

FLOW COMPARE: CONDITIONAL
NORMALIZING FLOWS FOR POINT CLOUD
CHANGE DETECTION

SAMUEL J. GALANAKIS



Artificial Intelligence
Utrecht University

August 2021

Samuel J. Galanakis: *Flow Compare: Conditional Normalizing Flows for Point Cloud Change Detection* , ©August 2021

SUPERVISORS:

Remco C. Veltkamp
Ronald Poppe
Bas Boom

LOCATION:

Utrecht, Netherlands

STUDENT NUMBER:

6950388

TIME FRAME: 1/11/2020 - 15/10/2021



cyclomedia

ABSTRACT

Despite significant progress in 3D deep learning for tasks such as classification and semantic segmentation, robust change detection techniques for complex, coloured environments have not been developed. This is in part due to the absence of labelled change detection datasets and the inherent difficulty of constructing such datasets despite the abundance of unlabeled data. Flow Compare is a fully unsupervised approach that leverages expressive generative models with iterative attention trained on multi-temporal coloured point clouds. Change detection is achieved by reframing the problem as anomaly detection given a learnt conditional distribution. Training pairs are formed by co-registered multi-temporal extracts from coloured point cloud scenes. The inherent class imbalance due to the rarity of semantically important change, which is problematic for supervised approaches, is here harnessed to guarantee that relevant changes are considered anomalies under the learnt distribution. This approach shows promise in detecting not only geometric change but also colour change whilst being robust to common semantically unimportant change.

ACKNOWLEDGEMENTS

Throughout the writing of my thesis, I have received a great deal of support.

I would first like to thank my supervisor, Remco for his valuable guidance and continued support. I would also like to thank Tao for his technical assistance, particularly regarding early dataset construction.

I could not have completed this thesis without the weekly meetings with Cyclomedia. My utmost thanks to Bas, Dan and Zill for the invaluable support and advice. I am especially thankful for helping me work through and refine often questionable ideas in the initial research stages. Furthermore, I greatly appreciated Mursit's help whenever I had any technical issues with the company systems and general logistics.

In addition, I would like to thank my parents for supporting me throughout, as they always have. Finally, my time spent would have been much duller without my friends whose company I am very fortunate to have had.

Samuel Galanakis
Utrecht, August 2021

CONTENTS

1	INTRODUCTION	1
2	BACKGROUND	4
2.1	Normalizing Flows	4
2.1.1	Coupling Blocks	4
2.2	Flow Expressivity & Extensions	5
2.2.1	SurVAE Flows	6
2.2.2	Augmented Normalizing Flows	6
2.2.3	Continuously Indexed Normalizing Flows	7
2.3	Attention	7
2.3.1	Dot-Product Attention	7
2.3.2	Cross & Self Attention	8
3	LITERATURE REVIEW	9
3.1	Change Detection	9
3.2	Deep learning on point clouds	9
3.3	Point Cloud Classification	10
3.4	Generative models for point clouds	11
4	METHODOLOGY	13
4.1	Change detection as Anomaly Detection	13
4.2	Data & Pre-processing	14
4.3	Conditioning	15
4.4	Model Architecture	16
4.4.1	Full Model	18
4.5	Voxel Dataloader	19
5	EVALUATION	22
5.1	Method of Evaluation	22
5.2	Experiments	22
5.3	Qualitative Assessment	23
6	DISCUSSION	30
6.1	Regarding Evaluation Method	30
6.2	Unwanted Bias Through Training Regime	30
6.3	Attention Maps	31
6.4	Generation Examples	31
6.5	Continuously Indexed Normalising Flows	32
7	CONCLUSION & FUTURE WORK	35
7.1	Conclusion	35
7.2	Future Work	36
	BIBLIOGRAPHY	39

INTRODUCTION

Cyclomedia is a company specializing in the large-scale and systematic visualization of environments based on 360° panoramic photographs (Cycloramas) enhanced by AI-powered analytics. Data is captured through a proprietary recording system (depicted by figure 2), mounted to the roof of each vehicle with data capture occurring at 5-meter intervals. At each location, 5 images are subsequently stitched together, and LiDAR data is captured. This data is then leveraged to provide services to entities including but not limited to local governments, utility companies, communication providers, and transportation departments.

Taking for example the case of transportation, reliable data capture replaces the need for costly in-person inspections of assets such as street signs and lamps with remote inspection. While this is a great improvement it still requires individual inspection of the digital representations. Fully automated change detection would allow only assets flagged as significantly changed to be individually inspected. Since in most cases the majority of assets do not exhibit significant change this results in substantial efficiency gains. Geo-databases may also be updated automatically with the full history of objects being tracked over successive scans.

The majority of research regarding change detection has been focused on the image domain. Specifically, repeated data acquisition by remote sensing technologies such as satellites and aircraft have been leveraged to identify changes of interest. Examples include landscape monitoring [26], large scale urban change detection and natural disasters [22].

Most recent research in this field leverages the recent developments in deep learning, utilizing Convolutional Neural Networks (CNN) to perform change detection. This is done in both a supervised [14, 50] and unsupervised manner [38, 82]. A comprehensive overview of change detection for images can be gained from the following reviews [8, 41, 40, 6, 4].

Thanks to the increased availability of mobile mapping systems (MMS) change detection for 3D data such as point clouds has garnered increased interest. 3D change detection has been applied to a wide variety of cases including topography [34], natural disasters [18, 64], at small scale [32] and for buildings [5].

Of special interest is change detection for dense urban scenes at the street level. However, conducting change detection in such an environment has significant challenges. These include complex 3D geom-

etry, occlusion, and density variations. Furthermore, the concept of change in such an environment is not semantically straightforward, especially with the inclusion of the colour space. Ideally, geometry wise the model should be able to detect changes such as the addition of signs to posts, significant deformation, replaced objects while ignoring changes such as the growth of vegetation and noise due to the data collection process. Colour-wise, the model should detect changes in posters, painted buildings, and graffiti while ignoring changes due to weather, time of day, and shadows.

The area of 3D to 3D change detection in complex environments such as street scenes does not yet, to the best of our knowledge, have robust methods that can distinguish both colour space, geometric changes and has not yet seen improvements due to the application of deep learning as seen in the image domain. This is likely due to the absence of comprehensive annotated datasets that are required for supervised learning. Such datasets are hard to construct due to the work-intensive process of labelling multi-temporal 3D data, often ill-defined semantics concerning change and the inherent class imbalance between change and no change examples. Additionally, given the variability of urban environments datasets would have to be specialized to the target area, further incentivising an unsupervised approach.

This paper aims to leverage the latest developments in 3D deep learning in combination with likelihood-based generative modelling to tackle the deficiencies of current approaches in an unsupervised way. Specifically, Normalizing Flow models are used to model the density of the cloud at time t_0 and then subsequently evaluate the likelihood of points at a later time t_1 . Given this framework, change detection can be viewed as a form of outlier detection under the original learned distribution.

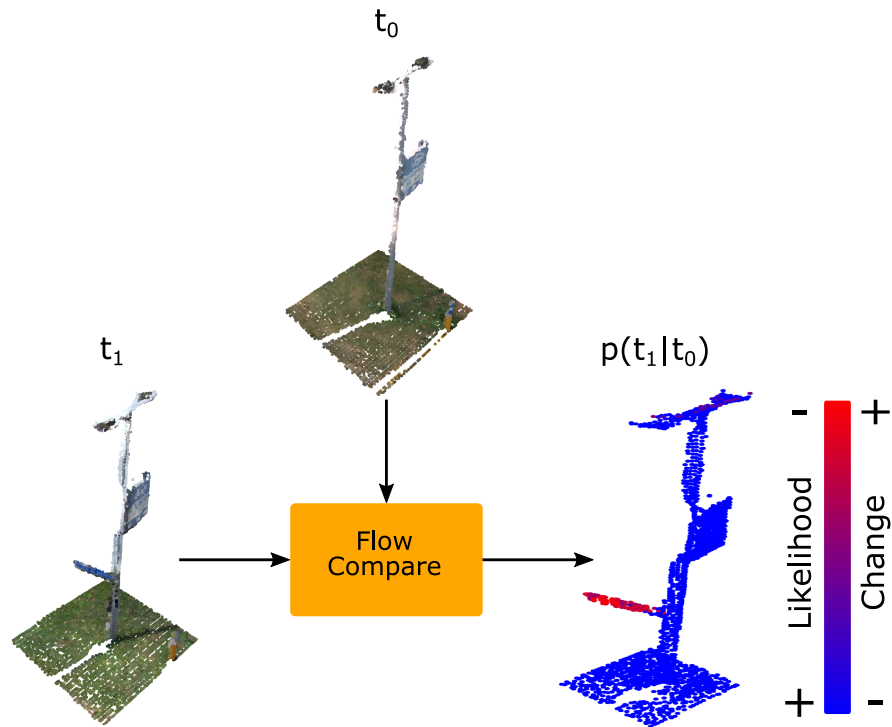


Figure 1: Flow Compare is a conditional Normalizing Flow architecture that detects changes in an unsupervised fashion through anomaly detection. Specifically, given point cloud samples at time t_0 and later time t_1 the model outputs conditional likelihood $p(t_1 | t_0)$ from which change is inferred.



Figure 2: DCR10L mobile mapping system from Cyclomedia Technology. The main components of this recording system are the cameras, positioning technology, data processing unit and the LiDAR system.

Source: <https://www.cyclomedia.com/en/product/data-capture/data-capture>

BACKGROUND

This section serves as a brief introduction to key methods and concepts that are referred to in the following sections.

2.1 NORMALIZING FLOWS

Normalizing Flows are a technique for learning the distribution of an unknown variable $x \sim p_x(x)$ through mapping from a known distribution $z \sim p_z(z)$. Since the distribution of z is predefined, it is usually chosen to be a simple distribution such as a Gaussian. In order for the transformation to be valid the total probability mass of the original pdf must be preserved. The relation between the densities is expressed by the change of variables formula, given that the transformation $f : Z \rightarrow X$ is invertible and differentiable the following holds:

$$p_x(x) = p_z(f^{-1}(x)) |\det J_f(f^{-1}(x))|^{-1} \quad (\text{likelihood})$$

$$\log p_x(x) = \log p_z(f^{-1}(x)) + \log |\det J_f(f^{-1}(x))|^{-1} \quad (\text{log-likelihood})$$

Where J_f is the Jacobian of partial derivatives of the transformation. For the transformation to be invertible, differentiable it must also be the case that dimensionality is preserved, $z, x \in \mathbb{R}^d$ and thus $J_f \in \mathbb{R}^d \times \mathbb{R}^d$. In practice, the log-likelihood is usually used whilst training a Normalizing Flow as it is equivalent and numerically preferable.

Once such a transformation has been learned, one can sample by first sampling from the known distribution $p_z(z)$ and then transforming through f . Model density can then be calculated through the change of variables formula.

In practice Normalizing Flows are usually constructed by chaining invertible, differentiable blocks parametrized by neural networks. To preserve these assumptions while still retaining a balance between computational feasibility and expressivity different architectures have been proposed including but not limited to coupling blocks [11], autoregressive models [30] and continuous-time Flows [7]. A comprehensive review has been made by Papamakarios et al. [49]

2.1.1 Coupling Blocks

Coupling blocks [11] are invertible nonlinear layers with efficient Jacobian calculation. This is achieved by splitting the input z into components z_0, z_1 and subsequently transform z_0 by an easily invertible

function whose parameters are determined by z_1 . In this way whilst the transformation of z_1 is invertible, the function that determines its parameters can be arbitrarily complex and does not need to be invertible, it is usually parameterized by a fully connected network. The Jacobian determinant of the transformation is efficiently calculated due to the triangular Jacobian matrix. A common choice for the transformation is a simple affine function involving scaling $s(z_1)$ and translation $t(z_1)$ terms 3:

$$\begin{aligned} z'_1 &= z_1 \\ z'_2 &= s(z_1) \cdot z_0 + t(z_1) \end{aligned}$$

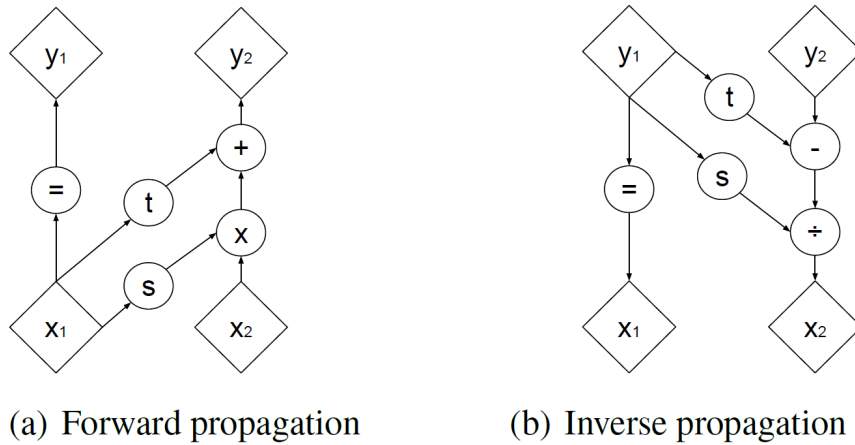


Figure 3: Affine coupling schematic reproduced from [11].

Since only a fixed subset of the input dimensions are transformed by each coupling block the dimensions must be permuted between stacked coupling blocks. Originally fixed permutations were used [11] but due to the inflexibility of these transforms learned generalizations such as invertible 1×1 convolutions [28] have been developed.

2.2 FLOW EXPRESSIVITY & EXTENSIONS

As Normalizing Flows are bijective maps, input dimensionality must be preserved. One of the most common applications of Flows is to the image domain, in such cases multi-scale architectures [11] are often employed in order to mitigate the computational cost of high-dimensional input dimension. On the contrary, in the case of point clouds the base object being modeled is a single point thus the problem is that of low dimensionality. Specifically, each colored point is of dimension 6 with dimensions split between coordinates and color. Following a strictly bijective Flow in this dimensionality limits the flexibility of the Flow. Furthermore, the topology-preserving property of homeomorphisms dictates that the (complex) topology of the

data must be preserved while being mapped to the (simple) topology of the target distribution [9]. In order to alleviate these issues both augmented Flows and continuously indexed Flows are employed.

2.2.1 *SurVAE Flows*

In order to incorporate these extensions into a single model, a unified approach is needed. Such an overarching theoretical and practical framework is provided by SurVAE Flows: Surjections to Bridge the Gap between VAEs and Flows [47]. This is a recent work that provides such a modular framework that connects VAEs and Normalizing Flows through the notion of surjective transformations. Surjective transformations here are those that are deterministic in one direction while being stochastic in the reverse. In the SurVAE framework, each transformation is characterized by a forward, inverse and corresponding likelihood contribution term. The likelihood contribution term is used in the calculation of the log-likelihood, needs to be tracked throughout the composed transformations and in the case of bijective transformations corresponds to $\log|\det J_f(f^{-1}(x))|^{-1}$. Surjective transforms that are utilised are summarily reproduced below for future reference, for detailed information one may refer to the original work.

INFERENCE TENSOR SLICING Let $f : X \rightarrow Z$ be a tensor slicing surjection that for input $X = (x_1, x_2)$ has deterministic forward $f(x) = x_1$ and stochastic inverse : $f^{-1}(z_1) = (z_1, z_2)$ where z_2 is sampled from distribution $q(z | z_1)$. Such a transform has corresponding likelihood contribution term $-\log q(x_2 | x_1)$.

GENERATIVE TENSOR SLICING Let $f : X \rightarrow Z$ be a tensor slicing surjection that for input x_1 has stochastic forward $f(x_1) = (x_1, z_2)$ where z_2 is sampled from distribution $q(z | x_1)$ and deterministic inverse : $f^{-1}((z_1, z_2)) = z_1$. Such a transform has corresponding likelihood contribution term $-\log q(z_2 | x_1)$.

2.2.2 *Augmented Normalizing Flows*

Augmented Normalizing Flows [23] address the limitations of low dimensionality and complex topologies by first embedding the data in a higher-dimensional space and subsequently using standard Normalizing Flows to model densities in that space. Specifically, independent noise $\mathcal{E} \sim q(\mathcal{E})$ is generated and then joint density $\chi \times \mathcal{E}$ is modeled. The distribution q is usually taken to be a simple distribution such as a standard normal. In the SurVae framework, such an augmented Flow may be implemented by a generative tensor slicing followed by bijections in the augmented space.

2.2.3 Continuously Indexed Normalizing Flows

Continuously indexed Normalizing Flows (CIFs) [10] address the issue of modelling targets with complicated topologies as compared to the base distribution being transformed, usually a simple Gaussian. This is done by replacing standard bijections by a continuously indexed family of bijections $F(\cdot; \mathbf{u}) : \mathcal{Z} \rightarrow \mathcal{X}$ where the bijection F can be chosen according to existing Normalizing Flow architectures. The full transformation is defined as follows:

$$\mathbf{Z} \sim P_{\mathbf{Z}}, \quad \mathbf{U} \sim P_{\mathbf{U}|\mathbf{Z}}(\cdot | \mathbf{Z}), \quad \mathbf{X} := F(\mathbf{Z}; \mathbf{U})$$

The authors propose a conditional Gaussian with parameters depending on \mathbf{Z} for $P_{\mathbf{U}|\mathbf{Z}}(\cdot | \mathbf{Z})$ and $F(\mathbf{z}; \mathbf{u}) = \mathbf{f}(e^{-s(\mathbf{u})} \odot \mathbf{z} - \mathbf{t}(\mathbf{u}))$ where s, \mathbf{t} are neural networks and \mathbf{f} is an invertible map such as for example a coupling block.

2.3 ATTENTION

The attention mechanism allows neural networks to attend to certain salient parts of available information while fading out the rest in a learnable fashion. This mechanism is at the core of transformer models, pioneered by Vaswani et al. [67] in the field of NLP and continues to be a core component of SOTA models whilst also seeing increasingly widespread use. In computer vision such mechanisms allow processing to be focused on specific images or local sub regions of a single image [2, 2]. Attention mechanisms are generally implemented as a function of query and a set of key-value pairs. Intuitively this can be thought of as generating a query, calculating how well the query matches each key and then summarizing the values accordingly.

2.3.1 Dot-Product Attention

Scaled dot-product attention [67] involves matrices Q, K, V of dimensions d_q, d_k, d_v respectively. These are usually generated via fully connected networks. Compatibility of the query and keys is calculated through dot products, implemented via matrix multiplication. The resulting values are then normalized via a softmax function and used as weights of the weighted sum of the values. Before applying the softmax the values may also be divided by $\sqrt{d_k}$ in order to avoid small gradients due to the shape of the softmax function at large input values, resulting in Scaled Dot-Product Attention.

$$A(Q, K, V) = \text{softmax}(QK^T)V \quad (\text{Dot-Product Attention})$$

$$A(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (\text{Scaled Dot-Product Attention})$$

2.3.2 *Cross & Self Attention*

Apart from the method used to calculate key-query similarity and the subsequent summarization, attention implementations can differ as to how the matrices Q, K, V are produced.

Self-attention is an attention mechanism that relates each entry in a sequence to each other in order to compute a representation of the same sequence. It has been shown to be very useful in machine reading, abstractive summarization, and image description generation.

Cross attention on the other hand is a method by which the keys are generated from a different source rather than from the sequence itself. This has been used for example in the case of modality fusion where information from one modality generates attention maps for the other modality [45, 35].

LITERATURE REVIEW

3.1 CHANGE DETECTION

Methods have been proposed to detect geometric changes in street scenes through solely imagery [37][62], and also a combination of point clouds and imagery [54, 66].

There has been limited work regarding the use of exclusively point clouds to conduct change detection for street scenes. Xiao et al. [75] detect change by comparing occupancy. Tsakiri, Anagnostopoulos [65] utilize point correspondence relations computed through nearest neighbour and iterative closest point algorithms.

There are also general distance-based approaches such as those of Palma et al. [48] and Montaut et al. [19] that can be applied to this domain. The former uses the comparison of multi-scale, context-aware shape descriptors. The latter employs an octree-based strategy, with cell wise distances being computed based on different distance metrics, the most accurate results being obtained with the Hausdorff metric.

More recently Tao Ku et al. [33] introduced a street furniture focused coloured point cloud dataset created as part of the present work. While the better performing of the two supervised methods, graph neural network-based SiamGCN performs well, creating more comprehensive datasets as would be needed for use in production would likely prove problematic. The distance-based approach PoCha-DeHH performs relatively poorly on the set challenge and due to the lack of a learning component necessarily suffers the associated limitations of handcrafted methods.

The focus on geometric change and the lack of learning-based approaches, especially when seen in the light of the recent use of deep learning in the image space provides a clear motivation for exploring different approaches in 3D to 3D change detection.

3.2 DEEP LEARNING ON POINT CLOUDS

Two key tasks of point cloud understanding are that of classification and segmentation. The former corresponds simply to a classification of the point cloud as a whole while the latter is a point-wise classification.

The adaptation of classic network architectures from 2D images to 3D point clouds is challenging due to the inherent irregularity and unordered nature of 3D points.

3.3 POINT CLOUD CLASSIFICATION

Earlier methods proposed various approaches to project the point clouds to 2D space to then utilize classical convolutional networks (CNN). MvCNN [61] was an early approach that extracts features from snapshots taken at various angles via a CNN, applies cross-view pooling, and then obtains a classification through another CNN. GVCNN improves this multi-view approach by incorporating information regarding the content of each view and then processing it according to corresponding groupings [17]. Further improvements to multi-view classification methods have been made by [80, 78].

While multi-view methods allow direct use of classical CNN networks, the projection to 2D unavoidably results in loss of information and may introduce errors due to rendering and interpolation. One way of addressing this issue is to extend the CNN architecture to 3D and split the point cloud into a volumetric grid. VoxNet [42] is an early method which maps the point cloud to a $32 \times 32 \times 32$ voxel grid and then applies a 3D CNN for classification. The volumetric approach was further improved by Vote3Deep [15] through the addition of a voting procedure.

Although volumetric methods produce respectable results they are inherently limited by the memory and computational cost of high-resolution grid representations. To improve the efficiency of this approach OctNet [56] was proposed which hierarchically partitions the point cloud through a hybrid grid-octree representation. O-cnn [70] then proposed an adaptation of 3D CNN that can be used to extract features from octrees. Despite such developments, volumetric approaches still struggle with striking a balance between efficiency and sufficient resolution.

Non-volumetric focused approaches to extending convolution to point clouds directly have also been proposed. These usually entail a mapping to a space with desirable properties followed by spectral [79, 69] or spatial [60, 39, 63] convolutional filters.

In contrast to most previous approaches that try to extend classical methods by projecting the point cloud onto a regular structure, methods that process the points directly have been the focus of most recent work. The approach was pioneered by PointNet [52]. PointNet first transforms the points by a learned affine transformation matrix to achieve invariance to geometric transformations. Subsequently, each point is passed through fully connected layers with shared weights and followed by a learned feature transformation which fulfils a role similar to the first transform in feature space. Finally, after being mapped through more fully connected layers with shared weights a max-pooling layer gives the latent representation. This method was iterated upon in PointNet++ [53] by enhancing its ability to capture

local structures. This was done by recursively applying PointNet to nested partitions of the point cloud.

PointNet++ has seen further iterations, PointWeb [81] introduced Adaptive Feature Adjustment to improve feature quality by including local neighbourhood context. Duan et al. [12] proposed the addition of a Structural Relation Network module that facilitates structural reasoning between the local neighbourhoods to further improve performance.

An alternative approach is to represent point clouds as graphs. In this case, graphs are formed by treating the points as nodes and forming edges with neighbouring points. This approach was first taken by ECC [58] where edges are formed between each neighbouring pair and subsequently edge conditioned convolution (ECC) is applied with a filter generating network such as an MLP.

DGCNN [72] proposed a neural network module dubbed EdgeConv which is a convolution like operation that is applied to local neighbourhood graphs that are dynamically generated at each layer using k-nearest neighbours.

3.3.0.1 Point Cloud Segmentation

Since 3D segmentation requires point-wise labelling the model must collect both global context and detailed local information at each point such tasks are generally more demanding than classification. Most all point classification methods can easily be applied to semantic segmentation with minimum architectural changes, semantic segmentation tasks are generally used as one of the main forms of evaluations for classification method backbones. The most common way of achieving this is concatenating the per point features with the global latent representation vector and subsequently individually classifying each point.

3.4 GENERATIVE MODELS FOR POINT CLOUDS

Early works [1, 16] in generative modelling of point clouds mostly focused on generating fixed size sets of points which limits flexibility significantly. AtlasNet [21] drops this limitation, is able to generate a varying number of points by mapping a set of 2D patches using patch-specific MLPs based on the 2D patch coordinates and a global shape embedding.

Variational AutoEncoders (VAEs) [29], Generative Adversarial Networks (GANs) [20] have long been the focus of research into generative models in general. VAEs use neural networks as function approximators and maximize a lower bound on the data log-likelihood to model a continuous latent variable with an intractable posterior distribution. GANs sidestep the need for likelihood estimation com-

pletely by exploiting an adversarial strategy between a generator and a discriminator network.

A different approach is taken by Normalizing Flows [55] which has recently been gaining attention. Normalizing Flows model a distribution through a series of learned invertible transformations which map the intractable posterior to a simple preset density. This method allows for exact log-likelihood estimation, stable training, and flexible approximate posterior distributions.

Both GANs [57, 73], VAEs [43, 46] have been used for generative modelling of point clouds. For the current work, change detection is done based on (conditional) likelihood estimation which makes Normalizing Flows an attractive choice although approximate likelihood estimation methods like VAEs offer superior flexibility due to decreased limitations on the mapping.

There has been limited research done regarding the application of these methods to point clouds. The first work is PointFlow [77] which employs a two-level hierarchy of distributions. The first level is the distribution of object shapes and the second is the distribution of points given a shape which is modelled using a Normalizing Flow.

C-Flow [51] employs two parallel Flow branches with mutual connections formed by coupling layers to learn conditional distributions and a 3D Hilbert curve projection to order and process points in an invertible manner. This approach allows for conditional density modelling and thus is suitable for tasks such as 3D reconstruction.

Stypulkowski et al. [59] propose a method for generative modeling of point clouds that combines Normalizing Flows and point encoders, specifically PointNet [52]. The model is trained by first taking two sample subsets of a given point cloud and then encoding one as a latent vector via PointNet. Subsequently, the latent vector is mapped to a normally distributed latent vector e by a Flow function g and a second Flow function f is conditioned on it. Both the Flows and the encoder are then trained end-to-end by maximizing the log-likelihood of the second sample being mapped through f . In order to sample a new shape, a latent vector e is sampled, f is conditioned on it, then points are sampled by first sampling from the prior and then mapping them through f^{-1} .

Discrete Point Flow (DPF) [68] takes a similar approach using a combination of VAEs and Normalizing Flows. Discrete Normalizing Flows are used to construct shape conditional densities and expressive latent shape priors. PointNet is used to generate the mean and diagonal covariance matrix of a Gaussian which serves as the latent shape representation.

METHODOLOGY

4.1 CHANGE DETECTION AS ANOMALY DETECTION

Change detection is simply the process of identifying differences in the state of an object through multi-temporal samples. In practice, taking for example change detection for street scenes, only a small subset of all changes that occur are of interest. Changes due to lighting, weather, plant growth etc. may not be of interest whilst removed signs or graffiti may be. Often, this subset of changes that is of interest is also much less frequent than cases of no change or change that is not of interest. Picking a sign at random and monitoring its state over multiple years it is much more likely that one will observe unimportant or essentially no change rather than significant change like damage or removal. Attempting to tackle change detection in such cases in a supervised manner becomes very challenging due to the extreme class imbalance coupled with the semantically ill-defined and practically challenging task of creating such labelled datasets.

Due to these issues, a different, unsupervised approach is taken here. Given the assumption that the vast majority of objects do not change in semantically relevant ways, the change detection problem can be viewed as a form of anomaly detection. Specifically, one can model the distribution of change over multi-temporal samples and subsequently perform outlier detection under the learned distribution. If the learned distribution was accurate, outliers under the distribution should correspond to semantically relevant cases of change. Essentially, the model learns to become indifferent to common augmentations such as lighting and only detect rare changes. This method relies heavily on the aforementioned assumption of the comparative rarity of semantically relevant change as if such cases are too common they will not be considered outliers under the learnt distribution.

In order to learn the required distributions Normalizing Flows are chosen due to the desirable ability of exact log-likelihood estimation and the fact that generative performance is not relevant. First, the dataset and preprocessing steps for the input point clouds are detailed. The following subsections focus on different model architectures, the accompanying intuition and their respective features, weaknesses.

4.2 DATA & PRE-PROCESSING

The dataset is composed of scenes collected during 2019-2020 in a 500 and 150 meter radius area for the training and test sets respectively, at the centre of Amsterdam as depicted by figure 5. The unprocessed data consists of coloured point clouds with an associated centre point (approximate sensor location) and time of capture as illustrated by figures 4, 6. Since data capture occurs every couple of meters adjacent scans from the same date are combined to create one higher quality scan with reduced occlusion and more complete coverage. After combining adjacent scans a 20x20m cloud centred at the sensor position is extracted to minimize overlap with adjacent scenes and maximize quality. The dataset consists of 2250 scenes for the training set and 550 in the test set with 2-4 scans of the same area from different dates each. Subsequently, scans of the same scene are registered using the Iterative closest point (ICP) algorithm [3]. Lastly, voxel downsampling is applied with a voxel size of 7cm, significantly reducing the point density to keep the computational requirements manageable.

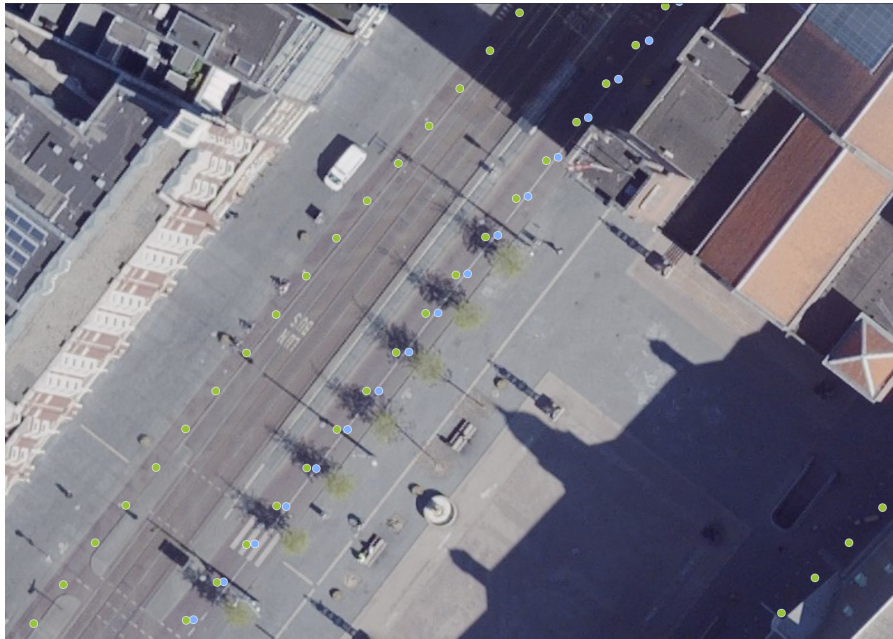


Figure 4: Aerial view of successive capture points denoted by coloured circles. Colour corresponds to the date of capture.

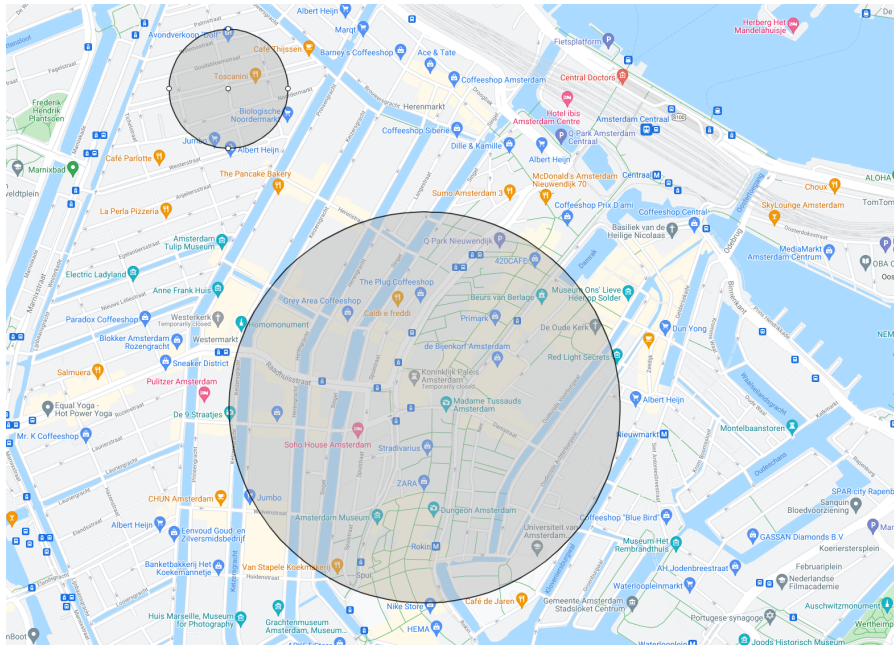


Figure 5: Aerial view of areas where dataset scenes are located. Larger and smaller radii correspond to training and test set respectively.

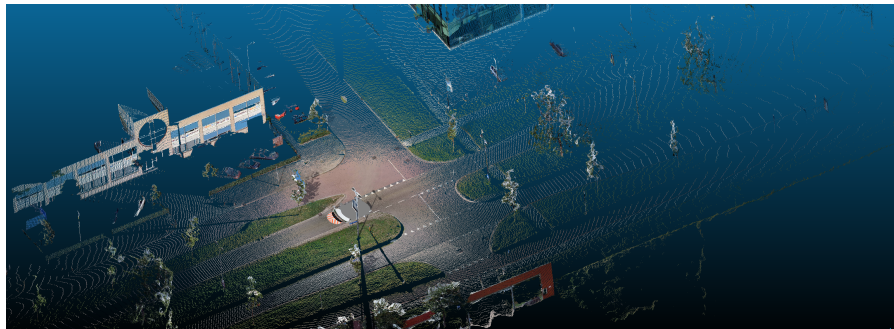


Figure 6: Example scene.

4.3 CONDITIONING

Given a dataset of point cloud pairs t_0, t_1 which correspond to the same coordinates the goal is to model the conditional probability $p(t_1 | t_0)$. Previous works that model point-wise densities [59, 31] achieve this by concatenating a latent representation z of the condition to the input of the fully connected networks that parametrize each coupling block. While this tactic may be sufficient to allow global information of the conditioned entity to be modelled it becomes problematic if fine-grained information needs to be accessed. Intuitively, given clouds t_1 that depicts a street sign, in order for the model to assign a likelihood to a given point p it must have access to not only global information (is there a sign in t_1 ?) but also detailed information regarding the points in the local neighbourhood. In order

for such fine-grained information to be available given the aforementioned method, the latent embedding z would have to encode all such information in essence resulting in a sort of compression rather than just relevant global information. In order to achieve such compression, z would have to be of high dimensionality and lead to significant computational and memory requirements as this vector is processed by each coupling block. Ideally, the processing of each point should be able to access only the information that is relevant to it and not have to process full detailed information of the whole condition. This is achieved by using a combination of segmentation-like embedding network on the conditional cloud t_0 in combination with cross-attention modules at each coupling block. The classical approach of simply concatenating a latent embedding to the input of each coupling block is also tested and referred to as the global embedding approach.

4.4 MODEL ARCHITECTURE

4.4.0.1 *Conditioning Embedder*

In the global embedding approach, an embedding z of the conditional cloud $N_0 \times 6$ is computed through a classification like architecture, namely a Dynamic Graph CNN (DGCNN) [71]. This output is then concatenated to the input of fully connected layers at each coupling block.

The cross attention approach roughly resembles that taken by [25] as far as the iterative attention mechanism is concerned although in that case there is no conditioning embedder, with the raw point clouds being used as input to each attention module. The embedder is added here in order to process the condition and aggregate relevant features at each point once so that this initial processing does not have to be repeated at each block. The architecture used for the embedder is very similar to those used for semantic segmentation as in both cases the goal is to aggregate local neighbourhood as well as global information at each point. Specifically, a Dynamic Graph CNN [71] network and a PConv [76] network with PointNet++ [53] backbone are compared. In both cases the embedder takes input condition cloud $N_0 \times 6$ and outputs point-wise embedding $N_0 \times E$. This output is then further processed by the Cross Attention Modules.

4.4.0.2 *Cross Attention Module*

Before computing the attention matrices, layer norm [36] is applied to the inputs for increased stability and efficiency. Subsequently, the cross attention module generates $K, V \in \mathbb{R}^A$ from condition embedding $N_0 \times E$ and $Q \in \mathbb{R}^A$ from the main Flow input using fully connected networks. The dimension A is the inner dimension in which

the main attention operation is performed and is set as a hyperparameter. Specifically [Scaled Dot-Product Attention](#) is used with a single head, multiple heads did not show increased performance. This module is used for all modules of the main Flow that require conditioning, weight sharing was not explored although it may be a viable option for reducing overall parameter count.

4.4.0.3 *Flow Block*

A coupling block architecture is used for the Flow blocks, parametrized by fully connected networks with residual connections. Affine coupling blocks [11] are chosen as they offer a balance between expressivity and memory requirements. Rational Quadratic Spline Flows [13] and Matrix Exponential Flows [74] were also considered but chaining a larger number of memory efficient affine layers was found to perform better. Each such Flow block is conditional, accompanied by a cross attention module that takes as input the part of the input not being transformed and outputs an attention embedding used as input to the fully connected network. In the global embedding case, the embedding is instead simply concatenated before going through the fully connected network. A sigmoid scaling function is used in place of the exponential for increased stability.

4.4.0.4 *Augmenter*

Augmented Normalizing Flows [23] address the limitations of low dimensionality and complex topologies by first embedding the data in a higher-dimensional space and subsequently using standard Normalizing Flows to model densities in that space. Specifically, independent noise $\mathcal{E} \sim q(\mathcal{E})$ is generated and then joint density $\chi \times \mathcal{E}$ is modeled.

A conditional normal distribution is chosen for q , with location and scale parameters being a function of the context, using cross attention or a global embedding as in the Flow block. The module is implemented as in the SurVae framework [47] with a generative tensor slicing resulting in a larger augmented dimension in which all following bijections are done. Experimentation showed that increasing the augmented dimension size is beneficial with a value of 300 being chosen as a balance between dimensionality and the number of layers allowed by memory limits.

4.4.0.5 *Permuter*

Since coupling blocks only transform a fixed subset of the input dimensions, they must be shuffled between stacked coupling blocks. Originally fixed permutations were used [11] but due to the inflexibility of these transforms learned generalizations such as invertible 1×1 convolutions [28] have been developed. Here 1×1 convolutions correspond to simply multiplication by a learned matrix as each point

is modelled separately. The matrix is parametrized by a learned LU Decomposition so as to avoid training instabilities associated with singular matrices.

4.4.0.6 Actnorm

Each coupling block is followed by an Actnorm layer [28] which is used as an alternative to batch normalization [24] which uses data-dependent initialization.

4.4.1 Full Model

Given cloud t_1 with corresponding context t_0 as input, t_1 is first passed through the condition embedder resulting in a $N_0 \times E$ point-wise embedding. Subsequently t_1 is (conditionally) augmented resulting in a $N_1 \times A$ output. Subsequently, the augmented input is passed through chained Flow blocks each consisting of a LU matrix permuter, an Actnorm layer and a conditional coupling block. After the last Flow block, the log-likelihood is calculated using the base distribution (standard normal) and the cumulative likelihood contribution term according to the SurVAE framework.

At test time, anomaly detection given the conditional likelihoods needs to be performed. If the distribution of conditional likelihoods outputted by the model lie in a somewhat consistent range a minimum likelihood threshold can be chosen so that lower likelihood points are considered anomalies and thus changed. The remaining points are set to o change. The threshold can be chosen by inspection or optimized given a small dataset of labelled examples. In this case all points

If the distribution of conditional likelihood values isn't consistent one can use the likelihoods of points given context from their own cloud as a comparison for anomaly detection. At test time, given clouds t_0, t_1 the following likelihoods are calculated: $p(t_0 | t_0), p(t_1 | t_1), p(t_0 | t_1), p(t_1 | t_0)$. The likelihoods $p(t_i | t_i)$ are always calculated with respect to the associated context and are needed in order to serve as a reference point for anomaly detection, with change of t_i given t_j calculated as follows:

$$C_{ij} = \begin{cases} \max(p(t_i | t_j)) - p(t_i | t_j) & p(t_i | t_j) < \mu_{j|j} - m\sigma_{j|j} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Where μ, σ correspond to the mean, standard deviation and m correspond to the multiplier parameter. The parameter m sets a minimum threshold for change at m standard deviations from the mean in the negative direction. Only a one-sided threshold is used as only low likelihood outliers need to be detected. The multiplier can be

set according to desired sensitivity, with larger values corresponding to lower sensitivity as only lower likelihood points are flagged as change. A default value of 5.4 was found to work well.

In the present case, both methods were found to perform similarly as the likelihoods were sufficiently consistent. Thus, the direct thresholding approach was taken as it requires significantly less computation. In what follows the threshold is set to 5 unless otherwise specified.

The final change values are normalized to the range 0, 1. For each pair both $C_{0|1}$ and $C_{1|0}$ are computed so as to detect change for points that are not present in both clouds such as removed or added objects. The final change map is arrived at by merging the two change maps, where changed points can be marked according to their point cloud of origin.

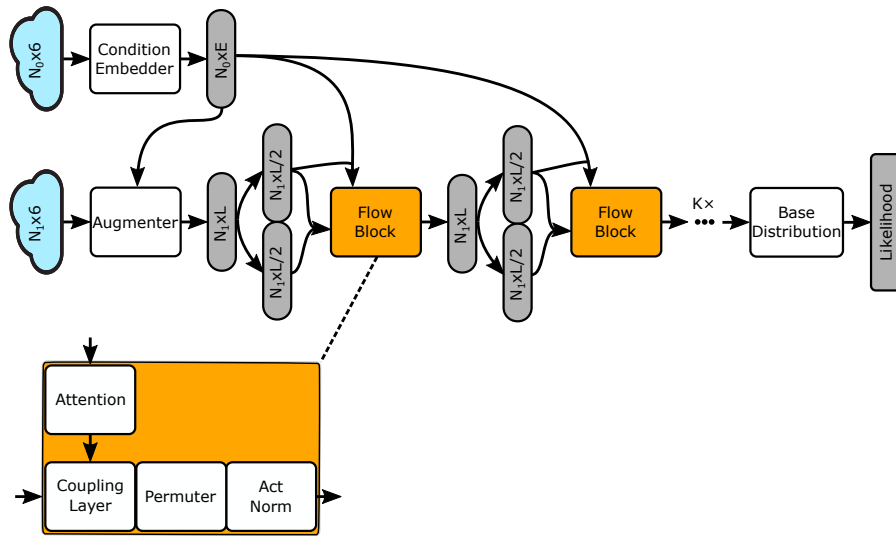


Figure 7: Flow Compare with cross attention. Cloud t_1 (being evaluated) and t_0 (context) are taken as input, t_0 is processed by the condition embedder and subsequently and serves as input to each conditional module. t_1 first goes through the augments, being augmented with conditionally generated noise to have L dimensions. Subsequently, for each Flow block, the tensor dimensions are split, half used as input to the attention along with the context and then the tensor is transformed by the coupling, permuter and act norm modules. This process is repeated K times ending with likelihood evaluation through the base distribution.

4.5 VOXEL DATALOADER

Ideally, for each point being evaluated, the context would consist of either all points within a certain distance or the k nearest neighbours. With the present model architecture, such an approach with a unique context for each point was not computationally viable and thus a voxel-based approach was taken. Given a list of clouds correspond-

ing to the same scene a common voxel grid with dimensions $(2, 2, 4)$ is imposed and each voxel is then downsampled separately using the iterative farthest point sampling (FPS) algorithm [44]. This is done as opposed to random sampling in order to achieve even coverage. When considering a pair of clouds, for each voxel a corresponding, slightly larger voxel $(2.2, 2.2, 4.2)$ with the same centre is taken from the other cloud as context and vice versa. This voxel is taken to be slightly larger so as to guarantee that points that are on the edge of the voxel have sufficient context. During training, only voxels with a minimum of 1024 points and 1250 context points are considered. Furthermore, pairs from the same cloud are also considered, with slight noise added to the coordinates which results in a 1:1 ratio of self-pairs and different time training pairs. This ratio was kept constant throughout all experiments but different values may prove advantageous. At test time only different cloud pairs are considered. Each voxel-context pair is normalized so as to be zero mean and within the unit sphere. Lastly, a randomized rotation augmentation is applied to the xy plane.

Due to the uniform placement of the voxel grid, without regard for the geometry, voxels contain partial objects and the frequency distribution of objects types is highly skewed. Specifically, the dataset is dominated by building facades with other objects such as street signs and trees making up the remaining samples. Figure 8 depicts voxel pairs that can be found in the training set. Examples a,b,d,f show common examples of colour differences due to lighting conditions and shadows. Example c shows a rare example of actual change where what appears to be a utility box has been installed. Lastly, example e depicts a case of occlusion where the pole of a traffic light isn't captured. Such cases are some extent unavoidable due to the nature of data collection in urban areas and the model may learn to at least cope with partial occlusion.



Figure 8: Example voxel pairs from the dataset.

EVALUATION

5.1 METHOD OF EVALUATION

The absence of a point-wise labelled change detection dataset and the unsupervised nature of the method means that the performance can't be quantified directly on the task at hand using standard metrics and necessitates a different evaluation approach. A combination of quantitative evaluation via log-likelihood on a test set and qualitative evaluation through hand-picked varied change detection examples is used. Log-likelihood is used as a proxy for change detection performance because intuitively a more accurate model of the conditional distributions should translate into better change detection, as long as overfitting is avoided. This intuition matches the qualitative results observed during the experiments.

Regarding model architecture ablations are performed with respect to the global vs cross attention embeddings, embedder architecture and the inclusion of extra context. Where extra context refers to the height of the voxel centre relative to the ground. This is done as the point cloud normalization step removes such information which could be useful as knowing the height of the voxel being assessed does provide certain information such as not expecting ground or cars at certain heights.

There are many other hyperparameters associated with the various modules which were set in accordance with other generative modelling approaches when possible or based on limited experimentation otherwise. Due to the novelty of the approach and the limited computational resources only certain key, hyperparameters were focused on. A more comprehensive hyperparameter optimization may lead to significantly increased performance in future works.

5.2 EXPERIMENTS

In all cases, training is performed for a total of two epochs (2 days) with an initial learning rate of 10^{-4} with the Adam optimizer [27] or until early stopping on a Nvidia A100-40GB graphics card. Detailed hyperparameter configurations and training code can be found at the associated [repository](#).

EMBEDDER	NATS (\uparrow)	PARAMETERS (M)	Extra Context
DGCNN Global	1.737	199.5	No
PACONV Attention	2.034	170.7	No
PACONV Attention	2.125	170.8	Yes
DGCNN Attention	2.144	165.1	No
DGCNN Attention	2.222	165.2	Yes

Table 1: Log-likelihood of the test set in nats (higher is better)

From the experiments, it is clear that the global embedding approach is significantly inferior despite the higher number of parameters. The parameters here are added by using a larger embedding dim $E = 124$ as opposed to 64 used for the attention approach and extending the fully connected layers that parameterize each flow block to offset the parameter count of the absent attention blocks. This is likely due to the inability of the global embedding to encode all needed local information. The attention approach sidesteps this by allowing direct, dynamic access to the local information at each flow block.

All further experiments are conducted with the use of Attention, differ with respect to embedder architecture and the use of extra context. The extra context seems to help to some extent as it increases the scores for both embedder types despite the extra parameters added being insignificant. This makes sense as knowing the relative voxel height is intuitively useful in assigning likelihoods but more numerous experiments would be needed to more precisely quantify the effect. Between the two embedder types, the DGCNN network outperforms in both cases despite having fewer parameters, it is not clear why this is the case but it may have to do with the dynamic graph creation (and thus message passing) that is present in DGCNN.

5.3 QUALITATIVE ASSESSMENT

All visualizations are done with the best performing model according to the previous experiments. Change 0 corresponds to change of voxel 0 conditioned on the corresponding context voxel in cloud 1 and vice versa. Blue denotes no change and red denotes change. Full 3D view of the chosen examples and for the rest of the test set is provided through the associated [repository](#).

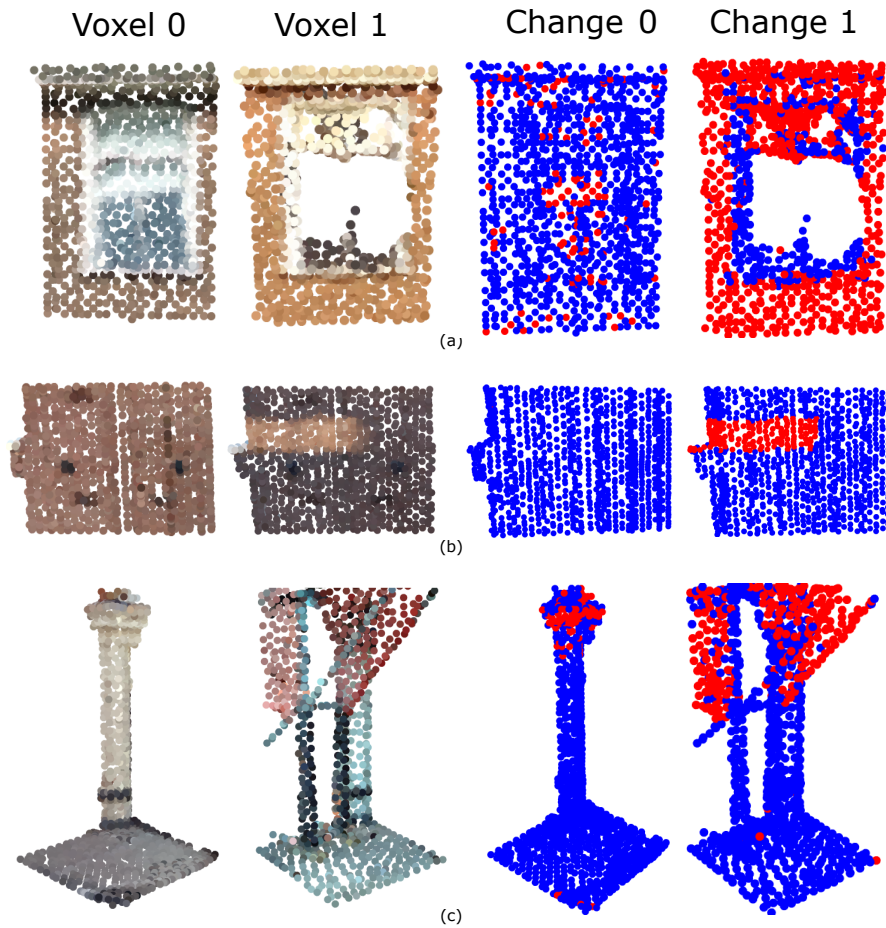


Figure 9: Change detection examples. Example a depicts a window with significant light change. Example b depicts part of a wall with significant light change with a bright patch in the second view due to the rest being shadowed. Example c depicts significant light change with the addition of a tall tent-like covering.

In figure 9, in examples a,b the model wrongly assigns change to points with dramatic lighting difference from dark to light. This is clearly problematic but is likely due to the absence of sufficient training voxel pairs where such large lighting changes are present. Given that only 2-4 scans are used per scene this is to be expected and a straightforward solution would be to use more varied multi-temporal samples to become robust to such changes. This effect is present in other examples to a lesser extent and explains some portion of the spurious change.

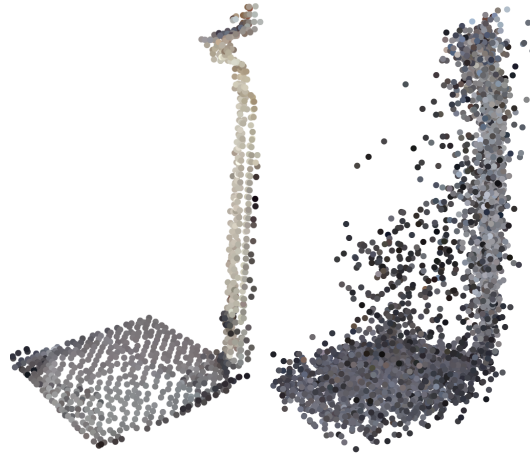


Figure 10: Generated point cloud conditioned on voxel 0 of example c, problem figure 9 with a standard deviation of 0.6

In the last example of this figure, the model is able to detect the top of the added object while ignoring change in lighting in the rest of the scene. Although the upper parts of the object are classified the lower supports are not, looking at the generation 10 gives an indication as to why. Essentially the model seems to expect points in the centre of the voxel with the same approximate colour as the missed points. Why this is the case is not obvious but it may be due to the fact that only voxels with a certain number of points are used in the training. This has the effect that very few voxels have a vacant middle section and the model thus learns to expect points of some average colour in that area by default. This could be addressed by using a padding approach to batching rather than fixed point counts and thus not filtering training voxel pairs by a minimum size.

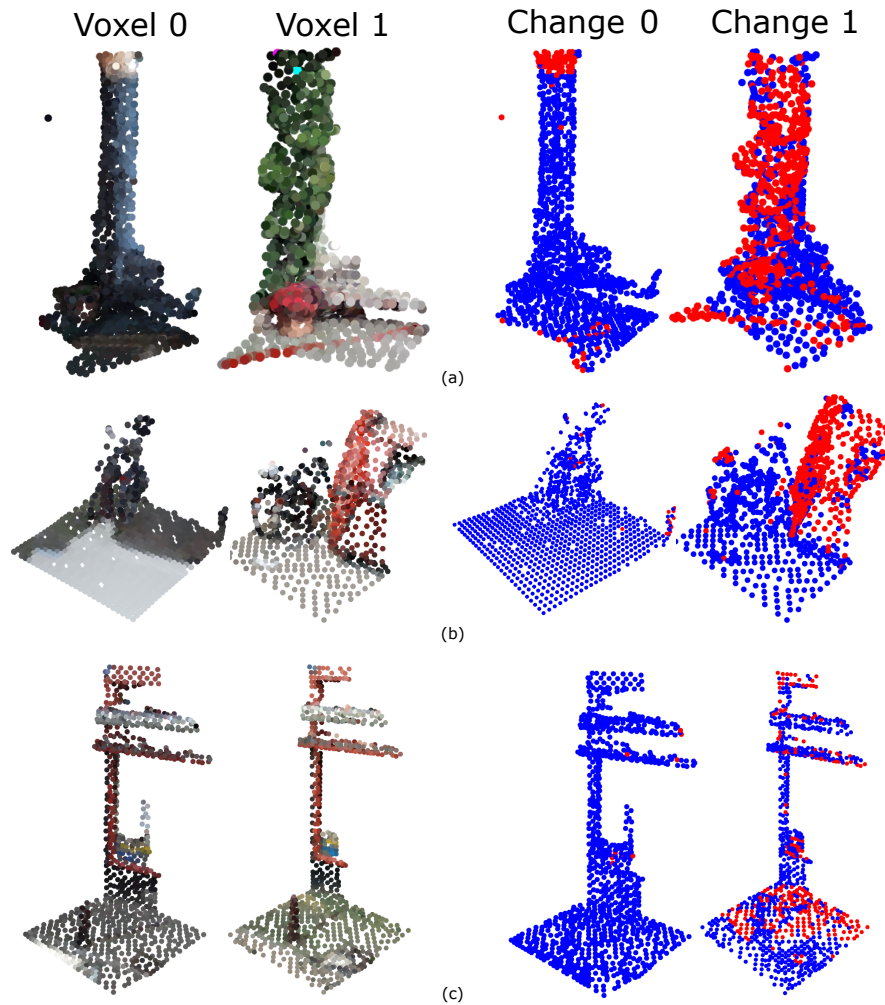


Figure 11: Change detection examples. Example a depicts a pillar and a plant pot. The former is covered by plant growth and the latter is flowering in the second voxel. Example b depicts an area where a bike is parked and the addition of a red garbage bin in the second view. Example c depicts a partial building facade and a cobblestone floor which is mossy in the second view.

In the first row of figure 11, the model is able to pick up the added plant growth on the pillar, the changes on the flowerpot and the lines in the ground whilst not assigning change values due to voxel 0 being significantly darker. Example b shows a bike in the first voxel and a bike in roughly the same area in the second voxel next to a newly added garbage bin. The model assigns change to the added bin and correctly assigns no change to the rest of the scene. In the last example, a partial building front is shown, the main difference between the two times being the change in the ground colour due to moss growing between the cobblestones. The model assigns change where the moss is present but also assigns some spurious change to the top of the facade, likely due to the aforementioned issue with bright light spots. Changes such as the appearance of moss may or may not be

of interest and the model will learn to ignore them if they are very common in the training set.

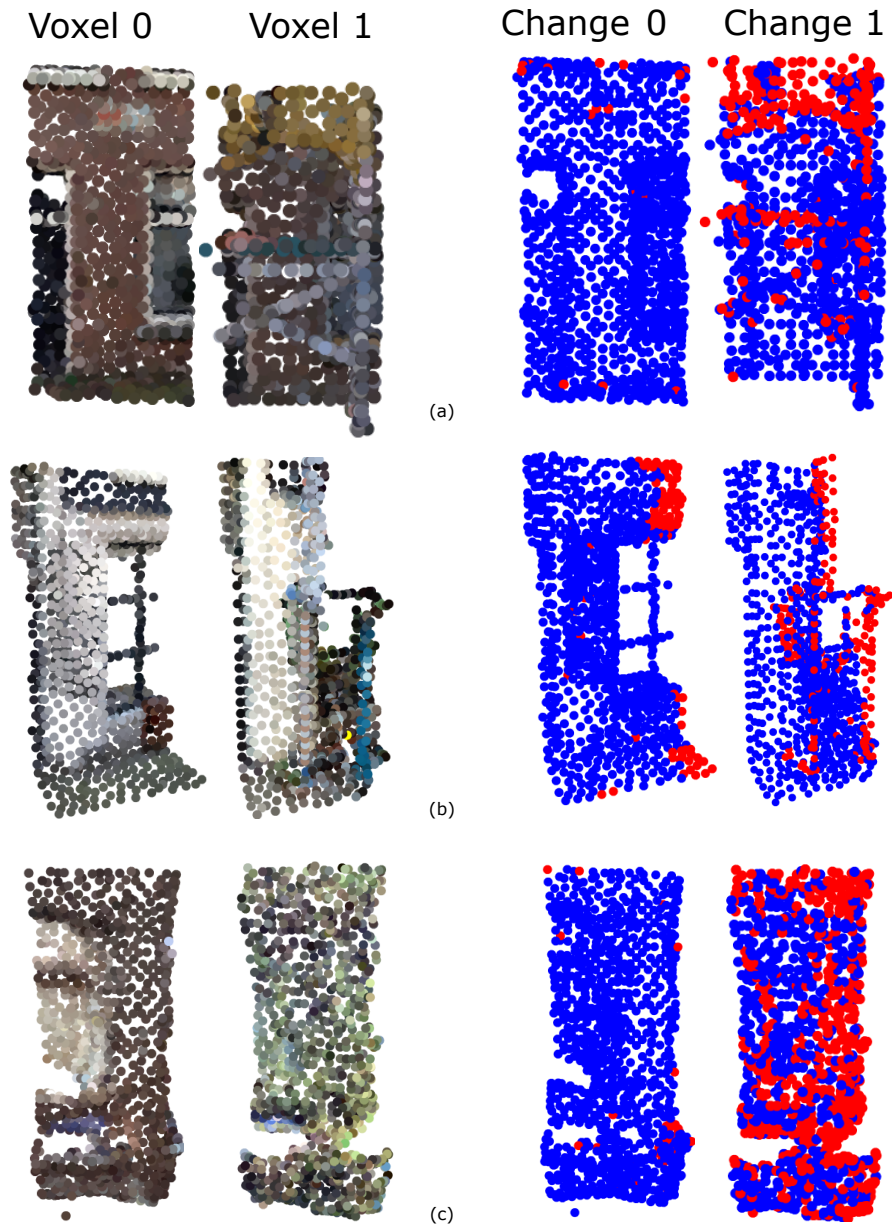


Figure 12: Change detection examples. Example a depicts a building facade where scaffolding is added in the second view. Example b depicts a partial facade where a flag and a pole shaped object appears in the second view. Part of the building visible in the first case is occluded in the second. Example c shows a partial window that is later covered by plant growth.

Switching to figure 12, the first example depicts newly added scaffolding to the front of a building. The model correctly classified most of the scaffolding as change whilst assigning no change to the unchanged facade. Some lower parts of the scaffolding are not detected and this is likely the same issue observed earlier where spurious den-

sity is assigned to the centre of the voxel to such colours. Example b shows two added pole-like objects in front of a building, both detected as change. In this case, even the parts close to the ground are detected, likely due to them being of a different colour. Further assigned change is likely due to severe occlusion of parts of the building in the second scan. Lastly, example c shows a partial window which is later covered in vine growth. The vine growth is classified fairly accurately as change whilst the underlying wall is not.

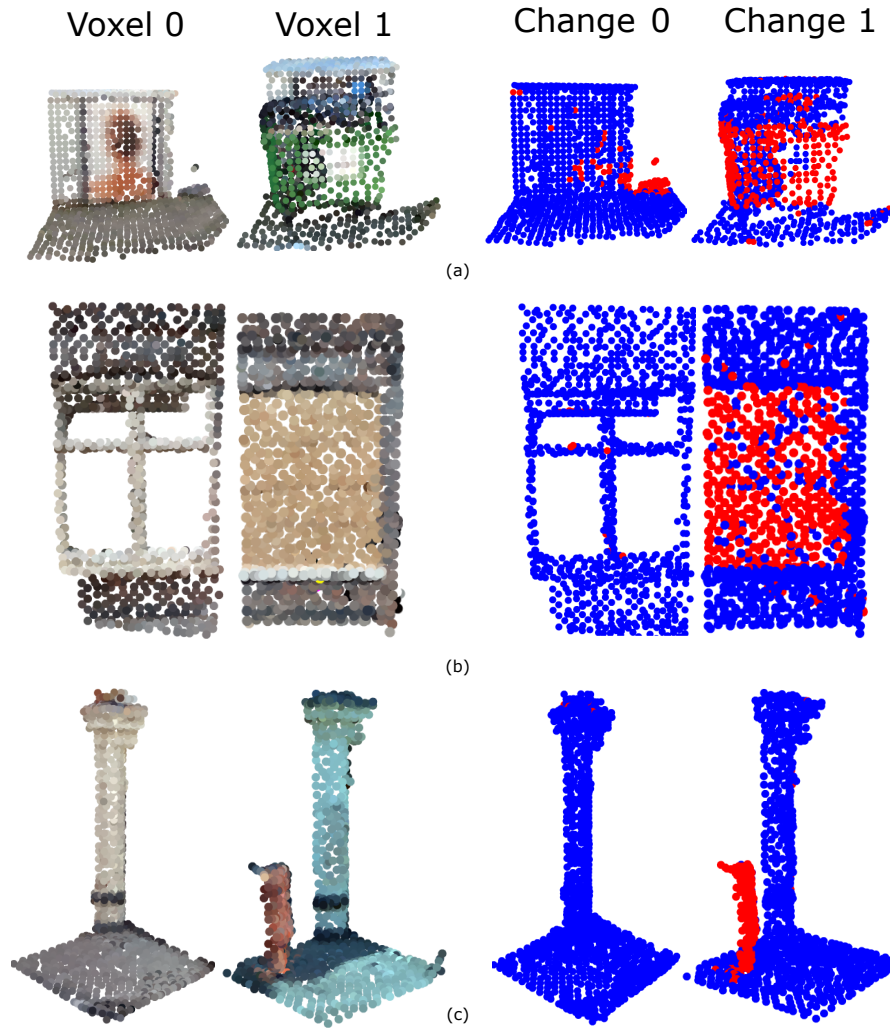


Figure 13: Change detection examples. Example a depicts a poster that is subsequently changed and occluded by a newly placed garbage bin. Example b shows a window that is subsequently boarded up. Example c shows significant colour change and the partial appearance of a red object.

Example a of figure 13 shows a fairly complex scene where a poster is changed and partially occluded by a garbage bin. At the current threshold of 5, the green part of the bin is detected but not the advert behind it. Trying a higher threshold value of 7.5n as seen in figure 14, this is mostly corrected although at the cost of slightly more spuri-

ous change classifications. The second example shows a window that is subsequently boarded up, the model is able to detect it accurately. Lastly, example c depicts a case of significant lighting change and shadow which are both ignored and an added red object being classified correctly. Again in this case the whole object is classified without issue likely due to being a different colour than that of the spurious density problem observed earlier.

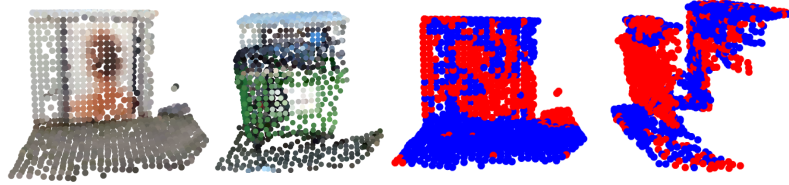


Figure 14: Change of example a figure 13 for looser threshold 7.5

Overall the model shows promise, is able to achieve the set goals to some extent. Problems observed in the examples are likely due to problems in the training regime such as the voxel point count filtering and the lack of enough, sufficiently variable data to allow robustness to be learned. These and other avenues for improvements are outlined in the following section.

DISCUSSION

6.1 REGARDING EVALUATION METHOD

The Change3D benchmark [33] was created as an evaluation set for Flow Compare it was later decided that it was not fit for this purpose. The first key reason for this decision was the mismatch between the annotations provided by the dataset and the output of Flow Compare. Specifically, the dataset provided object-wise labels (one label per object) which are fitting for most supervised methods but problematic given point-wise change values. A mapping from the point-wise change values to the object wise labels of the dataset is non-trivial, would likely require supervised training and would significantly obscure evaluation. The other concern was the mismatch in the distribution caused by the location of the point clouds in the Change3D benchmark and that of the available training set, the benchmark itself not having enough point clouds to train the present model.

The specific evaluation route taken was chosen as an alternative due to the absence of any other labelled datasets. The quantitative analysis with respect to the log-likelihood on the test set is the standard approach in likelihood-based generative modelling and is an effective way of comparing different architecture variations. While the qualitative assessment is not an objective assessment it showcases how the model performs in varied change detection cases and highlights problematic aspects. Future work on this task would benefit from the creation of a point-wise labelled change detection benchmark for evaluation purposes.

6.2 UNWANTED BIAS THROUGH TRAINING REGIME

As pointed out in the results section the proposed method can perform problematically in certain cases such as due to the erroneous density assigned to the centre of the voxel and when certain significant lighting variations are present. The former problem is suspected to be due to the training regime (minimum voxel point count filtering) and the latter is due to such variations not being present in the training set. Thus, it is clear that much care must be taken to not implicitly introduce unwanted bias through the chosen training regime and that the training set contains sufficient examples of the conditions that the model is expected to be robust to at test time.

6.3 ATTENTION MAPS

The key reason for using the cross attention approach to conditioning is to allow the main flow to dynamically focus on different parts of the context and thus avoiding the bottleneck of a global embedding. One intuitive way that such a model may have attended to the context could have been going from general focus to more targeted (around the point being processed) in the deeper layers, or vice versa. From visual inspection of attention maps, as depicted by figure 15, it was found that this was not generally the case and that no such easily interpretable pattern was present. The attention maps do not seem to focus around the point being processed, it is conjectured that different flow blocks specialize on semantically similar structures or specific colours but this requires further investigation.

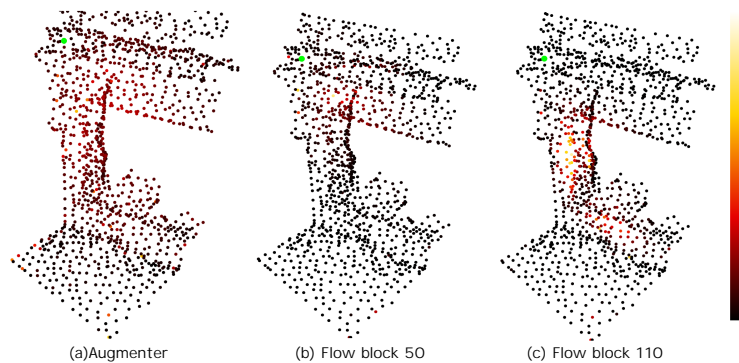


Figure 15: Attention maps of modules of progressively later conditional modules, with the lighter colour corresponding to more attention. The exaggerated green point is the location of the point being processed.

6.4 GENERATION EXAMPLES

Although the goal is not to generate point clouds, given the dual nature of Normalizing Flows one can generate point clouds by sampling from the base distribution and then passing the points through the inverse conditioned on a context cloud with such examples depicted by figures 16, 17. This is useful as it offers a way of visualizing the learnt distributions, seeing to some extent what the model expects. This may be informative and allow deficiencies to be identified or explained but must be done with caution as the interpretation of the results can be misleading. This is chiefly the case due to the fact that while the geometric component of the distribution is clearly visualized the colour component is not made clear to the same extent by the generation.

Each shown generated voxel consists of 4000 points that are sampled from a Gaussian with a standard deviation of 0.6 rather than 1 which was used for training so as to obtain tidier visualizations. The

ideal output is not necessarily an exact reproduction of the condition as the generation roughly represents the average expected scene which may differ significantly. By inspecting the generations it seems that the model in most cases expects a slightly fuzzier version of the given geometry, filling in small gaps (present due to coarse downsampling and irregular capture) by interpolating between adjacent points. The colours generally match the context but are often of slightly different hues, usually in the direction of more common variations of the given colour. Taking for example f of figure 17 where a darker ground is expected, likely due to such light ground being relatively uncommon in the training dataset as suspected due to the erroneous change detection results found for similar cases.

6.5 CONTINUOUSLY INDEXED NORMALISING FLOWS

Continuously indexed Flows were implemented as proposed by the authors, with the f mapping being an affine coupling block and distribution U a conditional Gaussian with an attention block. Due to the changes in architecture and the extra mapping a smaller augment dimension had to be used due to memory constraints. Limited experimentation found that the training was very unstable, comparable results to the other approaches were not able to be obtained thus results are omitted included in the experiments. Further experimentation and hyperparameter tuning are needed with this method as the main use case, modelling complex topologies are clearly present in the problem at hand where complex coloured geometry is modelled often containing disconnected objects.

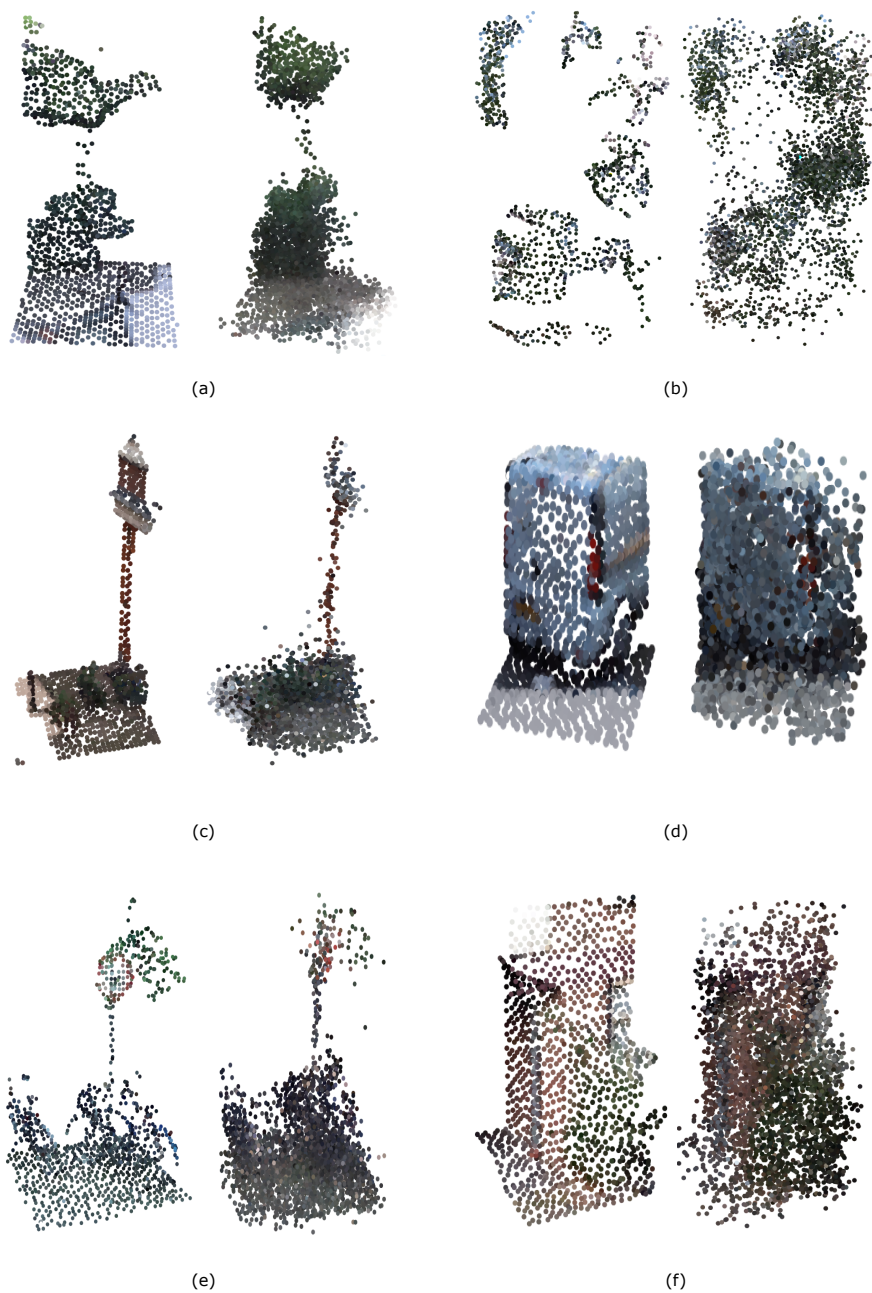


Figure 16: Examples of condition-generation pairs with the condition on the left and generation on the right respectively.

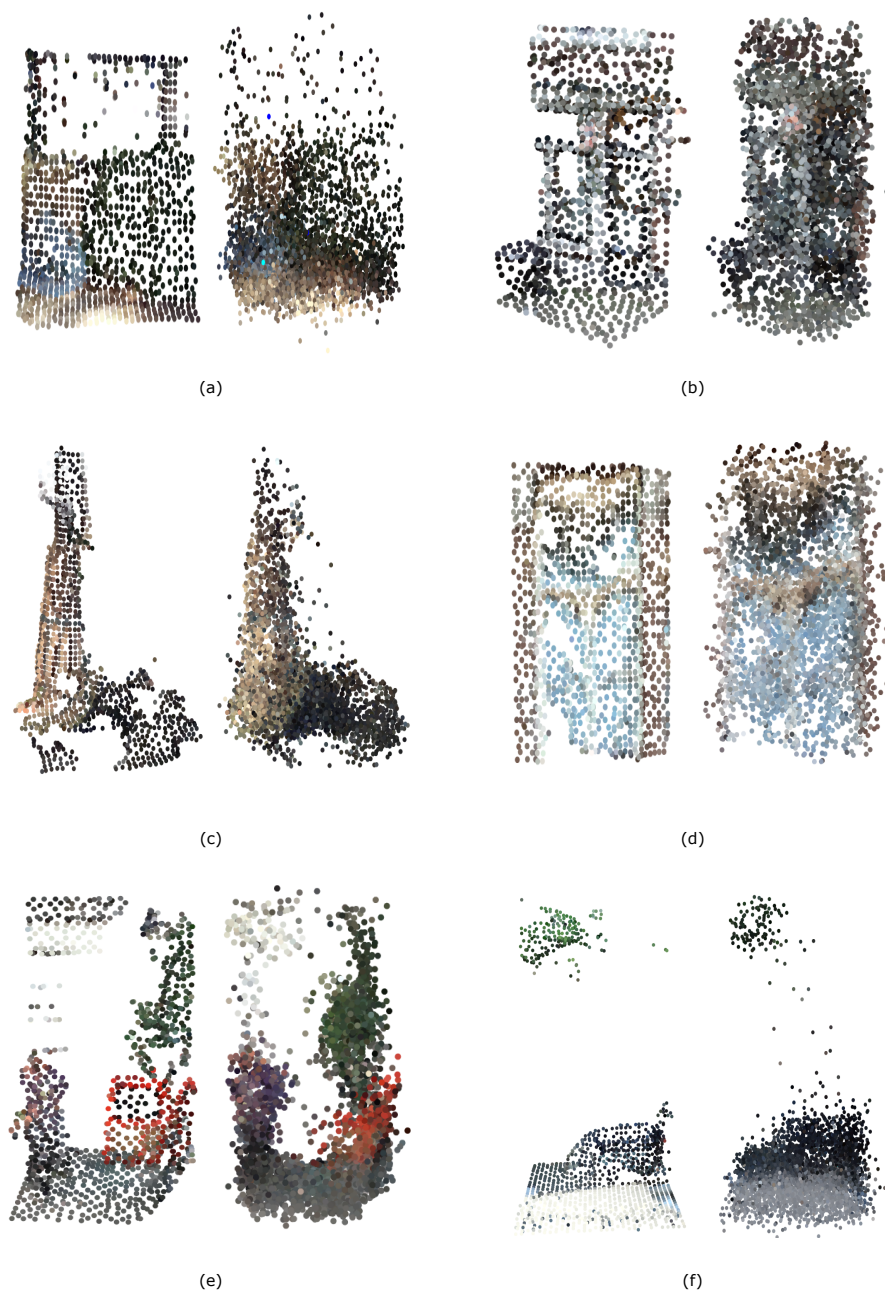


Figure 17: Examples of condition-generation pairs with the condition on the left and generation on the right respectively.

CONCLUSION & FUTURE WORK

7.1 CONCLUSION

In this thesis, the problem of robust change detection in complex street scenes through coloured point clouds is addressed. We present the first approach to this important task that leverages the latest advancements in deep learning. Significant challenges associated with the creation of comprehensive labelled datasets such as costly annotation and extreme class imbalance are sidestepped through a fully unsupervised approach. Additionally, this novel approach has the potential to detect geometric and colour change while being robust to semantically unimportant changes.

The method works by first using a Normalizing Flow model to learn the conditional distribution of points from unlabeled multi-temporal samples. Once this distribution is learned the problem of change detection can be reframed as one of anomaly detection under the learned distribution. Essentially, the model learns what to expect on average given one sample and then significant deviations i.e. anomalies are classified as changed. In order for this approach to be effective two complementary assumptions are needed. First that changes of interest are rare and second that common changes are not of interest. Furthermore, we detailed a voxel-based training regime and pre-processing pipeline which allows the model to learn the target distribution.

In order for the approach to be realized we designed a model architecture capable of modelling the target conditional density. The two key challenges here are maintaining sufficient flexibility of the main flow mapping and designing an effective conditioning mechanism.

Flexibility is addressed by pairing affine coupling blocks with an Augmenter module which embeds the initially low dimensional points into a high dimensional space.

The standard conditioning method was shown to be ineffective for the problem at hand due to not being conducive to capturing local information which is key for point-wise change detection. As an alternative a novel attention-based conditioning method is proposed which allows the model to dynamically query point-wise embeddings of the context. This allows not only each point but also each conditional module at different layers to flexibly extract information and was shown to outperform the standard approach significantly.

The proposed model is quantitatively evaluated with respect to the average likelihood on the test but also qualitatively through visual in-

spection of varied examples. It was demonstrated that the approach is able to effectively detect geometric change, colour change while being robust to semantically unimportant change, the last two of which have not been achieved by previous methods.

Lastly, although the approach is here applied to point cloud data straightforward adaptations could be made for change detection in different modalities such as the image domain or even combining modalities for increased performance.

7.2 FUTURE WORK

Due to the novelty of the proposed method and the accompanying model architecture, there is much room for further improvements. The following outlines how identified deficiencies may be addressed as well as promising avenues for further improvement.

LARGE SCALE TRAINING: Due to the training paradigm, this model benefits from large scale training as the training set needs to contain sufficient examples of all semantically uninteresting changes so that the model can learn to expect such cases (not flag them as anomalies). For example, if the training set does not contain fallen leaves present in autumn it would likely flag such cases as anomalies. In turn, the model must have the expressive capacity to model such diverse relations. Thus, the method itself necessitates large scale training which was not possible through the current project and highlights the need for improved efficiency to manage computational costs. An ideal training procedure may mirror in certain ways the training of large language models where large, varied unlabeled text corpuses are used. The equivalent being point clouds covering significant geographic areas of interest over varying conditions such as seasonal changes.

AUGMENTATION AND SIMULATION: In absence of sufficient multi-temporal data, augmentation and simulation methods can be used to synthetically generate new variations of the existing data. For example, modern game engines are capable of simulating realistic, full day and night cycles which would allow any lighting variations of the available point clouds to be generated. Training could then be done using both the original and generated data.

DATA FILTERING: In the current project minimal filtering was done with regards to the voxel pairs used for training, only with regards to the number of points present. Given that the voxel grid is placed uniformly over large geographic areas it is natural that certain classes or scenes dominate the dataset. For example, building facades may be overabundant while benches are relatively rare. Performing filtering

in advance, possibly with the aid of point cloud segmentation and classification methods may alleviate this issue, lead to faster training and improved overall performance. Some form of importance sampling may also prove useful but runs the danger of focusing on actual changes present in the data which would be detrimental.

IMPROVING COLOUR ACCURACY: The point cloud data used for training while generally geometrically accurate does contain colour inaccuracies which are artefacts of the image colour to coordinate the matching process used in their construction. The consistent presence of these artefacts seems to lead the model to expect large variation in colour which hinders its ability to detect actual semantically important colour change. Using data with improved colour accuracy or mitigating such issues in pre-processing will improve colour wise change detection.

WEIGHT SHARING: One way of decreasing the parameter count of the model would be to share parameters across different flow blocks, especially those that parametrize the attention modules. Although not explored it seems likely that some level of parameter sharing could be employed without significantly degrading performance. For example, weights could be shared between a certain number of consecutive flow blocks with the intuition that consecutive flow blocks may be summarizing the context in similar ways.

SPECIALIZED NORMALIZING FLOW ARCHITECTURE: There are few works modelling point clouds with Normalizing Flows and the core architectures used (usually affine coupling blocks) are used essentially as-is. Future work may improve such modelling by introducing architectures specialized to point clouds by for example leveraging permutation invariance and further tackling the problem of low dimensionality. Such advancements would of course directly improve the performance of this approach.

SEMANTIC INFORMATION: For more efficient training semantic information could be included for each point. This could for example be achieved by including the class of each point as given by a pre-trained semantic segmentation network or even the point-wise embeddings of such a network. This should significantly speed up the learning of the network as it does not need to learn to recognize classes from scratch.

HYPERPARAMETER OPTIMIZATION: Due to the novelty of the approach, there are limited existing works to base hyperparameter choices on, thus most hyperparameters were chosen heuristically with limited experimentation and optimization due to prohibitive computa-

tional costs. Further optimization of hyperparameters is key to improved efficiency and performance.

BIBLIOGRAPHY

- [1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. "Learning representations and generative models for 3d point clouds." In: *International conference on machine learning*. PMLR. 2018, pp. 40–49.
- [2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. "Bottom-up and top-down attention for image captioning and visual question answering." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 6077–6086.
- [3] K Somani Arun, Thomas S Huang, and Steven D Blostein. "Least-squares fitting of two 3-D point sets." In: *IEEE Transactions on pattern analysis and machine intelligence* 5 (1987), pp. 698–700.
- [4] Anju Asokan and J Anitha. "Change detection techniques for remote sensing applications: a survey." In: *Earth Science Informatics* 12.2 (2019), pp. 143–160.
- [5] Mohammad Awrangjeb, Clive S Fraser, and Guojun Lu. "Building change detection from LiDAR point cloud data based on connected component analysis." In: *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences* 2 (2015), p. 393.
- [6] Yifang Ban and Osama Yousif. "Change detection techniques: a review." In: *Multitemporal Remote Sensing*. Springer, 2016, pp. 19–43.
- [7] Changyou Chen, Chunyuan Li, Liqun Chen, Wenlin Wang, Yunchen Pu, and Lawrence Carin Duke. "Continuous-time flows for efficient inference and density estimation." In: *International Conference on Machine Learning*. PMLR. 2018, pp. 824–833.
- [8] JonckheereI CoppinP et al. "DigitalChangeDetection Method-sinEcosystem Monito ring: a Review." In: *InternationalJournalof Remote Sensing* 25.9 (2004), p. 1565.
- [9] Rob Cornish, Anthony Caterini, George Deligiannidis, and Arnaud Doucet. "Localised generative flows." In: (2019).
- [10] Rob Cornish, Anthony Caterini, George Deligiannidis, and Arnaud Doucet. "Relaxing bijectivity constraints with continuously indexed normalising flows." In: *International Conference on Machine Learning*. PMLR. 2020, pp. 2133–2143.
- [11] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using real nvp." In: *arXiv preprint arXiv:1605.08803* (2016).

- [12] Yueqi Duan, Yu Zheng, Jiwen Lu, Jie Zhou, and Qi Tian. "Structural relational reasoning of point clouds." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 949–958.
- [13] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. "Neural spline flows." In: *Advances in Neural Information Processing Systems* 32 (2019), pp. 7511–7522.
- [14] Arabi Mohammed El Amin, Qingjie Liu, and Yunhong Wang. "Convolutional neural network features based change detection in satellite images." In: 10011 (2016), 100110W.
- [15] Martin Engelcke, Dushyant Rao, Dominic Zeng Wang, Chi Hay Tong, and Ingmar Posner. "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks." In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 1355–1361.
- [16] Haoqiang Fan, Hao Su, and Leonidas J Guibas. "A point set generation network for 3d object reconstruction from a single image." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 605–613.
- [17] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. "Gvcnn: Group-view convolutional neural networks for 3d shape recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 264–272.
- [18] Sajid Ghuffar, Balázs Székely, Andreas Roncat, and Norbert Pfeifer. "Landslide displacement monitoring using 3D range flow on airborne and terrestrial LiDAR data." In: *Remote Sensing* 5.6 (2013), pp. 2720–2745.
- [19] Daniel Girardeau-Montaut, Michel Roux, Raphaël Marc, and Guillaume Thibault. "Change detection on points cloud data acquired with a ground laser scanner." In: *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 36.part 3 (2005), W19.
- [20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative Adversarial Networks." In: *Commun. ACM* 63.11 (Oct. 2020), 139–144. ISSN: 0001-0782. DOI: [10.1145/3422622](https://doi.org/10.1145/3422622). URL: <https://doi.org/10.1145/3422622>.
- [21] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. "A papier-mâché approach to learning 3d surface generation." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 216–224.

- [22] Daniel Hölbling, Barbara Friedl, and Clemens Eisank. “An object-based approach for semi-automated landslide change detection and attribution of changes to landslide classes in northern Taiwan.” In: *Earth Science Informatics* 8.2 (2015), pp. 327–335.
- [23] Chin-Wei Huang, Laurent Dinh, and Aaron Courville. “Augmented normalizing flows: Bridging the gap between generative flows and latent variable models.” In: *arXiv preprint arXiv:2002.07101* (2020).
- [24] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift.” In: *International conference on machine learning*. PMLR, 2015, pp. 448–456.
- [25] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. “Perceiver: General Perception with Iterative Attention.” In: *arXiv preprint arXiv:2103.03206* (2021).
- [26] Robert E Kennedy, Philip A Townsend, John E Gross, Warren B Cohen, Paul Bolstad, YQ Wang, and Phyllis Adams. “Remote sensing change detection tools for natural resource managers: Understanding concepts and tradeoffs in the design of landscape monitoring projects.” In: *Remote sensing of environment* 113.7 (2009), pp. 1382–1396.
- [27] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization.” In: *CoRR* abs/1412.6980 (2015).
- [28] Diederik P. Kingma and Prafulla Dhariwal. “Glow: Generative Flow with Invertible 1x1 Convolutions.” In: *Advances in Neural Information Processing Systems*. Vol. 31. 2018, pp. 10215–10224.
- [29] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes.” In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2014. URL: <http://arxiv.org/abs/1312.6114>.
- [30] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. “Improved variational inference with inverse autoregressive flow.” In: *Advances in neural information processing systems* 29 (2016), pp. 4743–4751.
- [31] Roman Klokov, Edmond Boyer, and Jakob Verbeek. “Discrete point flow networks for efficient point cloud generation.” In: *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII* 16. Springer, 2020, pp. 694–710.

- [32] Ryan A Kromer, Antonio Abellán, D Jean Hutchinson, Matt Lato, Tom Edwards, and Michel Jaboyedoff. "A 4D filtering and calibration technique for small-scale point cloud change detection with a terrestrial laser scanner." In: *Remote Sensing* 7.10 (2015), pp. 13029–13052.
- [33] Tao Ku, Sam Galanakis, Bas Boom, Remco C. Veltkamp, Darshan Bangera, Shankar Gangisetty, Nikolaos Stagakis, Gerassimos Arvanitis, and Konstantinos Moustakas. "SHREC 2021: 3D point cloud change detection for street scenes." In: *Computers Graphics* 99 (2021), pp. 192–200. ISSN: 0097-8493. DOI: <https://doi.org/10.1016/j.cag.2021.07.004>. URL: <https://www.sciencedirect.com/science/article/pii/S0097849321001369>.
- [34] Dimitri Lague, Nicolas Brodu, and Jérôme Leroux. "Accurate 3D comparison of complex topography with terrestrial laser scanner: Application to the Rangitikei canyon (NZ)." In: *ISPRS journal of photogrammetry and remote sensing* 82 (2013), pp. 10–26.
- [35] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. "Stacked cross attention for image-text matching." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 201–216.
- [36] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization." In: *ArXiv e-prints* (2016), arXiv-1607.
- [37] Yinjie Lei, Duo Peng, Pingping Zhang, Qihong Ke, and Haifeng Li. "Hierarchical Paired Channel Fusion Network for Street Scene Change Detection." In: *IEEE Transactions on Image Processing* 30 (2020), pp. 55–67.
- [38] Xuelong Li, Zhenghang Yuan, and Qi Wang. "Unsupervised deep noise modeling for hyperspectral image change detection." In: *Remote Sensing* 11.3 (2019), p. 258.
- [39] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. "Pointcnn: Convolution on x-transformed points." In: *Advances in neural information processing systems* 31 (2018), pp. 820–830.
- [40] Dengsheng Lu, Guiying Li, and Emilio Moran. "Current situation and needs of change detection techniques." In: *International Journal of Image and Data Fusion* 5.1 (2014), pp. 13–38.
- [41] Dengsheng Lu, Paul Mausel, Eduardo Brondizio, and Emilio Moran. "Change detection techniques." In: *International journal of remote sensing* 25.12 (2004), pp. 2365–2401.
- [42] Daniel Maturana and Sebastian Scherer. "Voxnet: A 3d convolutional neural network for real-time object recognition." In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 922–928.

- [43] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. "Occupancy networks: Learning 3d reconstruction in function space." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4460–4470.
- [44] Carsten Moenning and Neil A Dodgson. *Fast marching farthest point sampling*. Tech. rep. University of Cambridge, Computer Laboratory, 2003.
- [45] Satyam Mohla, Shivam Pande, Biplab Banerjee, and Subhasis Chaudhuri. "Fusatnet: Dual attention based spectrospatial multimodal fusion network for hyperspectral and lidar classification." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 92–93.
- [46] Charlie Nash and Christopher KI Williams. "The shape variational autoencoder: A deep generative model of part-segmented 3D objects." In: *Computer Graphics Forum*. Vol. 36. 5. Wiley Online Library. 2017, pp. 1–12.
- [47] Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. "Survae flows: Surjections to bridge the gap between vaes and flows." In: *Advances in Neural Information Processing Systems* 33 (2020).
- [48] Gianpaolo Palma, Paolo Cignoni, Tamy Boubekeur, and Roberto Scopigno. "Detection of geometric temporal changes in point clouds." In: *Computer Graphics Forum*. Vol. 35. 6. Wiley Online Library. 2016, pp. 33–45.
- [49] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. "Normalizing flows for probabilistic modeling and inference." In: *Journal of Machine Learning Research* 22.57 (2021), pp. 1–64.
- [50] Daifeng Peng, Yongjun Zhang, and Haiyan Guan. "End-to-end change detection for high resolution satellite images using improved unet++." In: *Remote Sensing* 11.11 (2019), p. 1382.
- [51] Albert Pumarola, Stefan Popov, Francesc Moreno-Noguer, and Vittorio Ferrari. "C-flow: Conditional generative flow models for images and 3d point clouds." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 7949–7958.
- [52] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. "Pointnet: Deep learning on point sets for 3d classification and segmentation." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

- [53] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. “Deep Hierarchical Feature Learning on Point Sets in a Metric Space.” In: *Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS’17*. Long Beach, California, USA: Curran Associates Inc., 2017, 5105–5114. ISBN: 9781510860964.
- [54] Rongjun Qin and Armin Gruen. “3D change detection at street level using mobile laser scanning point clouds and terrestrial images.” In: *ISPRS Journal of Photogrammetry and Remote Sensing* 90 (2014), pp. 23–35.
- [55] Danilo Rezende and Shakir Mohamed. “Variational inference with normalizing flows.” In: *International Conference on Machine Learning*. PMLR, 2015, pp. 1530–1538.
- [56] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. “Octnet: Learning deep 3d representations at high resolutions.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 3577–3586.
- [57] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. “3d point cloud generative adversarial network based on tree structured graph convolutions.” In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 3859–3868.
- [58] Martin Simonovsky and Nikos Komodakis. “Dynamic edge-conditioned filters in convolutional neural networks on graphs.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 3693–3702.
- [59] Michał Stypułkowski, Kacper Kania, Maciej Zamorski, Maciej Zieba, Tomasz Trzeciński, and Jan Chorowski. “Representing point clouds with generative conditional invertible flow networks.” In: *Pattern Recognition Letters* (2021).
- [60] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. “SPLATNet: Sparse Lattice Networks for Point Cloud Processing.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.
- [61] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. “Multi-view convolutional neural networks for 3d shape recognition.” In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 945–953.
- [62] Aparna Taneja, Luca Ballan, and Marc Pollefeys. “Geometric change detection in urban environments using images.” In: *IEEE transactions on pattern analysis and machine intelligence* 37.11 (2015), pp. 2193–2206.

- [63] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. "Tangent convolutions for dense prediction in 3d." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3887–3896.
- [64] J Trinder and M Salah. "Aerial images and LiDAR data fusion for disaster change detection." In: *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci* 1 (2012), pp. 227–232.
- [65] M Tsakiri and V Anagnostopoulos. "Change detection in terrestrial laser scanner data via point cloud correspondence." In: *IJEIR* 4.3 (2015), pp. 476–486.
- [66] Ali Osman Ulusoy and Joseph L. Mundy. "Image-Based 4-d Reconstruction Using 3-d Change Detection." In: *Computer Vision – ECCV 2014*. Ed. by David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars. Cham: Springer International Publishing, 2014, pp. 31–45. ISBN: 978-3-319-10578-9.
- [67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.
- [68] Jakob Verbeek. "Discrete Point Flow Networks for Efficient Point Cloud Generation." In: ().
- [69] Chu Wang, Babak Samari, and Kaleem Siddiqi. "Local spectral graph convolution for point set feature learning." In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 52–66.
- [70] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. "O-cnn: Octree-based convolutional neural networks for 3d shape analysis." In: *ACM Transactions on Graphics (TOG)* 36.4 (2017), pp. 1–11.
- [71] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. "Dynamic Graph CNN for Learning on Point Clouds." In: *CoRR abs/1801.07829* (2018). arXiv: [1801.07829](https://arxiv.org/abs/1801.07829). URL: <http://arxiv.org/abs/1801.07829>.
- [72] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. "Dynamic graph cnn for learning on point clouds." In: *Acm Transactions On Graphics (tog)* 38.5 (2019), pp. 1–12.
- [73] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling." In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2016, pp. 82–90.

- [74] Changyi Xiao and Ligang Liu. "Generative Flows with Matrix Exponential." In: *International Conference on Machine Learning*. PMLR. 2020, pp. 10452–10461.
- [75] Wen Xiao, Bruno Vallet, Mathieu Brédif, and Nicolas Paparoditis. "Street environment change detection from mobile laser scanning point clouds." In: *ISPRS Journal of Photogrammetry and Remote Sensing* 107 (2015), pp. 38–49.
- [76] Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. "PAConv: Position Adaptive Convolution with Dynamic Kernel Assembling on Point Clouds." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 3173–3182.
- [77] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. "Pointflow: 3d point cloud generation with continuous normalizing flows." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 4541–4550.
- [78] Ze Yang and Liwei Wang. "Learning relationships for multi-view 3D object recognition." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 7505–7514.
- [79] Li Yi, Hao Su, Xingwen Guo, and Leonidas J Guibas. "Syncspec-cnn: Synchronized spectral cnn for 3d shape segmentation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2282–2290.
- [80] Tan Yu, Jingjing Meng, and Junsong Yuan. "Multi-view harmonized bilinear network for 3d object recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 186–194.
- [81] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. "Pointweb: Enhancing local neighborhood features for point cloud processing." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5565–5573.
- [82] K. L. de Jong and A. Sergeevna Bosman. "Unsupervised Change Detection in Satellite Images Using Convolutional Neural Networks." In: (2019), pp. 1–8. DOI: [10.1109/IJCNN.2019.8851762](https://doi.org/10.1109/IJCNN.2019.8851762).