2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

City-scale Scene Change Detection using Point Clouds

Zi Jian Yew[1] and Gim Hee Lee[1]

***Abstract*— We propose a method for detecting structural changes in a city using images captured from vehicular mounted cameras over traversals at two different times. We first generate 3D point clouds for each traversal from the images and approximate GNSS/INS readings using Structure-from-Motion (SfM). A direct comparison of the two point clouds for change detection is not ideal due to inaccurate geo-location information and possible drifts in the SfM. To circumvent this problem, we propose a deep learning-based non-rigid registration on the point clouds which allows us to compare the point clouds for structural change detection in the scene. Furthermore, we introduce a dual thresholding check and post-processing step to enhance the robustness of our method. We collect two datasets for the evaluation of our approach. Experiments show that our method is able to detect scene changes effectively, even in the presence of viewpoint and illumination differences.**

I. INTRODUCTION

3D point clouds reconstructed from image-based Structure-from-Motion (SfM) are often frozen in time and thus gradually loses its ability to model the constantly changing environment with high fidelity. The first step towards maintaining an up-to-date city-scale 3D model is to detect changes in the geometric structure of the scene, while excluding other nuisance factors such as appearance changes from illumination or viewpoint differences. Detecting temporal changes in a city is an important problem, with many applications such as maintaining updated maps for autonomous driving systems [1], surveillance [2], and disaster damage assessment [3].

One naïve way of computing the changes is to directly compare images between the two traversals using some variant of image differencing [4], [5]. However, such approaches are sensitive to illumination differences between the two acquisitions. In addition, they require near pixel-perfect alignment to work well which can be hard to achieve on a moving camera. Several works [1], [6] tackle this problem using a 3D model of the scene. These typically reconstruct a dense 3D model and recover the camera poses using SfM and multi-view stereo (MVS) techniques. The dense models can later be used for dense image alignment [1] or to reproject pixels between images to detect inconsistencies [6]. However, obtaining accurate camera poses across traversals can be challenging. GNSS errors can often be several meters in urban environments; SfM techniques can also fail due to appearance differences which can be due to illumination differences or even large scene changes. In addition, reconstructing the dense model is computationally expensive and dense reconstructions are arguably unnecessary for many applications such as localization of autonomous vehicles [7].

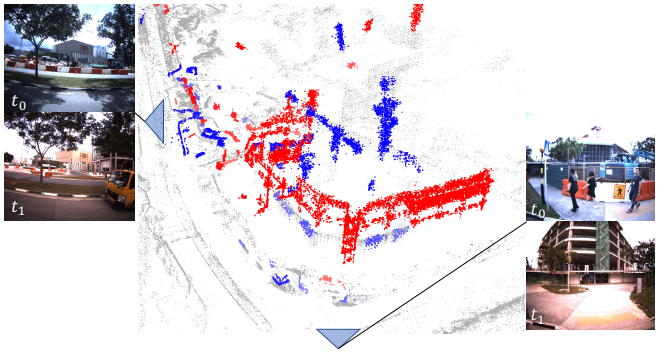

Fig. 1. Visualization of changes detected in the Business District using our approach. Blue and red indicate points which disappeared or appeared respectively (i.e. only present during t_0 or t_1). We also show images capturing the changed scene. Our approach detects the appearance of a new building as well as the disappearance of cranes and road barriers.

To avoid these problems, we register and detect changes using the 3D point clouds instead of registering images across traversals. We first reconstruct sparse point clouds separately for each traversal using SfM [8]. The reconstruction can be fairly robust since each traversal is captured within a short timespan and contains limited scene and illumination changes. The reconstructed point clouds from the two traversals are geo-registered using GNSS/INS data, but may not align perfectly due to GNSS/INS and reconstruction inaccuracies. We tackle this by performing a non-rigid registration to warp one of the point clouds to the other.

The changes can now be detected by comparing the two point clouds. Comparing point clouds generated from SfM comes with its own challenges. In particular, reconstructions of the same scene at different times may vary significantly due to variations in imaging conditions. To alleviate this issue, we employ a dual thresholding scheme where we compare between subsampled and original point clouds to detect changes. The point clouds are subsampled by considering only points which can be reliably observed from a larger number of images. These points are more stable and likely to be reconstructed in the other traversal.

We demonstrate the effectiveness of our approach on two datasets collected over two different areas. Each dataset contains images and GNSS/INS readings of the same route over two traversals that are roughly two months apart. Our datasets contain large changes *e.g.* building construction or demolition, and smaller ones, *e.g.* vehicle movements or tree planting. Experiments show our approach can detect these

*This work is supported in part by the Singapore MOE Tier 1 grant R-252-000-A65-114.

[1]Zi Jian Yew and Gim Hee Lee are with Department of Computer Science, National University of Singapore {zijian.yew, gimhee.lee}@comp.nus.edu.sg

structural changes even in the presence of viewpoint and illumination differences. Fig. 1 shows an example of our detection result. Our contributions are as follows:

- Propose a deep learning-based non-rigid point cloud registration to align two imperfect point clouds.
- Design a point cloud comparison scheme to reliably detect changes.
- Collect two datasets of images and GNSS/INS readings to validate the effectiveness of our approach. Our reconstructed point clouds and annotated image pairs will be made available on our project webpage[1].

II. Related Work

Change Detection. Change detection methods can be broadly classified as 2D or 3D. 2D methods generally compare input images though some variant of image differencing [4], [5], and tend to be sensitive to illumination differences or misalignments between the images. To partially overcome these issues, the images are often preprocessed to remove illumination variations [9], and registered [10] to each other. Despite this, 2D methods remain sensitive to viewpoint changes as they typically require pixel perfect registration to work well. Also, most 2D methods detect appearance changes which may not correspond to actual scene changes. 3D methods make use of a known 3D scene structure or reconstruct it from the input images to better detect structural changes. Taneja *et al*. [6] assumes that a 3D model of the scene in the previous time step is available and detects changes by checking the consistency during projection between images in the later time step. Ulusoy and Mundy [11] extends this to infer changes in the 3D model itself. Sakurada *et al*. [3] foregoes the dense model and instead makes use of stereo pairs in both time steps to perform the reprojection. Another direction is to generate a spatio-temporal model from images captured at various times by incorporating time into SfM methods. [12], [13] infer the temporal ordering of images and the temporal extent of 3D points in the scene by analyzing the SfM output. Lee and Fowlkes [14] optimizes a probabilistic spatial-temporal model using expectation-maximization to simultaneously register 3D maps and infer the temporal extents of scene surfaces. Most of the above methods require accurate relative camera poses between the two times, which can be difficult to obtain especially when the scene has changed.

Change Detection using Deep Learning. More recently, several works [1], [15]–[19] apply deep learning to detect scene changes. These works learn to compare two input images to detect changes in a strongly supervised manner, requiring pixel level [1], [15], [18], [19] or patch level [17] annotations which can be highly tedious to obtain. To avoid the need for annotation, Sakurada and Okatani [20] compare normalized features extracted from an upper layer of a convolutional neural network pretrained on a image recognition task, and make use of super-pixel segmentations to obtain high resolution outputs. Despite the good performance shown by these works, these image-based methods remain sensitive to viewpoint differences since they do not consider the 3D structure of the scene. Alcantarilla *et al*. [1] partly alleviates this issue by performing a dense warp between images, but requires a computationally expensive dense reconstruction and accurate relative camera poses across the two traversals.

Point Cloud Registration. The above change detection works often require accurate image registration, which can be difficult to achieve under scene changes or illumination variations. We circumvent these challenges by generating 3D point clouds from the input and registering the point clouds instead. Point cloud registration methods can be broadly classified into 1) feature-based methods [21]–[23] which establish correspondences by matching descriptors before computing the transformation, and 2) simultaneous pose and correspondence methods [24], [25] that typically use iterative schemes to estimate both pose and correspondences. Learned variants of both feature-based [26]–[30] and simultaneous pose and correspondence [31]–[34] methods are also available. One particular work, DeepMapping [32] optimizes for the registration objective by training a neural network. Depending on the application, point clouds may undergo local deformations which requires estimating a non-rigid transformation. In this work, we extend DeepMapping to handle non-rigid deformations through the use of Gaussian Radial Basis Functions (RBFs), which have been used in many point cloud registration works [35]–[37].

III. Our Change Detection Pipeline: Overview

Fig. 2 shows our change detection pipeline. The inputs are images from two time steps and their approximate GNSS/INS poses (Section IV-A). We first use SfM to reconstruct geo-registered point clouds (Section IV-B). Inaccuracies in SfM result in deformations of the reconstructed point clouds. We remove these deformations by applying a non-rigid registration to align the two point clouds (Section V-A). Finally, the registered point clouds are compared to detect the scene changes (Section V-B). For convenience, we list important algorithm parameters in Table I.

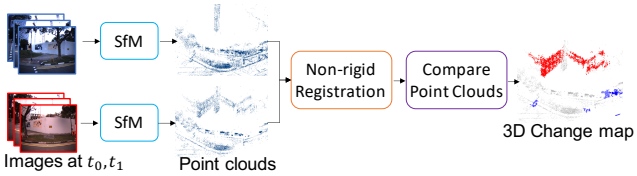

Fig. 2. Our change detection pipeline

IV. From Images to Point Clouds

A. Data Acquisition

Our data acquisition platform is a vehicle with two side-mounted wide-angle color cameras. The cameras capture images with a resolution of 2464×2056, and are set to auto-exposure to adapt to different lighting conditions. The vehicle is equipped with a GNSS/INS system which is time-synchronized to the camera system. Images are captured

[1] https://yewzijian.github.io/ChangeDet

TABLE I

TABLE OF PARAMETERS

Module	Parameter	Value
Registration	# basis points, K	36
	Max. chamfer distance δ_{reg}	10
	Regularization weight, λ_{reg}	0.01
Change detection	Subsampling track length τ_{ss}	7
	Max. chamfer distance δ_{cd}	10
	# neighbors for mean filtering, k	7
	# Min. distance for changes, τ_{cd}	2.0

every 0.6m of distance traveled. This distance was chosen taking into account reconstruction efficiency, while still allowing nearby objects to be observed from multiple images.

B. Sparse Reconstruction

We generate sparse point clouds from the images using a modified version of the COLMAP [8] SfM pipeline. Following [38], we minimize the time required for reconstruction by only matching images that are within 20m, and use the GNSS/INS readings to initialize the camera poses. Fig. 3 shows an example of the reconstructed point cloud with our Business District (BD) dataset.

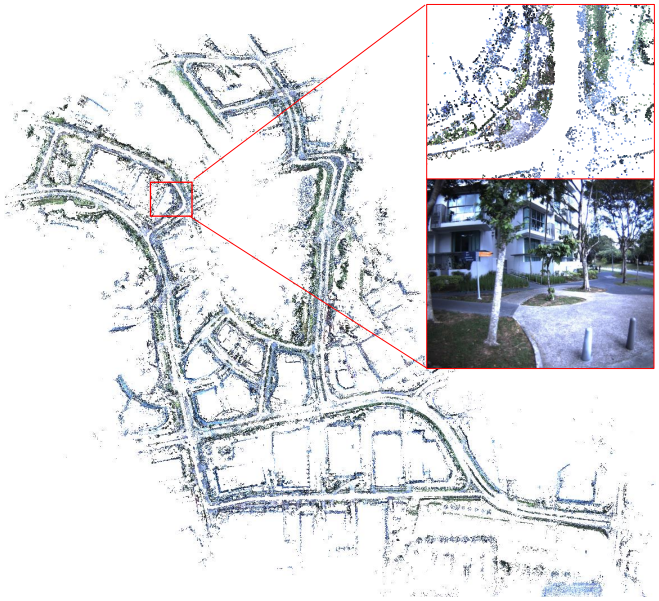

Fig. 3. Our reconstructed point cloud of the Business District (BD) dataset. We also show a zoomed in view of the point cloud and a sample image.

V. DETECTION OF CHANGES

A. Non-rigid Registration

The outputs from the previous stage are two geo-registered point clouds. However, GNSS/INS and reconstruction inaccuracies lead to local deformations in the reconstructed point clouds. The local deformations result in misalignment between the two point clouds, and they cannot be compared directly for change detection. To alleviate this problem, we perform non-rigid registration to reduce the misalignment of the two point clouds. More formally, given a reference $\mathbf{P}_{ref}$ and source $\mathbf{P}_{src}$ point clouds, the goal of non-rigid registration is to find the parameters $\boldsymbol{\theta}$ of the non-rigid warping $T(\mathbf{P}_{src};\boldsymbol{\theta})$ that warps $\mathbf{P}_{src}$ into the best alignment with $\mathbf{P}_{ref}$, i.e.

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(\mathbf{P}_{ref}, T(\mathbf{P}_{src};\boldsymbol{\theta})). \tag{1}$$

We parameterize the non-rigid warping $T(.)$ using Gaussian Radial Basis Functions (RBFs). The quality of alignment between reference and warped source point clouds $\mathcal{L}$ is given by the squared Chamfer distance, and a regularizer on the parameters. Instead of a direct minimization of $\mathcal{L}$ using solvers such as Levenberg–Marquardt, we use it as the registration loss in a deep neural network that takes $\mathbf{P}_{ref}$ and $\mathbf{P}_{src}$ as inputs and outputs the optimal $\boldsymbol{\theta}$. We now describe these components in more detail.

1) Gaussian RBF model: Gaussian Radial Basis Functions (RBFs) have been used for warping images [39] and point clouds [35], [36] due to its implicit smoothness. The smoothness property is important as it discourages warpings which are too arbitrary. A Gaussian RBF with K basis points maps a 3D point $\mathbf{x}_i \in \mathbb{R}^3$ to its target position $T(\mathbf{x}_i;\boldsymbol{\theta}) \in \mathbb{R}^3$ with the following function[2]:

$$T(\mathbf{x}_i;\boldsymbol{\theta}) = \mathbf{x}_i + \phi(\mathbf{x}_i)\cdot\mathbf{W}, \tag{2}$$

where $\mathbf{W}$ is the $K\times 3$ warping coefficient matrix. $\phi(x_i)$ is the RBF kernel, and is a $1\times K$ vector for each point $\mathbf{x}_i$:

$$\phi(\mathbf{x}_i) = \begin{bmatrix} g(\|\mathbf{x}_i - \mathbf{c}_1\|) & \dots & g(\|\mathbf{x}_i - \mathbf{c}_K\|)\end{bmatrix}, \tag{3}$$

where $\mathbf{c}_{1\dots K}$ denotes the anchor centers for the warp, $\|\cdot\|$ denotes the ℓ^2 norm and the kernel function is a Gaussian form, i.e. $g(t) = e^{-t^2/\sigma^2}$. Intuitively, the anchor centers $\mathbf{c}_i$'s control the regions to warp, and $\mathbf{W}$ controls the magnitude and direction of the warp around each of these regions. Since the 3D point clouds are reconstructed from images captured from a car moving on the ground, drifts typically occur along the x and y directions. Therefore, we only consider the distances along the xy plane when computing the kernel values. Putting the Gaussian RBF into Eq. 1, the goal now becomes finding the optimal RBF parameters $\boldsymbol{\theta} = \{\mathbf{c}_1, ..., \mathbf{c}_K, \sigma_i, ..., \sigma_K, \mathbf{W}\}$. Assuming spherical covariances for the Gaussians, there are a total of $2K + K + 3K = 6K$ parameters to be estimated.

2) Registration Loss $\mathcal{L}$: To encourage the alignment of the points, we minimize the squared Chamfer distance between the reference $\mathbf{P}_{ref}$ and the transformed source $\mathbf{P}'_{src} = T(\mathbf{x}_i;\boldsymbol{\theta})$ point clouds. The squared Chamfer distance between two point clouds $\mathbf{X}, \mathbf{Y}$ is defined as:

$$\begin{aligned}\mathcal{L}_{CD}(\mathbf{X},\mathbf{Y}) =& \frac{1}{|\mathbf{X}|}\sum_{\mathbf{x}\in\mathbf{X}} \rho_{reg}(\min_{\mathbf{y}\in\mathbf{Y}} \|\mathbf{x}-\mathbf{y}\|^2) \\ &+ \frac{1}{|\mathbf{Y}|}\sum_{\mathbf{y}\in\mathbf{Y}} \rho_{reg}(\min_{\mathbf{x}\in\mathbf{X}} \|\mathbf{x}-\mathbf{y}\|^2),\end{aligned} \tag{4}$$

where $\rho_{reg}(\cdot) = \min(\cdot, \delta_{reg})$ clamps the Chamfer distance for robustness, and we set $\delta_{reg} = 10\text{m}^2$ in all experiments.

[2]The equation assumes no rigid/affine component for the warp, which is reasonable since the point clouds are already registered using GNSS/INS.

Despite the use of smooth Gaussian RBFs, we observe empirically that severe warping may still occur, particularly in regions with changes. We further encourage smoothness by introducing an additional term $\mathcal{L}_{reg}$ to regularize the warping by penalizing large motions of small regions:

$$\mathcal{L}_{reg}(\mathbf{W}, \sigma_{1,\dots,K}) = \frac{1}{K}\sum_{i=1}^{K}\frac{\|\mathbf{w}_i\|}{\sigma_i^2}, \tag{5}$$

where $\mathbf{w}_i$ denotes the i^{th} row of $\mathbf{W}$. The final loss is then the sum of the two losses:

$$\mathcal{L} = \mathcal{L}_{CD} + \lambda_{reg}\mathcal{L}_{reg}. \tag{6}$$

3) Optimization with Neural Network: Inspired by DeepMapping [32], we design a deep neural network for the non-rigid registration. As explained in [32], an indirect optimization using a deep network is akin to changing variables [40] and leads to an empiricially easier optimization. Our network is shown in Fig. 4. We feed $\mathbf{P}_{src}$ into a permutation invariant PointNet [41] network, which outputs the transformation parameters $\boldsymbol{\theta}$ for alignment to $\mathbf{P}_{ref}$. We omit activations for the outputs with the exception of $\sigma_{1..K}$, where we use the softplus activation to constrain them to be positive. Furthermore, instead of predicting the absolute positions for the Gaussian centers $\mathbf{c}_{1..K}$, we find it beneficial to predict the offsets from predefined positions sampled using a uniform 2D grid over the entire point cloud. To register the point clouds, we simply train the network to reduce the alignment and regularization losses in Eq. 6 using the Adam optimizer [42] with a learning rate of 5e-4. For all experiments, we set $K = 6^2 = 36$.

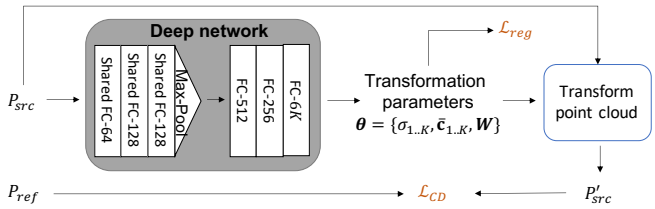

Fig. 4. Our network architecture for non-rigid registration (FC-N: denotes fully-connected layer with N output nodes).

B. Detecting Changes by Comparing Point Clouds

The reference and transformed source point clouds are compared in this step to compute the changed regions. Fig. 5 illustrates our change detection pipeline. Point clouds reconstructed through SfM contain a fair amount of noise and variation, and thus our change detection pipeline compares point clouds using a dual thresholding scheme and applies filtering to clean up the change map.

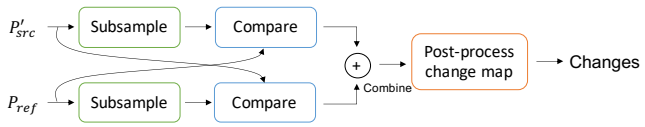

Fig. 5. Point cloud comparison pipeline

1) Dual Thresholding: Points reconstructed from SfM may vary between reconstructions due to many factors (*e.g.* illumination), which lead to false positives during comparison. To circumvent this problem, we employ a *dual thresholding* scheme, where two different sampling ratios or "thresholds" are used to detect the disappearance and appearance of points. Specifically, when detecting new points in the transformed source point cloud $\mathbf{P}'_{src}$, we compare its *subsampled* version with the *original* reference point cloud $\mathbf{P}_{ref}$. Similarly, we consider the subsampled $\mathbf{P}_{ref}$ when detecting new points in it. The points are subsampled by retaining only those with a track length above $\tau_{ss} = 7$. The motivation is that these points are observable in large number of images with different viewpoints, and are likely to be more stable and reconstructed in the other point cloud. We also detect and remove points on the ground during this stage as we find them to be less repeatable between reconstructions.

2) Point Cloud Comparison: We compute the change response by considering the distances between points in the two point clouds. Specifically, the change response $C(\mathbf{x})$ of a point $\mathbf{x} \in \mathbf{P}'_{src}$ describes how likely it is a changed point not present in $\mathbf{P}_{ref}$ and is given as the distance to the closest point in $\mathbf{P}_{ref}$:

$$C(\mathbf{x}) = \rho_{cd}(\min_{\mathbf{y}\in\mathbf{P}_{ref}} \|\mathbf{x}-\mathbf{y}\|), \tag{7}$$

where $\rho_{cd}(\cdot) = \min(\cdot, \delta_{cd})$ clamps the distance for robustness, and we set $\delta_{cd} = 10$m. To increase sensitivity, we only consider point pairs with similar normals within $40°$ of each other. Points in $\mathbf{P}_{ref}$ which are not present in $\mathbf{P}_{src}$ can be detected using a similar formulation, but this time considering the downsampled $\mathbf{P}_{ref}$.

3) Post-processing of Change Map: We post-process the change map in this step to filter out spurious changes. We first smooth out the change responses by applying a kNN-based mean filter to improve robustness against small misalignments: for each point $\mathbf{x}$, we replace its change response $C(\mathbf{x})$ by the average response of itself and its $k = 7$ nearest neighbors. Points with low change responses below $\tau_{cd} = 2.0$m after filtering, as well as isolated points with few neighboring points having large change responses are removed. Subsequently, we filter out points which are not observed in the field of views (FOVs) of the cameras in the other traversal. This step is important to handle situations where points are missing simply because the camera trajectory has changed (Fig. 6). Lastly, we repopulate the changed points with nearby points from the point cloud before the downsampling step in Section V-B.1.

VI. EXPERIMENTAL RESULTS

We collect two datasets from a Business District (BD) and Research Town (RT), respectively to test our approach. Each dataset contains data from two traversals that are two months apart and cover roughly 0.3 square kilometers. BD contains many high rise buildings, while RT contains mostly low rise buildings. Despite being captured just two months apart, construction and demolition activities lead to several

Fig. 6. The camera does not observe the top of the building in the first traversal. Without considering the camera FOVs, such regions (highlighted in magenta) will be erroneously detected as changed regions.

large structural changes in the scene. Additionally, smaller changes such as planting/removal of trees, and movement of vehicles and cranes are present in the dataset. We randomly select a total of 30 image pairs that contain changes and annotate pixel-level structural changes between them.

A. Non-rigid registration

We first show in Fig. 7 the output of our non-rigid registration algorithm. We observe that the alignment of buildings and roads in the scene improve significantly after registration. In Fig. 8, we also show the effectiveness of using neural networks for optimizing for the non-rigid parameters $\boldsymbol{\theta}$. Despite using the same momentum-based optimizer [42] and learning rate, the neural-network-based optimization converges in a smaller number of steps to a better minima compared to direct optimization.

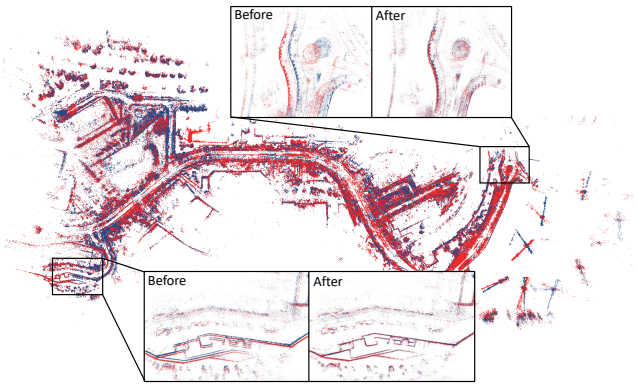

Fig. 7. Qualitative results of non-rigid registration output (Research Town). We also show zoomed-in views of the point clouds before and after registration. Best viewed in color.

B. Change Detection

Figs. 1 and 9 show qualitative examples of our change detection output. Our algorithm outputs sparse 3D point clouds that show the changed locations, and distinguishes between the disappearance and appearance of scene structures. Compared to algorithms which show the changes on images, our point cloud outputs are useful in giving a quick overview of changed regions in large scenes.

We also compare with two recent deep learning-based change detection works [15], [16]. We initialize the network weights with the weights trained[3] on the publicly

[3]Pre-trained weights are provided by the authors for [15]. For [16], we pre-train the network using the provided training code.

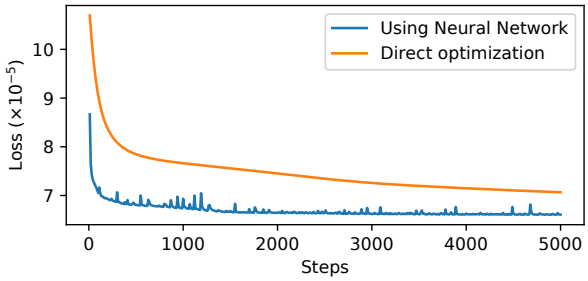

Fig. 8. Registration loss during optimization. Indirect optimization with a neural network mostly converges after about 1500 steps. In contrast, direct optimization of registration parameters using the same optimizer gives a higher loss even after 5000 steps.

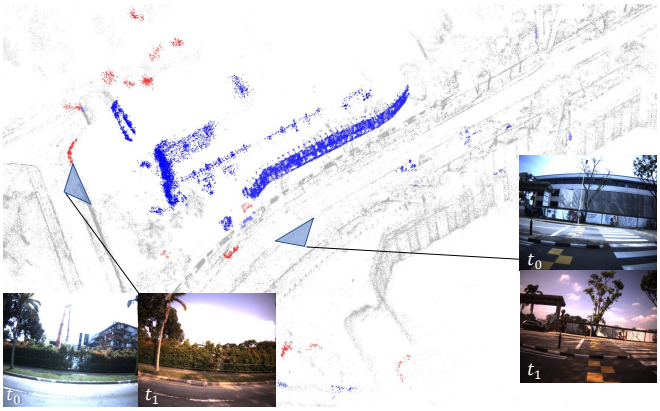

Fig. 9. Visualization of the changes detected on the point clouds (Research Town). Blue and red indicate points which disappeared or appeared respectively. We also show images capturing the changed scene.

available TSUNAMI dataset [20]. We then finetune the network weights on our dataset using a 5-fold cross validation scheme, where each fold uses a 18/6/6 train/val/test split. We evaluate the 2D change maps using the mean intersection-over-union (mIOU) as suggested in [1], [16]. Since the evaluation is performed on 2D images, we generate 2D change maps for our algorithm by projecting detected changed points within 100m of the camera location onto the image plane, and label all pixels within a radius of 20 pixels around each projected point as changed. Note that since we do not compute dense models, our approach is unable to account for occlusion of detected changes in the images, and this may lead to additional false positives during evaluation.

TABLE II
MIOU METRICS ON OUR DATASET

Method	BD	RT	(All)
CosimNet [15]	0.570	0.582	0.576
CSCDNet [16]	0.589	0.542	0.567
Ours	**0.597**	**0.633**	**0.613**

The results are shown in Table II. We also show qualitative comparisons of our detected changes in Fig. 10. The viewpoint differences in our dataset makes it highly challenging for the image-based change detection algorithms. In contrast, our algorithm compares 3D point clouds and can better handle viewpoint differences. As a result, our approach

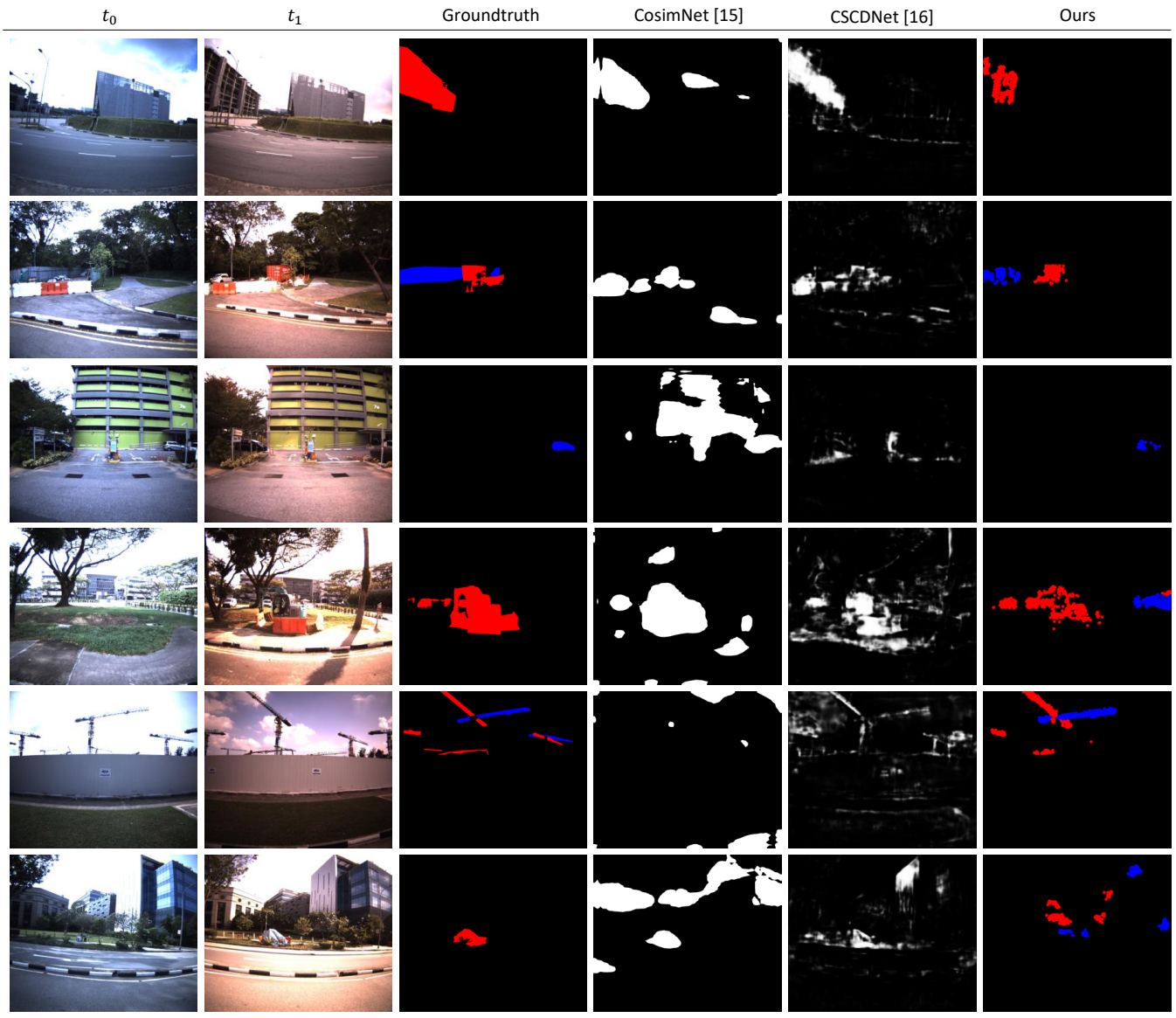

Fig. 10. Examples of change detection results, and images from the two traversals. For the groundtruth and our algorithm, blue and red denote the disappearance and appearance of scene content from time $t_0 \rightarrow t_1$. The image-based baseline algorithms [15], [16] are unable to differentiate between appearance/disappearance and all detected changes are drawn in white. Rows 1-3 are from the business district, and row 4-6 are from the research town. The last row shows a failure case by our algorithm. Best viewed in color.

outperforms image-based algorithms despite not accounting for occlusions during image projection. The last row in Fig. 10 shows a typical failure case of our algorithm, where strong lighting differences (*e.g.* in the building regions) as well as the complex foliage structures lead to inconsistent reconstructions and subsequently false alarms.

VII. Limitations of Our Approach

Although our approach outperformed existing image-based algorithms, it currently suffers from several limitations. Our approach requires point clouds of both time steps to be reconstructed which can be time consuming in larger scenes, although the subsequent registration and change detection steps can be performed reasonably fast. For example, our BD dataset requires 3 days, 25min, and 2min respectively for the reconstruction, registration, and change detection steps. Furthermore, since our algorithm operates on the output of SfM, structures which are hard to reconstruct, *e.g.* foliage lead to many false positives. Smaller scene structures or objects without distinct features are also likely to be missed as they tend to be not well reconstructed in the point cloud.

VIII. Conclusions

We propose a workflow to perform detect structural changes in large scenes by comparing sparse point clouds. This is useful for applications where sparse point clouds are available, *e.g.* maps generated for self driving cars, and our algorithm can detect regions in the point clouds that need to be updated. Experiments on two datasets with viewpoint differences show the feasibility of our approach.

REFERENCES

[1] P. F. Alcantarilla, S. Stent, G. Ros, R. Arroyo, and R. Gherardi, "Street-view change detection with deconvolutional networks," *Autonomous Robots*, 2018. DOI: 10.1007/s10514-018-9734-5.

[2] P. Sidike, A. Essa, F. Albalooshi, V. Asari, and V. Santhaseelan, "Automatic building change detection in wide area surveillance," in *2015 National Aerospace and Electronics Conference (NAECON)*, 2015, pp. 54–57.

[3] K. Sakurada, T. Okatani, and K. Deguchi, "Detecting changes in 3d structure of a scene from multi-view images captured by a vehicle-mounted camera," in *Computer Vision and Pattern Recognition (CVPR)*, 2013.

[4] P. Rosin, "Thresholding for change detection," in *International Conference on Computer Vision*, 1998.

[5] A. Singh, "Digital change detection techniques using remotely-sensed data," *International Journal of Remote Sensing*, 1989. DOI: 10.1080/01431168908903939.

[6] A. Taneja, L. Ballan, and M. Pollefeys, "Image based detection of geometric changes in urban environments," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, IEEE, 2011, pp. 2336–2343.

[7] T. Shi, S. Shen, X. Gao, and L. Zhu, "Visual localization using sparse semantic 3d map," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 315–319.

[8] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: A systematic survey," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 294–307, 2005.

[10] L. G. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, 1992. DOI: 10.1145/146370.146374.

[11] A. O. Ulusoy and J. L. Mundy, "Image-based 4-d reconstruction using 3-d change detection," in *European Conference on Computer Vision (ECCV)*, Springer, 2014, pp. 31–45.

[12] G. Schindler and F. Dellaert, "Probabilistic temporal inference on reconstructed 3d scenes," in *CVPR*, 2010. DOI: 10.1109/CVPR.2010.5539803.

[13] K. Matzen and N. Snavely, "Scene chronology," in *ECCV*, 2014. DOI: 10.1007/978-3-319-10584-0_40.

[14] M. Lee and C. C. Fowlkes, "Space-time localization and mapping," in *ICCV*, 2017. DOI: 10.1109/ICCV.2017.422.

[15] E. Guo, X. Fu, J. Zhu, M. Deng, Y. Liu, Q. Zhu, and H. Li, "Learning to measure change: Fully convolutional siamese metric networks for scene change detection," *arXiv preprint arXiv:1810.09111*, 2018.

[16] K. Sakurada, M. Shibuya, and W. Weimin, "Weakly supervised silhouette-based semantic scene change detection," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[17] S. Stent, R. Gherardi, B. Stenger, and R. Cipolla, "Detecting change for multi-view, long-term surface inspection," in *British Machine Vision Conference (BMVC)*, 2015. DOI: 10.5244/C.29.127.

[18] Y. Zhan, K. Fu, M. Yan, X. Sun, H. Wang, and X. Qiu, "Change detection based on deep siamese convolutional network for optical aerial images," *IEEE Geoscience and Remote Sensing Letters*, 2017.

[19] A. Varghese, J. Gubbi, A. Ramaswamy, and P. Balamuralidhar, "Changenet: A deep learning architecture for visual change detection," in *European Conference on Computer Vision Workshops (ECCVW)*, 2018.

[20] K. Sakurada and T. Okatani, "Change detection from a street image pair using cnn features and superpixel segmentation," in *British Machine Vision Conference (BMVC)*, 2015.

[21] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 3212–3217. DOI: 10.1109/ROBOT.2009.5152473.

[22] F. Tombari, S. Salti, and L. Di Stefano, "Unique shape context for 3D data description," in *ACM Workshop on 3D Object Retrieval*, ser. 3DOR '10, Firenze, Italy: ACM, 2010, pp. 57–62, ISBN: 978-1-4503-0160-2. DOI: 10.1145/1877808.1877821.

[23] S. Salti, F. Tombari, and L. Di Stefano, "Shot: Unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, pp. 251–264, 2014.

[24] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 14, no. 2, pp. 239–256, 1992, ISSN: 0162-8828. DOI: 10.1109/34.121791.

[25] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *IEEE International Conference on Robotics and Automation (ICRA)*, 1991, 2724–2729 vol.3. DOI: 10.1109/ROBOT.1991.132043.

[26] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[27] M. Khoury, Q.-Y. Zhou, and V. Koltun, "Learning compact geometric features," in *IEEE International Conference on Computer Vision (CVPR)*, 2017, pp. 153–161.

[28] H. Deng, T. Birdal, and S. Ilic, "Ppfnet: Global context aware local features for robust 3d point matching," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 195–205.

[29] Z. J. Yew and G. H. Lee, "3dfeat-net: Weakly supervised local 3d features for point cloud registration," in *European Conference on Computer Vision*, Springer, 2018, pp. 630–646.

[30] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *ICCV*, 2019.

[31] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "Pointnetlk: Robust & efficient point cloud registration using pointnet," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7163–7172.

[32] L. Ding and C. Feng, "Deepmapping: Unsupervised map estimation from multiple point clouds," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[33] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3523–3532.

[34] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[35] B. Jian and B. C. Vemuri, "Robust point set registration using gaussian mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1633–1645, 2011.

[36] A. Myronenko, X. Song, and M. A. Carreira-Perpinán, "Non-rigid point set registration: Coherent point drift," in *Advances in neural information processing systems*, 2007, pp. 1009–1016.

[37] "A new point matching algorithm for non-rigid registration," *Computer Vision and Image Understanding*, vol. 89, no. 2, pp. 114–141, 2003, Nonrigid Image Registration, ISSN:

1077-3142. DOI: 10.1016/S1077-3142(03)00009-2.

[38] L. Heng, B. Choi, Z. Cui, M. Geppert, S. Hu, B. Kuan, P. Liu, R. Nguyen, Y. C. Yeo, A. Geiger, *et al.*, "Project autovision: Localization and 3d scene perception for an autonomous vehicle with a multi-camera system," in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 4695–4702.

[39] N. Arad, N. Dyn, D. Reisfeld, and Y. Yeshurun, "Image warping by radial basis functions: Application to facial expressions," *CVGIP: Graphical models and image processing*, vol. 56, no. 2, pp. 161–172, 1994.

[40] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A rewriting system for convex optimization problems," *Journal of Control and Decision*, vol. 5, no. 1, pp. 42–60, 2018.

[41] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *IEEE conference on computer vision and pattern recognition (CVPR)*, 2017, pp. 652–660.

[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference for Learning Representations (ICLR)*, 2015.