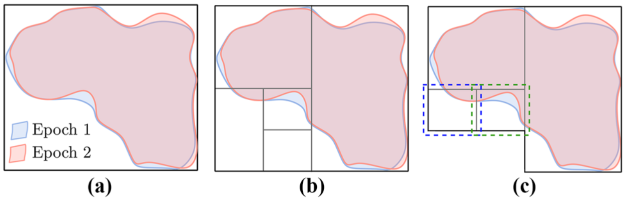


Fig. 3 Point cloud tiling. After an orthographic projection of the point clouds to 2D (**a**), we perform a recursive subdivision of the bounding box into disjoint tiles (**b**), until the condition of the maximum number of points within a tile is satisfied by both point clouds (**c**). To avoid bordering effects we consider a buffer zone of 20 m around each tile [blue and green dashed rectangles in (**c**)] in subsequent computations

vectors of all points of the source point cloud for which a reliable correspondence was established. The proposed approach is completely modular and individual blocks can be replaced in the future. For example, with the rapid progress of deep learning, better-performing local feature descriptors will soon be available and could be integrated into our workflow. A detailed description of the algorithms used to implement the individual blocks is therefore out of the scope of this paper, but we will refer the reader to the respective references.

Point cloud tiling

Processing point clouds with tens of millions of points or even more at once, would make the computation of the local feature descriptors slow and the correspondence search in the feature space intractable. We therefore divide the point clouds into sufficiently small tiles, which can be processed efficiently in parallel (Fig. 3). To this end, we first project the point clouds to 2D along the coordinate axis that yields the 2D orthographic projection with the largest bounding box. We then recursively subdivide the obtained 2D bounding boxes (binary subdivision at the middle point of the longer edge) into smaller tiles, until individual tiles contain less than the predefined number of points (Fig. 3). The maximum number of points per tile is limited by the memory needed for the subsequent operations. On the other hand, the correspondence search will later only be carried out within each tile such that the tiles should be larger than the (expected) displacements. We observed that tiles with approximately one million points offer a good trade-off between the computational efficiency and the spatial extent of the tiles. To avoid bordering effects in the computation of the local feature descriptors we extend each tile by a 20 m buffer zone (always larger than r_f) beyond each original edge (see Fig. 3c). While this simple way of tiling performs well for the geomonitoring datasets presented herein, a more sophisticated approach will be required to also obtain displacement vectors across the edges of the tiles and to handle strongly curved surfaces. We leave this for future research.

All subsequent steps of our pipeline are performed independently for each tile. The displacement vectors corresponding to the full original point clouds are finally obtained by recombining the tiles using a simple union operation.

Estimating the initial displacement vector field

We follow the framework of F2S3 and establish the initial displacement vector field based on the corresponding points in the feature space. Specifically, we (1) compute the local feature descriptors for all points in both point clouds, (2) establish correspondences based on the NN search in the feature space, and (3) determine the initial displacement vector field as 3D vectors connecting the two points of each correspondence. The individual steps are summarized in the following and a detailed description thereof is given in Gojcic et al. (2020).

Local feature descriptor

In recent years several handcrafted (Rusu et al. 2009; Tombari et al. 2010) and learned (Gojcic et al. 2019a; Choy et al. 2019) 3D feature descriptors were proposed. Here, *handcrafted* denotes that the function that maps the point to its feature descriptor is manually engineered based on heuristics and *learned* denotes that it is the result of machine learning based on annotated examples of corresponding points. Most state-of-the-art, local feature descriptors are invariant to rotation and translation by design but assume local rigidity, i.e., the local neighborhood around the interest point should not be deformed too much between the epochs.

Herein, we adopt a state-of-the-art learned local feature descriptor denoted as distinct 3D local descriptor (DIP) proposed by Poiesi and Boscaini (2021). Compared to 3DSmoothNet (Gojcic et al. 2019a) used in the initial F2S3 implementation, DIP is more efficient to compute and achieves a comparable performance (ratio of correct correspondences) with lower dimensional descriptors (64 vs. 128). The reduced dimensionality of the descriptors in turn reduces the computation time of the correspondence search. DIP takes the points within the spherical neighborhood with radius r_f centered at the interest point as input and outputs a 64-dimensional feature descriptor. We relate the feature radius r_f to the median resolution v of the point clouds such that $r_f := 10\sqrt{3} \cdot v$, where $v = \max(v_s, v_t)$ is computed for each tile independently and v_s and v_t denote the median resolution of source and target point cloud tile, respectively.

Due to the lack of annotated pointwise correspondences in geomonitoring data, we train DIP on the point clouds derived from the RGB-D images of indoor scenes (e.g., offices, hotel rooms, and tabletops) from the 3DMatch datasets (Zeng et al. 2017). We then directly use this pretrained DIP on the outdoor point clouds presented herein. This direct generalization (i.e., training on indoor and evaluating on outdoor data) is possible because DIP only considers local geometry. While on a large scale, the indoor and outdoor scenes look vastly different, locally the features such as edges and corners are similar enough to enable generalization. Despite the good generalization, certain domain gap remains and in the future effort should be put into generating outdoor datasets, which would enable training directly on the target domain.

Correspondence search in the feature space

Even though the correspondence search is performed independently for each tile, a relatively large number of points (≈ 1million) in combination with the 64-dimensional feature descriptors, still make it computationally challenging. On the one hand, brute-force computation (i.e., explicitly computing all the pairwise distances and selecting the smallest one) is intractable due to the large memory complexity. On the other hand, data structures such as kd-trees (Bentley 1975), which are often used to speed up the search, suffer from the curse of dimensionality (Bellman 1957), i.e., their efficiency drops quickly with increased dimension of the input data.

We therefore adopt a highly optimized graph-based *approximate* NN search algorithm HNSW (Malkov and Yashunin 2018) in the combination with the *l*2-distance. The recall[3] of HNSW can be tuned with the parameters e_c, M, and e_s, which control the quality of the graph construction, the maximum number of outgoing connections, and the quality of the search, respectively. With the empirically determined parameters $e_c = e_s := 300$ and $M := 12$, which favor efficiency over quality, HNSW reaches a high recall of $> 95\%$ in our experiments.

Filtering and smoothing the initial vector field

In geomonitoring, parts of the scene are stable over time, others move as rigid bodies, and some change completely, e.g., due to the flow of earth and debris. While on a macro-level this causes discontinuities in the displacement vector field, local patches tend to move in a spatially coherent, rigid manner over time. This notion of local rigidity is not only important for the extraction of local feature descriptors, but the regularity that it imposes can also be used as a (soft) constraint for filtering and smoothing of the initial noisy displacement vector field.

Supervoxel segmentation

The local rigidity constraint should only be applied within areas that do not cross discontinuities of the displacement vector field, which are unfortunately not known in advance. Empirically, these discontinuities occur primarily on the object boundaries, which are challenging to detect in the raw point clouds. Therefore, in the F2S3 framework instead of directly detecting the objects, the source point cloud is over-segmented into supervoxels, i.e., geometrically coherent point patches. The advantage of combining points into patches that are usually smaller than most rigid objects, is twofold (1) by reducing the size of the patches, the probability of a patch crossing a discontinuity is lower, and (2) if a patch does cross a discontinuity, a smaller area and hence fewer points are affected. However, the optimal size of the supervoxels is a trade-off. On the one hand, very small patches are more likely to move as a rigid body and less probable to cross discontinuities. On the other hand, they only contain few points and in the presence of noise and outliers might not carry enough information to deduct the underlying rigid motion. Hence, the supervoxels should contain enough points, while still being smaller than the objects that can be resolved and for which nearly rigid motion can be detected at the given resolution.

To extract such supervoxels, we adopt a recently proposed supervoxel segmentation algorithm (Lin et al. 2018), specifically designed for object boundary preservation. The object boundaries are preserved by incorporating the cosine similarity of the normal vectors[4] in the optimization energy function. The approximate size of the resulting supervoxels is controlled by an input parameter r_s, which we again relate to the median resolution v_s of each *source* point cloud tile such that $r_s := 10\sqrt{3} \cdot v_s$.

In the following, the points belonging to the same supervoxel are considered to act as a nearly rigid body. Because the spatial resolution of the geomonitoring point clouds is generally low, supervoxels will occasionally cover an area larger than the individual objects, e.g., individual stones in the gravel. In these cases, the points assigned to a single supervoxel might not move as a rigid body and will be removed by our filtering algorithm. However, the local feature descriptor is also based on the assumption of local rigidity, and hence the correspondences established in these areas will typically be unreliable anyway. A more rigorous deformation analysis of such areas necessities either acquisition with a higher spatial sampling rate or some sort of super-resolution approach. We leave this investigation for future work.

Filtering the noisy displacement vectors

The outlier filtering module is motivated by the regularity in the displacement vector fields and aims to impose the congruence of the displacement vectors within each individual supervoxel. To this end, we resort to a neural network-based outlier detection algorithm proposed by Gojcic et al. (2020).[5] This network takes the initial displacement vectors (coordinates of the start and endpoints) of an individual supervoxel as input. It outputs an inlier score between 0 and 1 for each of them, where 0 denotes a high probability that this displacement vector is an outlier and 1 denotes a high probability that it is an inlier. The inlier scores are computed in a single forward pass (i.e., function evaluation) for each supervoxel. They are based on the local context, which means that within the network the individual vectors have to be compared to each other. This is achieved by the special context normalization layers (Yi et al. 2018). As the training procedure again requires annotated point correspondences, we once more resort to the point clouds of indoor scenes (see "Estimating the initial displacement vector field" section). More information about the filtering network is available in Gojcic et al. (2020).

The inlier scores inferred by the filtering module can be used in two ways, (1) for filtering the outlier displacement vectors by thresholding the scores, which results in a "sparse" displacement vector field or, (2) additionally using the inlier vectors (after thresholding) to estimate the parameters of the congruence transformation and recompute the displacement vectors for all points within the supervoxel. In all evaluations presented in "Experimental evaluation" section we show the result of the latter case that provides an additional layer of smoothing within the supervoxels and further mitigates the noise caused by imposing the direct point-to-point correspondences.

[3] In the evaluation of the approximate NN algorithms, *recall* denotes the percentage of ground truth NNs that are correctly recovered.

[4] Boundary preservation is based on the assumption that the direction of the normal vectors changes significantly at the object boundaries.

[5] The filtering could also be performed with traditional robust estimation methods e.g., RANSAC (Fischler and Bolles 1981), but would be more time-consuming (Gojcic et al. 2020).

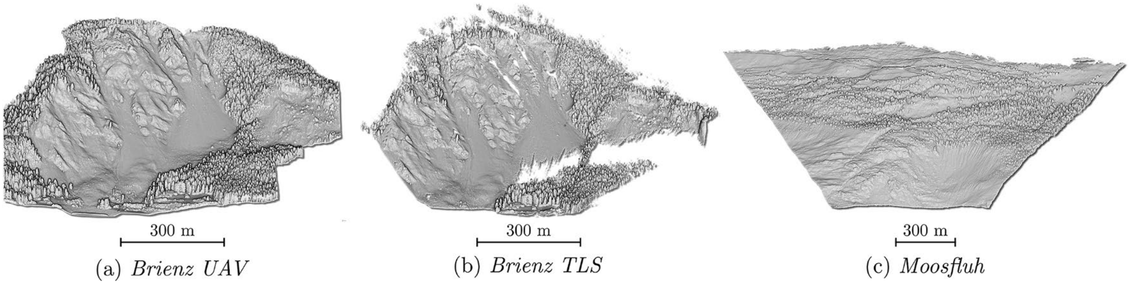


Fig. 4 Datasets used to evaluate the method proposed herein. *Brienz UAV* and *Brienz TLS* are point clouds of the same landslide acquired using different sensors. For improved readability only the point clouds of the respective first epoch are depicted here

Experimental evaluation

In this section, we evaluate the proposed point cloud-based deformation analysis method in two case studies: the Brienz landslide ("Brienz landslide" section) and the Moosfluh landslide ("Moosfluh landslide" section). In the former, we process point clouds acquired using UAV-based LiDAR as well as point clouds obtained from TLS. In the latter, we process point clouds acquired using TLS only. The point clouds of both case studies are depicted in Fig. 4.

Evaluation protocol

We compare the displacement magnitudes estimated using our method to the ones obtained using the established point cloud deformation analysis methods *C2C* and *M3C2* as well as to results derived from TS measurements, which are available for a few points and are considered as ground truth (GT) because of their expected superior accuracy. For *C2C* and *M3C2* we use the implementations available in the open-source point cloud processing software CloudCompare v2.10.1 (Girardeau-Montaut 2019). The TS data are not available in the same reference frame as the point clouds, and aligning them could introduce biases whose effects would be impossible to distinguish from errors of the displacement estimates. So, we use them only for the comparison of the displacement magnitudes and not the full 3D displacement vectors. However, we also evaluate the full 3D displacement vectors of our method by comparing them to manually annotated data. The annotations are obtained in the following manner: (1) we cut a small patch around each interest point in the source point cloud (source patches), (2) we coarsely register each individual patch to the target epoch by manually selecting the corresponding points, and (3) we refine the registration by an ICP algorithm (Besl and McKay 1992). The latter step yields target patches from which the 3D displacement vectors can easily be obtained due to the one-to-one correspondences. In the Brienz case study, where TS measurements are available, we use the locations of the TS targets as the interest points, whereas in the Moosfluh case study we sample the interest points such that they cover the large unstable area at the bottom of the scene (c.f. Fig. 11). When performing the annotation, we once more assume that the local patches move as a rigid body. To confirm this assumption and the annotation process in general, we compare the annotated displacement magnitudes with the reference data and we observe a good agreement, e.g., less than 5% difference in the *Brienz UAV* dataset ("Brienz landslide" section). Based on this comparison, we conclude that the annotated data faithfully represents the actual underlying displacement vector field.

For easier interpretation, we quantify the differences between the estimated and the annotated displacement vectors using the magnitude, the lateral horizontal deviation Δ_l, and the vertical one Δ_v instead of the deviations in the Cartesian coordinate frame. The lateral and vertical components are defined as

$$\Delta_l = \sqrt{p_1^2 + p_2^2} \quad \text{and} \quad \Delta_v = |p_3| \tag{1}$$

where

$$\mathbf{p} = \mathbf{o} - \left\langle \mathbf{o}, \frac{\mathbf{g}}{||\mathbf{g}||} \right\rangle \frac{\mathbf{g}}{||\mathbf{g}||} \tag{2}$$

and $\mathbf{o} \in \mathbb{R}^3$ and $\mathbf{g} \in \mathbb{R}^3$ denote the median estimated and median GT (annotated) displacement vector, respectively. $\langle \cdot, \cdot \rangle$ denotes the dot product and $|| \cdot ||$ the vector norm.

Brienz landslide

The Brienz landslide is located just north of the Brienz village in the Albula valley in Graubünden, Switzerland. The landslide area moves with a rate of up to several meters per year and the village itself is displaced by a few decimeters each year (Krähenbühl and Nänni 2017; Häusler and Fäh 2018). In our case study we consider two datasets. The point clouds of the first dataset (*Brienz UAV*) were acquired using a UAV-based lidar system (Riegl RiCOPTER) in two epochs about 2.5 months apart in summer 2018. The point clouds of the second dataset were acquired using a terrestrial laser scanner Riegl VZ-6000 in two epochs about 3 months apart in fall 2020. The spatial resolution of the *Brienz UAV* point clouds is more uniform and equals ≈ 7.5 cm (resulting in ≈ 160 M points per epoch) whereas the spatial resolution of the TLS point clouds varies between 5 cm at the bottom and 15 cm at the top part of the slope (resulting in ≈ 65M points per epoch).

Brienz UAV

We start the evaluation on the *Brienz UAV* dataset by comparing the displacement magnitudes estimated using the point cloud-based methods to the reference TS measurements. Here, the estimates of the point cloud-based methods are defined as the median displacement magnitude of all vectors that lie within a patch with

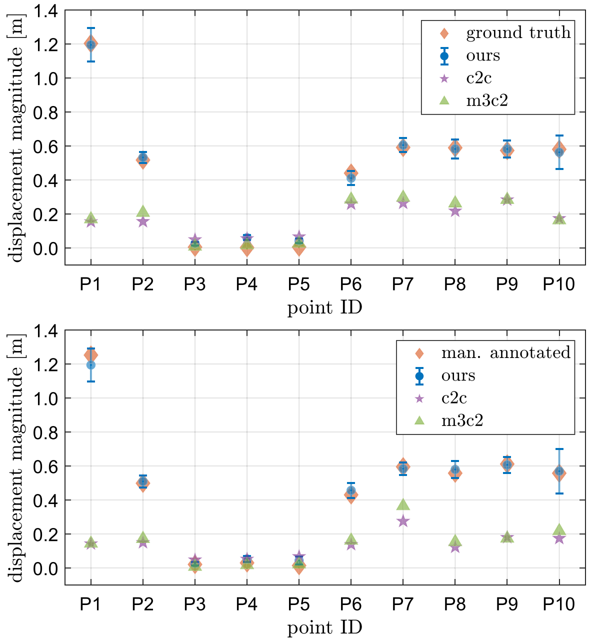


Fig. 5 Comparison of the displacement magnitudes estimated using our method, *C2C*, and *M3C2* with the reference TS measurements (top) and manually annotated data (bottom) on the *Brienz UAV* dataset

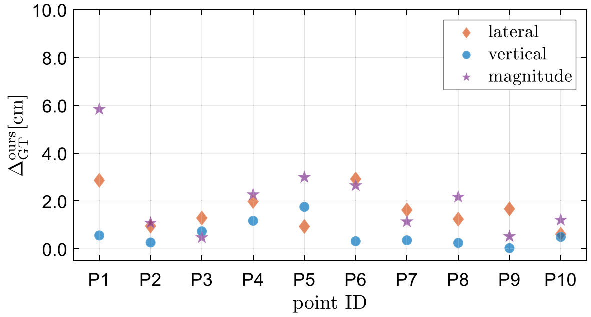


Fig. 6 Comparison of the 3D displacement vectors estimated by our method with the manually annotated data on the *Brienz UAV* dataset

approximately[6] 5 m radius centered at each of the TS targets. While the positions of the TS targets are not known with an accuracy that would facilitate using them for registration, their approximate coordinates are sufficiently accurate (≈ 30 cm) to center these patches. The same procedure is also used to obtain the estimates of the point cloud-based methods in the analysis of the *Brienz TLS* and *Moosfluh* datasets. Figure 5 shows that the maximum deviation of our method from the GT is about 5 cm (at point P5) and the average deviation lies just below 2 cm. Even though the average deviations might still include a possible bias due to the non-perfect registration of the two epochs, they are more than three times smaller than the mean resolution of the point clouds. So, our method yields correct estimates irrespective of the actual displacement magnitude. On the other hand, both *C2C* and *M3C2* greatly underestimate the displacement magnitudes for all points that actually move. The largest error (more than 1 m) occurs at point P1 that was displaced the most between the epochs. Similar results can be observed in the comparison of the point cloud-based methods to the manually annotated data (Fig. 5 bottom).

We further use the annotated data to evaluate the full 3D displacement vectors estimated by our method (Fig. 6). The lateral and vertical deviations of our estimates are all smaller than 4 cm, with most of them below 2 cm. This evaluation confirms that our method is not only capable of estimating the correct displacement magnitudes but in fact the full 3D displacement vectors. This opens up new possibilities for landslide analysis such as for example the automatic extraction of the sliding surfaces from the estimated 3D displacement vectors.

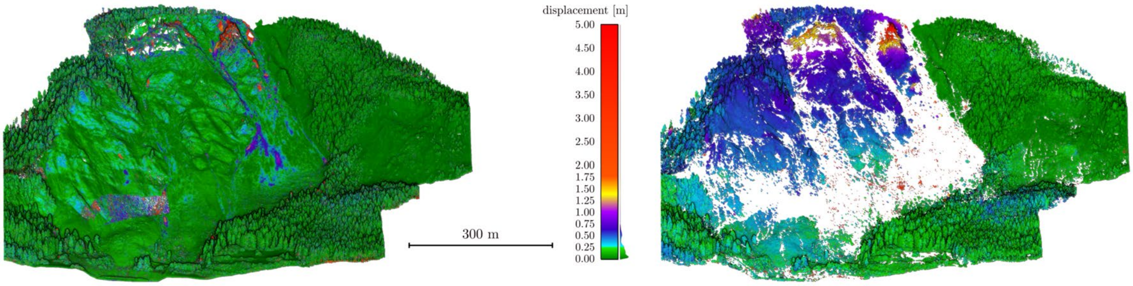
We conclude the evaluation on the *BrienzUAV* dataset with the qualitative analysis of the estimated displacement magnitudes. Figure 7 shows that the estimates of our method (right) and *M3C2* (left) are vastly different, except in the stable areas on right-hand side of the point clouds. In fact according to the *M3C2* estimates, the whole area should be almost completely stable. Relating this result to the available reference measurements again hints at the underestimation problem of the traditional point cloud-based deformation analysis methods. Contrarily, according to the estimates of our method, most of the area is unstable and the displacement vector field shows a smooth trend from low displacement magnitudes at the bottom to the larger displacements at the top of the slope. A similar trend can be seen in the TS measurements.

Brienz TLS

In the assessment of the *Brienz TLS* results we follow the same evaluation protocol as for *Brienz UAV*, but consider four additional TS reference points (P11–P14) that were installed epochs of both datasets. Top and bottom plot in Fig. 8 depict the comparison of the point cloud-based methods to the reference TS measurements and manually annotated data, respectively. The maximum and average deviation of our method are 6.7 cm and 3.2 cm, respectively, when compared to the TS reference data and 3.2 cm and 1 cm when compared to the manually annotated data.

The comparison of the full 3D vectors (depicted in Fig. 9) in which all the deviations are once more below 4 cm, not only reconfirms that our method is capable of estimating the true 3D displacement vectors but also that it can process point cloud data from different sensors. We additionally provide a 3D visualization of the estimated displacement vectors for a randomly selected point cloud tile in Fig. 10. For improved readability, only 0.5% of the calculated, reliable 3D vectors are plotted. They are randomly selected and thus provide a realistic impression of the displacement patterns and also show the regions where no or few vectors are found. The estimated 3D displacement vectors show high local consistency of both magnitude and direction.

[6] The exact size of each patch is dependent on the resolution of the point clouds and the local surface geometry in that neighborhood. However, exactly the same patches are used for all the methods.



(a) *Brienz UAV* - M3C2

(b) *Brienz UAV* - Ours

Fig. 7 Point clouds of the *Brienz UAV* dataset color coded by the displacement magnitudes estimated using *M3C2* (left side) and our method (right side). Our method estimates a smooth displacement vector field and filters out unreliable displacement vectors in areas of small cobble and debris. Note also the discontinuities of the *M3C2* estimates in the unstable areas

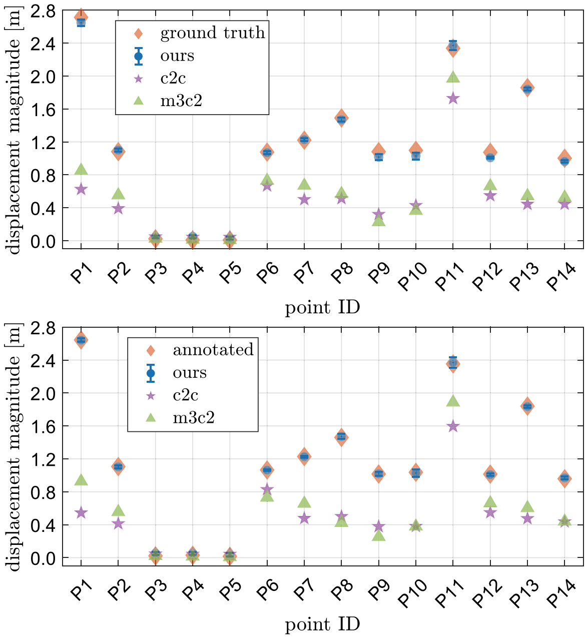


Fig. 8 Comparison of the displacement magnitudes estimated using our method, *C2C*, and *M3C2* with the reference TS measurements (top) and manually annotated data (bottom) on the *Brienz TLS* dataset

Moosfluh landslide

Moosfluh is a large landslide area located on the terminus of the Aletsch glacier located in southeast Switzerland. The slope movements of the Moosfluh area are due to the glacier retreat. Since the mid 1990s, when the glacial ice began to diminish at a much faster rate, the average motion reached several decimeters to meters per year and is still accelerating (Kos et al. 2016; Glueer et al. 2020). In recent years, the Moosfluh area was a subject of several monitoring campaigns using various measurement techniques including ground-based and satellite-based radar interferometry (Strozzi et al. 2010; Kos et al. 2016), photogrammetry (Strozzi et al. 2010),

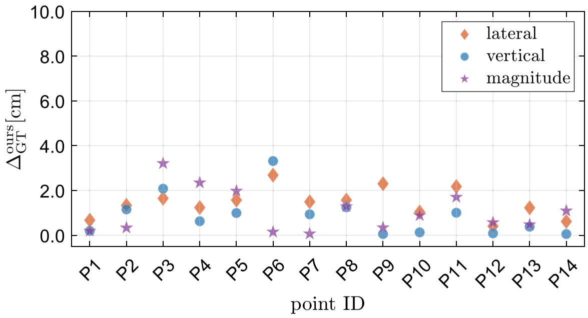


Fig. 9 Comparison of the 3D displacement vectors estimated by our method with the manually annotated data on *Brienz TLS* dataset

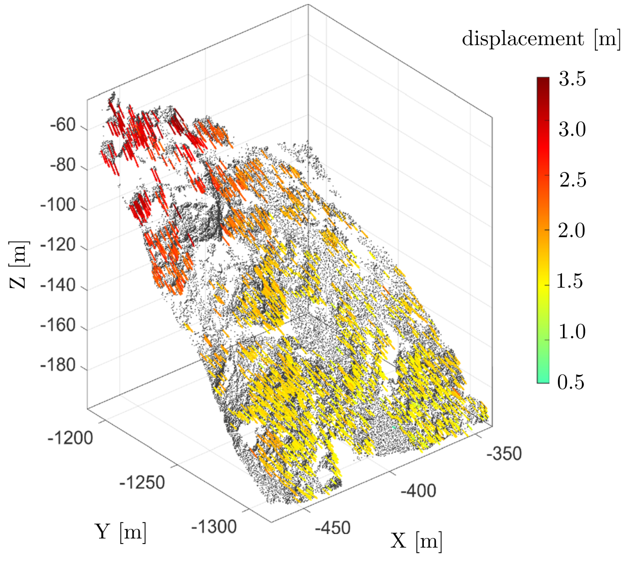
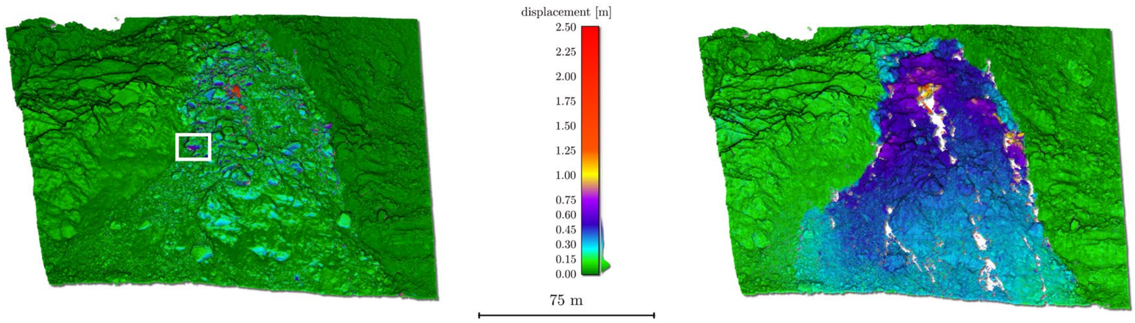


Fig. 10 3D displacement vectors for one tile of the *Brienz TLS* dataset estimated using the method proposed herein. Note the local consistency of the estimates and the smooth increase of the displacement magnitude from the bottom to the top. For improved readability only 0.5% of all vectors are depicted



(a) *Moosfluh one day (zoom-in)* - M3C2

(b) *Moosfluh one day (zoom-in)* - Ours

Fig. 11 Zoom-in to the unstable areas of the *Moosfluh one day* dataset. Point clouds are color coded by the displacement magnitudes estimated using *M3C2* (left side) and the method proposed herein (right side). Note how the estimates of *M3C2* are larger on the front faces (actual motion is orthogonal to these faces) of individual objects

GNSS (Strozzi et al. 2010) as well as terrestrial (Friedli et al. 2019) and airborne (Kos et al. 2016) laser scanning.

Herein, we use point clouds of three epochs acquired in the course of two measurement campaigns. The first epoch was acquired on 21.09.2017 and the second and third on 05.09.2018 and 06.09.2018, respectively. A long-range terrestrial laser scanner Riegl VZ-4000 with a specified range accuracy of 15 mm and a user-selected angular scanning resolution of 10 mgon was used in both campaigns. The point clouds were acquired from the opposite side of the glacier valley. The resulting distances are between 700 m at the bottom and up to 2500 m at the top of the slope. The spatial resolution of the point clouds varies accordingly from about 0.1 m at the bottom to 0.4 m at the top. In this evaluation, we consider two scenarios. We start by comparing epochs two and three (*Moosfluh one day*) and then proceed to epochs one and three (*Moosfluh 1 year*). In the latter case, displacements of several meters are expected at the most active locations.

Moosfluh 1 day

In line with the results of the previous measurement campaigns (Kos et al. 2016; Glueer et al. 2020), the estimates of our method (and M3C2) show that the most active part of the landslide is located at the bottom-middle part of the slope. We therefore focus our analysis only on this active part. Figure 11 shows the displacement magnitudes, estimated using M3C2 and our method. Our method yields displacements between 0.3 m (bottom part) and 1 m (top part) with a smoothly increasing trend from the bottom to the top. On the other hand, most of the displacements estimated using *M3C2* are below 0.2 m and reach higher values only on the front faces of the individual objects, i.e., in the small areas where the true displacement is nearly orthogonal to the surfaces. In fact, *M3C2* even yields estimates which are physically impossible. For example, according to the *M3C2* results the front of the boulder marked by the white rectangle in Fig. 11a moved by more than 0.5 m, while the top part of the same boulder is indicated as stable.

Due to the lack of independent reference data in the active parts of the slope, we once more resort to the manually annotated data in order to quantitatively evaluate the estimates of our method. Specifically, we select 15 points distributed across the active area and annotate their 3D displacement vectors as described in "Evaluation protocol" section. The results of this comparison are shown in Fig. 12. Whereas the traditional methods once more underestimate the displacement magnitudes (by up to 50 cm), the estimates of our method agree well with the annotated data. The maximum deviation of our method to the annotated data is 5.8 cm at point P2, and the average absolute deviation is slightly below 2.5 cm. When comparing the estimated 3D displacement vectors (Fig. 12 bottom) the lateral and vertical deviations are below 4.5 cm, which is much lower than the spatial resolution of the input point clouds (≈ 15 cm).

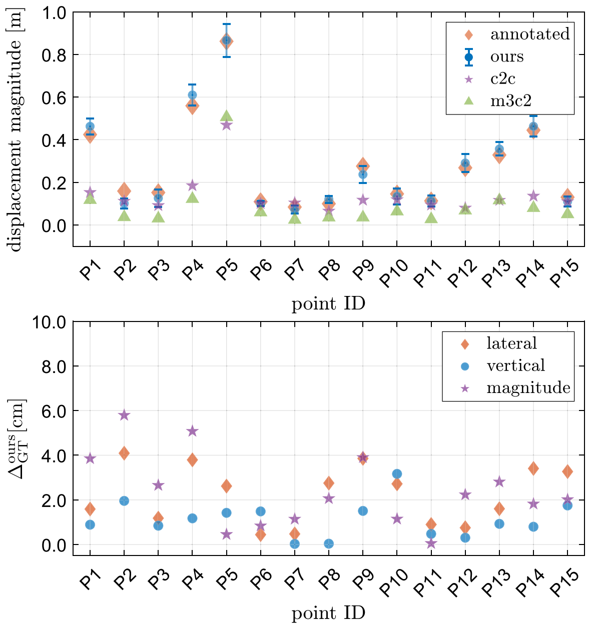
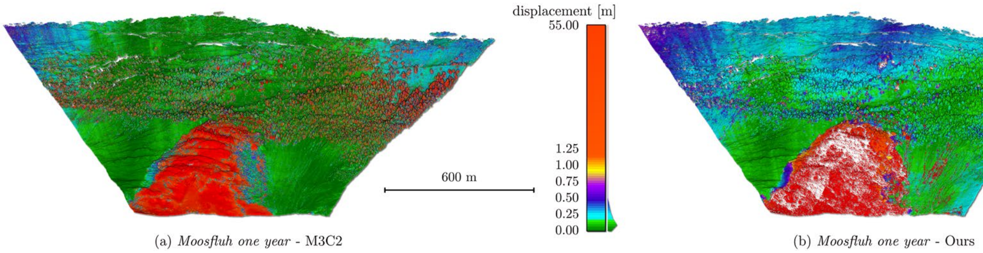


Fig. 12 Comparison of the displacement magnitudes (top) and 3D displacement vectors (bottom—only ours) estimated using our method, *C2C*, and *M3C2* with the manually annotated data on *Moosfluh one day* dataset



(a) *Moosfluh one year* - M3C2

(b) *Moosfluh one year* - Ours

Fig. 13 Point clouds of the *Moosfluh 1 year* dataset color coded by the displacement magnitudes estimated using *M3C2* (left side) and the method proposed herein (right side). Our method filters out a large part of the point in the active area. For improved readability we use a non-linear color scale

Moosfluh 1 year

We conclude this evaluation with a scenario in which the two point clouds of the Moosfluh landslide were acquired over a time span of 1 year. This represents a scenario in which the displacements in the active area are expected to be in the range of several tens of meters. Because there are no suitable artificial targets available in this dataset, the registration of the point clouds was performed using an ICP. Due to the fact that the whole area is unstable, the estimated registration parameters are impaired. This highlights that the point cloud registration is still a challenge in cases where the whole scene is unstable.

However, although the error of the registration is reflected in the estimated displacement vector fields (Fig. 13), the results can still be compared for plausibility or with the manually annotated data (Fig. 14), which is insensitive to the miss-alignment of the two epochs, i.e., the effect of a miss-alignment is the same for annotated data and our estimates.

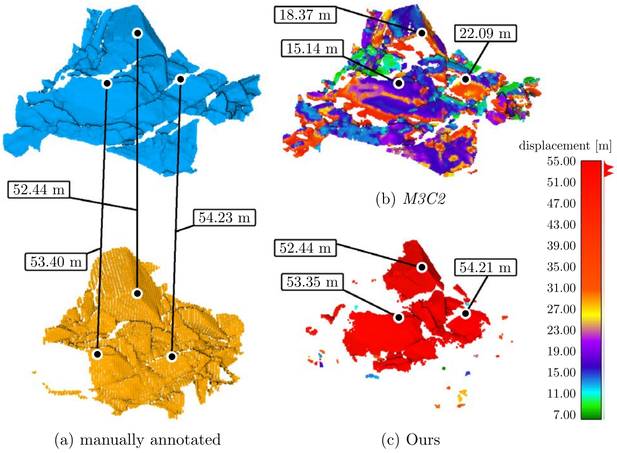
The qualitative results depicted in Fig. 13 show that our method filters out the majority of the points in the active area (> 85%), because their reliable correspondences could not be established. This is to be expected due to the large changes of the local geometry between the measurement epochs. Nevertheless, some larger patches that were not filtered out remain even in the most active areas. We show the results of one such patch in Fig. 14. For this comparison, we establish the ground truth by measuring the distance between the corresponding points, which can visually easily be recognized. Remarkably, our method can estimate correct displacement magnitudes even though the objects have moved more than 50 m between the epochs. On the other hand, the estimates of *M3C2* are very random as there is a high probability that either the displacement is not in the direction perpendicular to the surface or that due to the large motion some other point of the target epoch lies closer to the interest point.

This final analysis shows that our method has no upper bound of the estimated displacement magnitude if the local geometry does not change too much, which can in turn likely be assured for most monitoring applications by selecting a sufficiently short time interval between the measurements. In our current implementation, the tiling of the point clouds does bound the estimates, i.e., the correspondence has to lie within the same tile. The algorithm will thus not provide a displacement vector for points which move outside the tile between the analyzed epochs. Prior knowledge of the deformation process should thus be incorporated into the—possibly adaptive—tiling procedure, by introducing appropriate additional overlap between adjacent tiles and merging the results obtained from different tiles for the overlap region. We leave this improvement for future work.

Limitations and practical considerations

Despite the good performance across several datasets, our method naturally has limitations and certain consideration have to be made before using it in practice. As with other point cloud-based deformation analysis methods, the spatial resolution and temporal resolution of the point clouds are of great importance and should be determined on a per case basis. In case of rapid motion the temporal resolution should be increased in order to prevent that the scene gets too deformed between acquisitions. In case of smaller displacements, the spatial resolution might need to be increased. As shown in "Experimental evaluation" section our method is even capable of estimating displacements that are smaller than the resolution of the point clouds but a certain relation persists.

Our method also strongly relies on the distinct features in order to establish the pointwise correspondences. While false correspondences that occur in areas without distinct features (e.g., areas of cobble or debris) will be filtered out, the data gaps will remain and specifically tailored methods for such areas should be developed in



(a) manually annotated

(b) *M3C2*

(c) Ours

Fig. 14 Selected patch from the most active area of the *Moosfluh 1 year* dataset. We manually measure the distance between the (**a**) corresponding points of the source (blue) and target (orange) epoch, and compare it to the estimates of *M3C2* (**b**) and our method (**c**)

the future.[7] The same holds for man-made structures such as for example dams, which often consist of smooth surfaces. Currently, other methods such as M3C2 should be considered in such cases, especially if the dominant motion direction is orthogonal to the surface.

Summary and conclusions

In this work, we have proposed a novel deformation analysis pipeline that is capable of estimating real 3D displacement vector fields from point cloud data. To this end, we have extended the recently proposed F2S3 framework with a tiling procedure that enables efficient processing of the data in parallel and replaced all the hyperparameters with the ones derived directly from the input data. Our contributions result in a fully automated deformation analysis pipeline, which can be used to process point clouds of arbitrary size.

We evaluated the proposed pipeline on datasets of actual landslides on which our approach consistently outperforms the traditional point cloud-based deformation analysis methods. Even more, we have shown that the traditional methods that are commonly used in geomonitoring often underestimate the actual displacements, especially when motion and deformation occur parallel to the underlying surface. On the other hand, our method is also sensitive to in-plane motion and deformation and returns the true 3D displacement vectors. In turn, this opens up the possibility to derive more information about the deformation process, e.g., derive sliding surfaces from the estimated dense 3D displacement vector field.

While the focus of this paper lies in the deformation analysis, our method can easily be applied to other tasks such as point cloud registration, classification of stable/unstable regions, or other applications such as scene flow estimation in autonomous driving (Gojcic et al. 2021).

Acknowledgements

The point clouds of the Brienz TLS dataset used in this paper were collected and georeferenced by ALTAMETRIS. We thank Nicolas Ackermann (Swiss Federal Railways) for providing these data. The authors would also like to thank Robert Kenner (Institute for Snow and Avalanche Research) for providing the point clouds of the Brienz UAV dataset.

Authors' contributions

All authors read and approved the final manuscript.

Funding

Open Access funding provided by ETH Zurich.

Declarations

Competing interests The authors declare that they have no competing interests.

References

Amaral P, Malheiro A, Marques F, Moniz L, Furtado S, Loura N (2020) The use of total station for monitoring mass movements: application to Fajãzinha landslide at Flores Island (Azores Archipelago). In: Advances in natural hazards and hydrological risks: meeting the challenge. Springer, pp 59–62

Barbarella M, Fiani M (2013) Monitoring of large landslides by terrestrial laser scanning techniques: field data collection and processing. Eur J Remote Sens 46(1):126–151

Bellman R (1957) Dynamic programming. Princeton University Press, Princeton

Bentley JL (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18(9):509–517

Besl PJ, McKay ND (1992) Method for registration of 3-d shapes. In: Sensor fusion IV: control paradigms and data structures, vol 1611. International Society for Optics and Photonics, pp 586–606

Caduff R, Schlunegger F, Kos A, Wiesmann A (2015) A review of terrestrial radar interferometry for measuring surface change in the geosciences. Earth Surf Process Landf 40(2):208–228

Choy C, Park J, Koltun V (2019) Fully convolutional geometric features. In: Proceedings of the IEEE international conference on computer vision (ICCV)

Cignoni P, Rocchini C, Scopigno R (1998) Metro: measuring error on simplified surfaces. Comput Graph Forum 17:167–174

Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24(6):381–395

Friedli E, Wieser A (2016) Identification of stable surfaces within point clouds for areal deformation monitoring. In: 3rd Joint international symposium on deformation monitoring (JISDM)

Friedli E, Presl R, Wieser A (2019) Influence of atmospheric refraction on terrestrial laser scanning at long range. In: 4th Joint international symposium on deformation monitoring (JISDM), JISDM

Frukacz M, Presl R, Wieser A, Favot D (2017) Pushing the sensitivity limits of RTS-based continuous deformation monitoring of an alpine valley. Appl Geom 9(2):81–92

Girardeau-Montaut D (2019) Cloudcompare (version 2.10.1) [gpl software]. https://www.cloudcompare.org

Glabsch J, Heunecke O, Schuhbäck S (2009) Monitoring the Hornbergl landslide using a recently developed low cost GNSS sensor network. J Appl Geod 3(3):179–192. https://doi.org/10.1515/JAG.2009.019

Glueer F, Loew S, Manconi A (2020) Paraglacial history and structure of the Moosfluh landslide (1850–2016), switzerland. Geomorphology 355

Gojcic Z, Zhou C, Wegner JD, Wieser A (2019a) The perfect match: 3d point cloud matching with smoothed densities. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). pp 5545–5554

Gojcic Z, Zhou C, Wieser A (2019b) Robust pointwise correspondences for point cloud based deformation monitoring of natural scenes. In: 4th Joint international symposium on deformation monitoring (JISDM)

Gojcic Z, Zhou C, Wieser A (2020) F2S3: Robustified determination of 3d displacement vector fields using deep learning. J Appl Geod 14(2):177–189

[7] "Distinct" features are also strongly related to the spatial resolution of the point clouds. In general, with higher resolution better features can be extracted.

Gojcic Z, Litany O, Wieser A, Guibas LJ, Birdal T (2021) Weakly supervised learning of rigid 3D scene flow. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)

Häusler M, Fäh D (2018) Monitoring the slope instability at Brienz/Brinzauls using ambient vibrations and earthquake recordings. In: Swiss geoscience meeting. pp 225–226

Holst C, Schmitz B, Schraven A, Kuhlmann H (2017) Eignen sich in standardsoftware implementierte punktwolkenvergleiche zur flächenhaften deformationsanalyse von bauwerken? Eine fallstudie anhand von laserscans einer holzplatte und einer staumauer. Zeitschrift für Vermessungswesen zfv

Holst C, Janßen J, Schmitz B, Blome M, Dercks M, Schoch-Baumann A, Blöthe J, Schrott L, Kuhlmann H, Medic T (2021) Increasing spatio-temporal resolution for monitoring alpine solifluction using terrestrial laser scanners and 3d vector fields. Remote Sens 13(6):1192

Huang R, Jiang L, Shen X, Dong Z, Zhou Q, Yang B, Wang H (2019) An efficient method of monitoring slow-moving landslides with long-range terrestrial laser scanning: a case study of the Dashu landslide in the three Gorges reservoir region, China. Landslides 16(4):839–855

Huang S, Gojcic Z, Usvyatsov M, Wieser A, Schindler K (2020) Predator: registration of 3d point clouds with low overlap. 2011.13005

Kenner R, Phillips M, Limpach P, Beutel J, Hiller M (2018) Monitoring mass movements using georeferenced time-lapse photography: Ritigraben rock glacier, western Swiss Alps. Cold Reg Sci Technol 145:127–134

Kos A, Amann F, Strozzi T, Delaloye R, von Ruette J, Springman S (2016) Contemporary glacier retreat triggers a rapid landslide response, Great Aletsch Glacier, Switzerland. Geophys Res Lett 43(24):12–466

Krähenbühl R, Nänni C (2017) Ist das dorf Brienz-Brinzauls bergsturz gefährdet? Swiss Bull Angew Geol 2:33–47

Lague D, Brodu N, Leroux J (2013) Accurate 3d comparison of complex topography with terrestrial laser scanner: application to the Rangitikei canyon (NZ). ISPRS J Photogram Remote Sens 82:10–26

Lane SN, Westaway RM, Murray Hicks D (2003) Estimation of erosion and deposition volumes in a large, gravel-bed, braided river using synoptic remote sensing. Earth Surf Process Landf J Br Geomorphol Res Group 28(3):249–271

Lin Y, Wang C, Zhai D, Li W, Li J (2018) Toward better boundary preserved supervoxel segmentation for 3d point clouds. ISPRS J Photogram Remote Sens 143:39–47

Lindenbergh R, Pfeifer N (2005) A statistical deformation analysis of two epochs of terrestrial laser data of a lock. In: Proceedings of the 7th conference on optical 3-D measurement techniques 2

Malet JP, Maquaire O, Calais E (2002) The use of global positioning system techniques for the continuous monitoring of landslides: application to the Super-Sauze earthflow (Alpes-de-Haute-provence, France). Geomorphology 43(1):33–54. https://doi.org/10.1016/S0169-555X(01)00098-8

Malkov YA, Yashunin DA (2018) Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE Trans Pattern Anal Mach Intell 42:824–836

Monserrat O, Crosetto M (2008) Deformation measurement using terrestrial laser scanning data and least squares 3d surface matching. ISPRS J Photogram Remote Sens 63(1):142–154

Neuner H, Holst C, Kuhlmann H (2016) Overview on current modelling strategies of point clouds for deformation analysis. AVN Allg Vermess Nachr 123:328–339

Niethammer U, James M, Rothmund S, Travelletti J, Joswig M (2012) Uav-based remote sensing of the Super-Sauze landslide: evaluation and results. Eng Geol 128:2–11

Poiesi F, Boscaini D (2021) Distinctive 3D local deep descriptors. In: IEEE Proceedings of int'l conference on pattern recognition

Rusu RB, Blodow N, Beetz M (2009) Fast point feature histograms (FPFH) for 3D registration. In: IEEE international conference on robotics and automation (ICRA)

Strozzi T, Delaloye R, Kääb A, Ambrosi C, Perruchoud E, Wegmüller U (2010) Combined observations of rock mass movements using satellite SAR interferometry, differential GPS, airborne digital photogrammetry, and airborne photography interpretation. J Geophys Res Earth Surf 115(F1):1–11

Teza G, Galgaro A, Zaltron N, Genevois R (2007) Terrestrial laser scanner to detect landslide displacement fields: a new approach. Int J Remote Sens 28(16):3425–3446

Tombari F, Salti S, Di Stefano L (2010) Unique signatures of histograms for local surface description. In: European conference on computer vision (ECCV)

Wagner A (2016) A new approach for geo-monitoring using modern total stations and RGB+D images. Measurement 82:64–74. https://doi.org/10.1016/j.measurement.2015.12.025

Wujanz D, Krueger D, Neitzel F (2013) Defo scan++: surface based registration of terrestrial laser scans for deformation monitoring. In: Joint international symposia on deformation monitoring (JISDM)

Wunderlich T, Niemeier W, Wujanz D, Holst C, Neitzel F, Kuhlmann H (2016) Areal deformation analysis from TLS point clouds—the challenge. AVN Allg Vermess Nachr 123:340–351

Ye X, Kaufmann H, Guo X (2004) Landslide monitoring in the three gorges area using D-InSAR and corner reflectors. Photogram Eng Remote Sens 70(10):1167–1172

Yi KM, Trulls E, Ono Y, Lepetit V, Salzmann M, Fua P (2018) Learning to find good correspondences

Zeng A, Song S, Nießner M, Fisher M, Xiao J, Funkhouser T (2017) 3dmatch: Learning local geometric descriptors from RGB-D reconstructions. In: CVPR

Zan Gojcic (✉) · Lorenz Schmid · Andreas Wieser
ETH Zurich, Stefano-Franscini-Platz 5, Zurich, Switzerland
Email: zgojcic@ethz.ch