PLGP: Point Cloud Inpainting with Patch-based Local Geometric Propagating

Yan Huang, Chuanchuan Yang, Yu Shi, Hao Chen, Weizhen Yan and Zhangyuan Chen

***Abstract*—The booming of LiDAR technologies has made the point cloud become a prevailing data format for 3D object representation. However, point cloud usually exhibits holes of data loss mainly due to occurrence of noise, occlusion or the surface material of the object, which is a serious problem affects the target expression of point clouds. Point cloud inpainting is the key solution for holes problem. In this paper, we propose a novel approach called PLGP to automatically fill the lost data obtained by three-dimensional scanning. Different from typical methods transform the data into range image to conduct the hole-detection or even complete the whole hole-filling then transform back to 3D point clouds, this work tends to detect the hole directly in 3D space and inpaint it by iteratively searching for the proper context and making it propagate appropriately along the occlusion's local geometric structure. Experimental results with comparisons demonstrate its competitive effectiveness with a F-score as high as 0.89 and a 23.45 dB average gain in GPSNR with consumed time reduced by up to 60%.**

***Keywords*—Point cloud, inpainting, local geometric, patch propagating.**

I. Introduction

RECENT advance in Light Detection and Ranging (LiDAR) technologies enables the convenient acquisition of real-world 3D point clouds by directly capturing the three-dimensional information about a specific environment. Lidar point clouds consist of a set of points in the 3D space, sometimes with attributes like intensity or color. They are used to depict the geometry of 3D objects or scenes, which have inspired various applications such as heritage reconstruction, augmented reality and navigation in self-driving. Lidar scanner collects point cloud data by deducing distances to objects' surface with the speed of light and the time laser pulses reflect off the surfaces, or use the phase difference of continuous-wave signal to acquire the points distance. However, incomplete scan views from the scanning equipment, low reflectance and high absorption of the scanned surface against the laser beam or even the inherent limitations of acquisition units, etc. [1] may result in defective point clouds that generate shadowlike missing regions on the object surface. Therefore, it is indispensable to handle the hole-inpainting during the data acquisition. Point cloud inpainting aims to infer or search for plausible voxel data from a partial point clouds to achieve the automatic inpainting of missing regions and make the inpainted object present a complete visual effect [1]-[3]. This study has drawn increasing attention in recent years, but it remains developing and challenging due to the difficulties of computational cost and high requirement on the inpainted consistency in point clouds. Existing methods for point cloud inpainting can fall into three main categories. 1) Inpainting by interpolating data from database. 2) Method based on triangular mesh using the shape information of the model. 3) Method based on scattered point clouds using the surrounding shape.

The methods of category 1) tend to search for a shape similar to the target object from the database and integrate them to fulfill the inpainting [4], [5]. These methods may suffer from a significant drawback since they need to model and store the similar shapes in advance in the database. The methods in the second category make the utmost of 3D mesh model to reconstruct the target objects. Base on the mesh, it is easy to locate the missing region and get its nearby geometry and topology information, then use this information to fill in the missing area [6], [7]. However, these methods overly rely on the quality of mesh construction and they are costly because they must firstly construct and obtain the three-dimensional mesh from the input point clouds.

The third category is aimed at fitting the scattered point clouds with occluded holes into a complete and continuous surface, which is more preferred than the aforementioned ones due to its pleasant cost. Many studies attempt to pursue a lower complexity while maintain the favorable inpainted effect. A typical work [3] dedicates to fill large holes in LiDAR data by inpainting depth gradients, which needs to transform the 3D point clouds into a range image, and use an image-based method to complete the inpainting then retransform the depth image back to 3D point clouds. Another typical case, Hu *et al.* [8] propose an inpainting method based on the local frequency interpretation and non-local self-similarity on graph. It projects the 3D point cloud into 2D range map to accomplish the hole detection and complete the inpainting in the cube-based 3D

This work was supported by National Key R&D Program of China under Grant 2019YFB1802904.(Corresponding author: C. Yang.)

Y. Huang, C. Yang, Z. Chen are with State Key Laboratory of Advanced Optical Communication Systems and Networks, Department of Electronics, Peking University, Beijing, 100871, China (e-mail: huang_yan@pku.edu.cn,yangchuanchuan@pku.edu.cn,chenzhy@pku.edu.cn).

Y. Shi is with Department of Electronics, Peking University, Beijing, 100871, China (e-mail: woodchloe@pku.edu.cn) .

H. Chen and W. Yan are with the SenseFuture Technologies Co., ltd, Beijing 100025, China (e-mail: chenhao@sensefuture.cn, yanweizhen@sensefuture.cn).

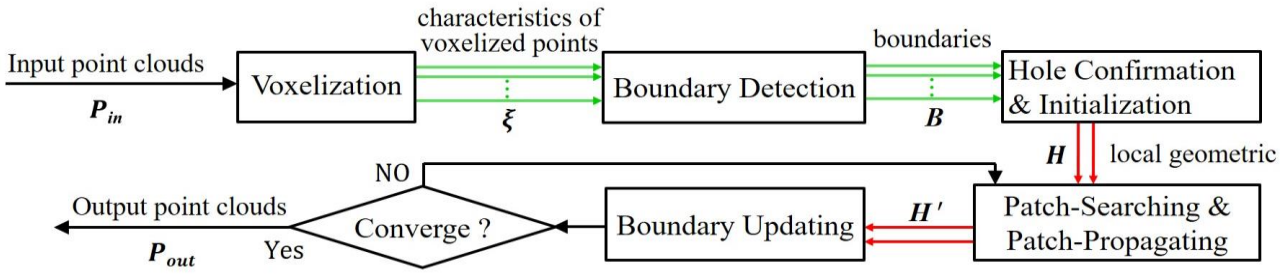


Fig. 1. Pipeline of PLGP for lidar point clouds inpainting, illustrating the complete process from point clouds input and processing to output.

space. Though the above methods make good achievements in point clouds inpainting, they switch back and forth between a 2D depth environment and 3D point cloud space not only increase the cost but may also contribute to ectopic points and even geometric loss. Instead, Fu *et al.* [9] handle the inpainting task by retrieving the most similar region for the occlusion based on the normal and non-local similarity in the point clouds. Lai *et al.* [10] estimate the rough geometry of the hole and copy the detailed structure from elsewhere to refine the initial estimation then select appropriate elastic warps to ensure the copied patch match the surrounding hole. Though these works pick the holes' surrounding data for point clouds inpainting with a low complexity, their results incline to be more complanate than the actual ground-truth due to the reference from local similar neighbor, and it may produce artifacts around the holes in complicated structure.

In this paper, a novel inpainting approach called patch-based local geometric propagating (PLGP) method is proposed for lidar point clouds, which aims to construct a complete and consistent 3D lidar data representation from incomplete surface scans with a coarse-to-fine strategy. The novelties of this work are listed as follows:

1) This paper voxelize the point clouds to directly detect and confirm the missing regions in 3D space rather than a 2D environment, which can avoid ectopic points and geometric loss in the inter-transformation of 2D and 3D.
2) We insist to detect all the boundaries before the hole confirmation to differentiate the holes that to be inpainted and a complete surface boundary so as to make the method accommodate various inputs.
3) Different from existing methods that fill the hole with the whole best-match content in one turn, this work retrieves suitable context in each iteration and propagates the proper retrieved content into the hole gradually along its surrounding geometry until the hole is full to break the loop. The strategy fully exploits the local information of the occlusion to achieve the inpainting, which helps reduce the occurrence of artifacts and improve the inpainted consistency.

The proposed PLGP provides efficient hole-inpainting based on local patch propagating in lidar point clouds, which allows the occlusion to be quickly and elaborately inpainted. Unlike typical methods handling the 3D point clouds into a 2D environment that takes a risk of core data corruption, PLGP completes the whole process in 3D space which maintains cost-effective and data consistency especially in the case of large amounts of point data. The proposed method makes good performance both in objective and subjective quality. Elaborate evaluation of comparative experiments indicates its inpainted results with a F-score as high as 0.89 and an average 23.45 dB gain in GPSNR with consumed time reduced by up to 60%.

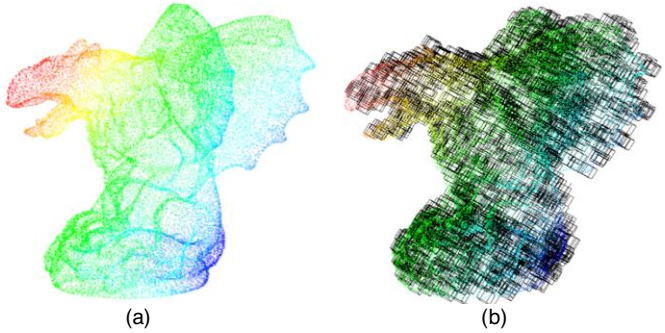
II. Patch-based Local Geometric Propagating (PLGP) Method For Lidar Point Clouds

The proposed PLGP is built upon a direct hole detection in the 3D point cloud space and a patch-based coarse-to-fine propagating and inpainting. Fig.1 illustrates the entire pipeline of our framework. It takes an incomplete point clouds $\boldsymbol{P_{in}}$ with arbitrary occlusion as input. Define $\boldsymbol{\xi}$ to be the set of location features for the voxelized points, let $\boldsymbol{B}$ denote the set of boundaries in $\boldsymbol{P_{in}}$, where $\boldsymbol{B} \subsetneq \boldsymbol{P_{in}}$, $\boldsymbol{H}$ denotes the set of occluded holes to be inpainted in $\boldsymbol{P_{in}}$, and $\boldsymbol{H} \subsetneq \boldsymbol{B}$. $\boldsymbol{H'}$ is the set of updated boundaries from $\boldsymbol{H}$. $\boldsymbol{P_{out}}$ is the completed inpainted output. The framework visually describes the whole process of processing the input $\boldsymbol{P_{in}}$ to get the output $\boldsymbol{P_{out}}$.

When $\boldsymbol{H}$ is extracted from $\boldsymbol{B}$, the occluded hole is initialized into multiple local blocks with grid voxels in advance. For every occluded block, we retrieve its suitable context, warps the retrieved voxels to conform with the local geometry around the block, then consistently blending and propagating the warped data into the block along its boundaries structure. Every time it completes a circle of propagation, $\boldsymbol{H}$ will be updated into $\boldsymbol{H'}$ immediately for a new iteration until the entire occlusion is filled to output the final point clouds. The aim of this strategy is to ensure the local details in the 3D structure of an occlusion can be delicately inpainted and it can deal with multiple discontinuous occluded holes in the point clouds simultaneously, which however, cannot be done in existing methods, since they incline to find out a whole patch for every occlusion. The presence of dense occlusions may increase their consumed time and decrease the inpainted quality.

A. Voxelization for Point Clouds

For the inpainting goal, the first step is to conduct the voxelization. As shown in Fig. 2, the step is designed to adjust



(a) (b)

Fig. 2. A demonstration of our voxelization for point clouds with Gargo. (a) The input lidar point clouds. (b) The voxelized result of the input data.

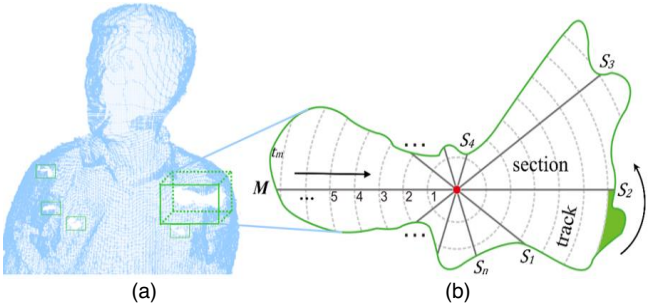


Fig. 3. A projective example of 3D irregular occlusion in point clouds. (a) Point cloud with occlusion. (b) Projective plane of the occlusion.

the scattered point clouds into regular grids to facilitate the delineation of point-based distribution, which benefits the occluded detection and the location depiction for that points to be filled. For the voxelization of PLGP, KNN-search [11] is applied to process the point set demarcation and handle local outlier factors. We manage the original global coordinates to fine-tune the distance among every two adjacent points close to 0.7 and translate the input point clouds into the voxelized coordinate. Base on the octree decomposition, a small cube is performed to be the root node, which is recursively subdivided into eight sub-voxels. Nonempty voxels will continue to be partitioned until either the residual threshold or the minimum pixel size criterion is reached. The residual threshold and eigenvector of the coordinated voxelized point clouds are computed and applied to extract their voxelized location information to provide the input data for boundary detection.

B. Boundary Detection & Hole Confirmation

Occluded boundary detection in 3D point clouds is a required preprocessing step directly affects the occluded hole inpainting thus brings changes to their further applications such as 3D reconstruction [12] and objects detection [13]. It is aimed at seeking out the set of consecutive points with boundary features in the established spatial-topological relation of scattered point clouds through a decision-making mechanism.

This work stick to complete the hole detection in 3D point clouds space rather than a rang image, which can reduce the transform cost for one turn. It is difficult for many methods to distinguish occluded and complete surface boundaries, such as the boundary of a sign board [14]. PLGP concentrates on the extraction of occluded borders from all the detected boundaries of point clouds to confirm the holes to be inpainted. If the input point cloud data exhibits occluded holes without complete surface boundaries, skip the step of boundary detection and go straight to that hole extraction and confirmation. This process aims to help the proposed PLGP deal with challenging uncertain missing regions of various input point clouds.

To achieve the boundary detection, we use grid-R-tree [15] to dynamically spatial index the whole topological relations of the voxelized point clouds, and reference to the local profile of k-nearest neighbor points based on sampling points to find out the boundary edges. If there is a sequence of points connected by an edge that belongs to a k-near point surface, it will be used as the seed edge to continue to look for its adjacent boundary edge until it finally finds out a closed ring formed by all the boundary edge, which is called a closed-loop boundary. To distinguish an occluded hole to be inpainted from all the closed-loop boundaries, fit the microtangent plane of boundaries in point clouds with least square and then project the plane. To construct the least squares surface, let $\boldsymbol{H} = \{C_i\}, i = 0,1, \dots, n$, where C_i denotes the boundary vertex of in the hole $\boldsymbol{H}$, $\pi(\Lambda, \boldsymbol{N_v})$ represents the least square plane, $\boldsymbol{N_v}$ is the plane normal and Λ is a point in the plane, which can be defined as:

$$\Lambda = \frac{1}{n}\sum_{i=0}^{n} C_i \,. \tag{1}$$

In order to get $\boldsymbol{N_v}$, we calculate the eigenvalue λ_i and its corresponding eigenvector $\boldsymbol{V_i}, i = 0,1, \dots, n$, of the covariance matrix $\boldsymbol{K}$ of each boundary vertex.

$$\boldsymbol{K} = \sum_{i=0}^{n} (C_i - \Lambda)\,(C_i - \Lambda)^T \,, \tag{2}$$

according to the principal element, $\boldsymbol{N_v}$ is the eigenvector corresponding to the minimum eigenvalue. The eigenvalue λ_i and its $\boldsymbol{V_i}$ of each boundary vertex are detected according to the maximum angle of the lines between the sampling points and its adjacent projection points. If the projected polygon on the least square plane doesn't cross itself, we think the boundary shape is a hole. To further confirm a hole from the cases of self-cross, based on the λ_i of every seed boundary (SB) point in a plane and its last echo information, SB points are grown into neighbor points with the occluded boundary growth function [14] to detect the hole's boundary points within a certain radius from the detection center. To confirm all the occluded holes for initialization, the weight center of gravity based on the occluded boundaries is extracted to estimate the hole's detection center, which will contribute to the follow-up patch-based propagating and inpainting.

C. Best Match Searching and Propagating

Occlusion filling consists of two core steps: best-match searching and proper propagation, applied iteratively until the occluded occlusion is filled. Define O_h to be the hole's detected center in the 3D space. Υ denotes the max-radius from O_h to the hole's boundaries. The voxelized 3D occlusions in the geometric grids are mapped to M tracks and N sections in the 3D space according to Υ with reference to the curvature of occluded boundary and gravity-center of occlusion. In order to understand the mapping strategy intuitively, a projective example of 3D irregular occlusion is provided, see Fig. 3. Define t_m and s_n to be the $m\text{-}th$ $(m = 1,2, \dots, M)$ track and the $n\text{-}th$ $(n = 1,2, \dots, N)$ sector respectively. $\Phi(m,n)$ is the block in the occlusion across t_m and s_n. $\delta\Phi(m,n)$ denotes the boundary of $\Phi(m,n)$ and let $\Omega(\Phi(m,n))$ represent the local geometric structure around $\Phi(m,n)$. From the outer track to the first track, the first sector to the last, search context-based patches for a best-match one for every block and adjust the patch to propagate into the missing block in the occlusion along its detailed structure. Algorithm 1 depicts the algorithm of our patch-based searching and local geometric propagating, which visually expresses the core idea of our inpainting strategy.

Best-match searching. Specially, a neighbor patch or a symmetric patch in the voxelized 3D space is normally more likely to be the "best-match" of the occluded hole. The search area located around the hole based on its geometric feature can help direct the patch-match so as to reduce search space and achieve good preservation of the 3D-spatial coherency. With

Algorithm 1 PLGP Algorithm for Point Clouds

Input: $\boldsymbol{P_{in}}$, $\boldsymbol{H}$, $\boldsymbol{H'}$, M, N, $\Phi(m,n)$, $\Omega(\Phi(m,n))$
Function: *InpaintingLidarData* ($\boldsymbol{P_{in}}$, $\boldsymbol{H}$)
getInputInpaintingSource ($\boldsymbol{P_{in}}$, $\boldsymbol{H}$)
VoxelizationGeometricGrid (ψ_{in}, M, N)
for (i=N; i>0; i--)
  for (j=1; j<=M; j++)
    if ($\Phi(m,n)$==*full*) **then**
      continue
    else if ($\Phi(m,n)$==*all null*) **then**
      3DPatchSearch($\psi_{in}, \Phi(m,n), \Omega(\Phi(m,n))$
      δ=*getBestMatchPatch* ($\psi_{in}, \Omega(\Phi(m,n))$
      PatchProperPropagating (δ, ψ_{in})
      η=*BoundaryUpdate* ($\boldsymbol{P_{in}}$, $\boldsymbol{H}$)
    else
      InpaintingLidarData ($\eta, \boldsymbol{H'}$)
    end if
  end for
end for
Output: $\boldsymbol{P_{out}}$

this verdict, many candidate patches can be eliminated during the search to speed up the process. Define $\boldsymbol{\mathcal{F}}$ to be the set of candidate patches, ℓ to be the target best-match patch for $\Phi(m,n)$. The $\Phi(m,n)$ with its surrounding voxels is denoted $\boldsymbol{\varpi}$. The search for the appropriate patch, which actually is a process of computing the best pairwise matches during the iterations in a specific search space. Leveraging on the local voxels from the boundaries of the occluded block, we match patches' similarity based on the target voxels with references to the location features of voxelized points in grids with the following weighted average:

$$u(p,q) = \frac{\sum_{q\in\boldsymbol{\mathcal{F}}} s_q^p u(p-\sigma q)(p-q)}{\sum_{p\in\boldsymbol{\varpi}} s_q^p}, \tag{3}$$

where the weight s_q^p is defined as:

$$s_q^p = \exp\left(-\frac{\varphi_q^p}{2\sigma^2 \xi_q \omega_q}\right) \xi_q \omega_q \varphi_q^p . \tag{4}$$

Here ξ_q denotes the distance from target voxel q to the nearest border of the current block, ω_q represents the offsets of target voxel q to the center of $\Phi(m,n)$, and φ_q^p is a confidence map defined by the flowing equation.

$$\varphi_q^p = \begin{cases} (1-t)\exp\left(-\frac{\xi_q}{\sigma^2}\right), & if\ q \notin \delta\Phi(m,n) \\ 1, & otherwise. \end{cases} \tag{5}$$

u, σ and t are the tuning parameters. The confidence map is used to demonstrate the voxels data from the boundary of the candidate patch in $\boldsymbol{\mathcal{F}}$ towards its center point and eliminate some border artifacts. To propel the search efficiency, when ℓ is located by iteratively calculating the structure similarity of the block and its candidate contents, ℓ is first translated so that its center point can coincide with that of $\Phi(m,n)$, the translated ℓ is denoted ℓ', the best 3D rotation matrix $\boldsymbol{R_t}$ is determined to align ℓ' with $\Phi(m,n)$, the rotated 3D patch is denoted $\boldsymbol{\mathcal{H}}$. Specifically, the rotation matrix is illustrated as:

$$\boldsymbol{R_t} = \arg\min_{\boldsymbol{R}} d(\boldsymbol{\mathcal{H}}, \boldsymbol{\varpi}), \tag{6}$$

where $\boldsymbol{R}$ is the rotation matrix, $d(\boldsymbol{\mathcal{H}}, \boldsymbol{\varpi})$ is the one-sided Hausdorff Distance from $\boldsymbol{\varpi}$ to $\boldsymbol{\mathcal{H}}$.

$$d(\boldsymbol{\mathcal{H}}, \boldsymbol{\varpi}) = \max_{A\in\boldsymbol{\varpi}} \min_{B\in\boldsymbol{\mathcal{H}}} ||A-B||_2, \tag{7}$$

where $||.||_2$ is the Euclidean norm. The rotated patch from ℓ, is adjusted to fill in the $\Phi(m,n)$ along its surrounding local geometry to obtain spatial continuous surface.

Proper propagation. Since the surfaces of point clouds are not always plane in 3D space, and the numbers of points around the boundary-point and non-boundary-point are different, for $\forall C_i \in \boldsymbol{H}$, $\boldsymbol{X} = \{X_0, X_1, \dots, X_{k-1}\}$ is defined to be the k-nearest-neighbors set of C_i. Based on the principal component analysis, a plane L is established through point C_i that fits all its k-nearest-neighbor points. A projective coordinate is constructed with C_i as the center point in the plane L, and the coordinate axis are $\boldsymbol{e_1}$, and $\boldsymbol{e_2}$ respectively, which are denoted to be the eigenvector corresponding to the maximum eigenvalue, the secondary eigenvalue of a scatter matrix $\boldsymbol{S_M}$, respectively. $\boldsymbol{S_M}$ here can be represented as:

$$\boldsymbol{S_M} = \sum_{i=0}^{k-1} (X_i - C_i)(X_i - C_i)^T . \tag{8}$$

Project the points of $\boldsymbol{X}$ into the coordinate, then obtain a set of projective points as $\boldsymbol{X'} = \{X_0', X_1', \dots, X_{k-1}'\}$, in which, the coordinate of the projective point $X_i'(x_i', y_i'), i = 0,1,\dots,k-1$, can be calculated from the following formulas:

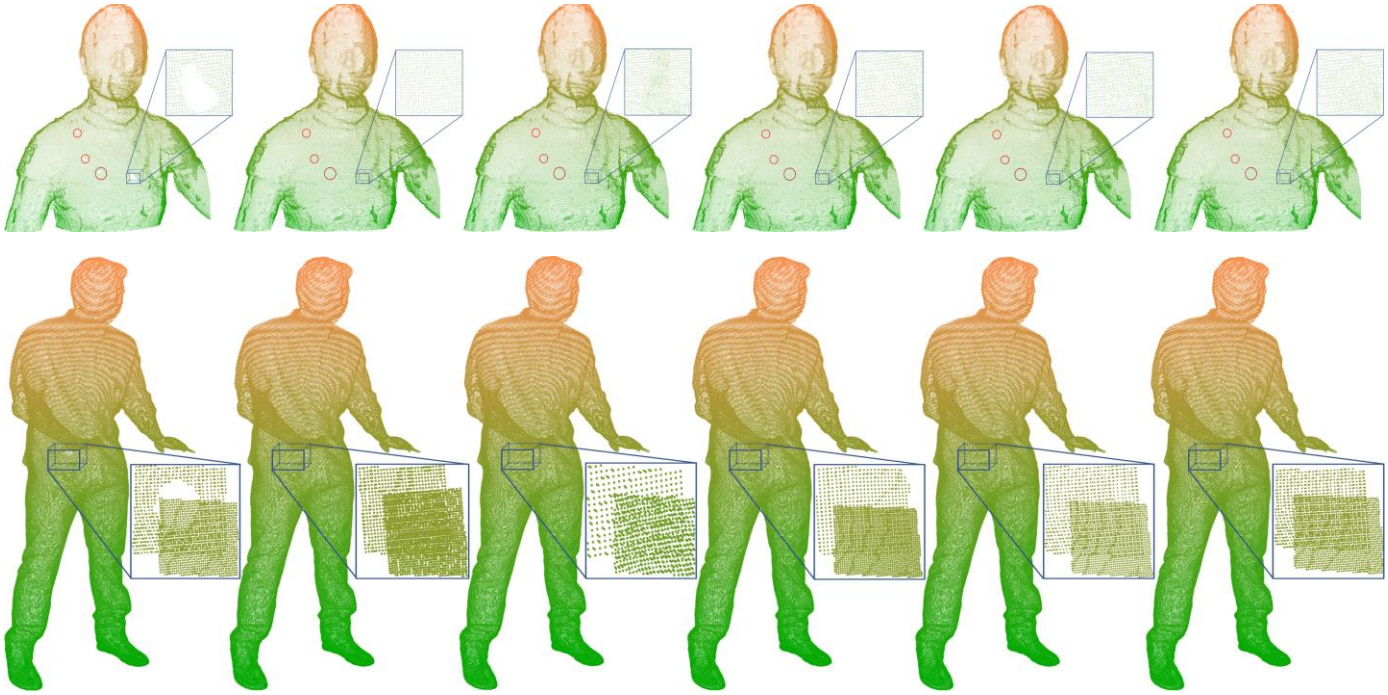
$$x_i' = \boldsymbol{e_1}^T (X_i - C_i), \tag{9}$$

$$y_i' = \boldsymbol{e_2}^T (X_i - C_i). \tag{10}$$

Then the angle between $C_i x_i'$ and $\boldsymbol{e_1}, i = 0,1,\dots,k-1$, is calculated in sequential order to get the angle set $\boldsymbol{\theta} = \{\theta_0, \theta_1, \dots, \theta_{k-1}\}, 0^\circ \le \theta_i \le 360^\circ$. All the fitted points in $\boldsymbol{X}$ are sorted according to the angle from the smallest to the largest, if $\forall\theta_i$, $|\theta_i - \theta_{i+1}| > \mu$,where μ is a preset threshold used to control the angle of the projective point in the plane and its corresponding 3D x-axis coordinate, warp the rotated patch along the consistent direction of its boundary points with the fitted surface to confirm with $\Omega(\Phi(m,n))$ and propagate the voxels of the patch into $\Phi(M,N)$. If $|\theta_i - \theta_{i+1}| \le \mu$, the current hole's detected center O_h will be updated to ensure the curvature of occluded boundary and gravity center from the hole center accord with the local geometric around $\boldsymbol{H}$, thus make it go to a new iteration of patch searching and propagating. With the loop operations, the point cloud data is propagated in the occluded blocks with the target patches that both contain the spatial structure and local geometric, thus it saves more cost in the filling work while ensuring the inpaintied effect.

D. Boundary Updating

When the patch propagation of the outer ring is completed, new boundary of the rest of the occluded hole will be updated as well as its detected center for the purpose of avoiding the discontinuous point clouds in the next iterative propagation. If no boundary is left for updating or the iteration time exceeds the predefined threshold, we tend to consider that the patch-based inpainting has converged. Since the occluded regions of missing data in lidar point clouds are not always closed, the threshold is likely to be set depending on the size of the missing regions to prevent the algorithm from a nonconvergence. In practice, the threshold is usually set to 8-10 times of the number of boundary points in the occluded hole.



(a) Input with occlusion (b) Doria *et al.* [3] (c) Sahay *et al.* [5] (d) Hu *et al.* [8] (e) The proposed PLGP (f) Ground Truth

Fig. 4. Inpainted results of different methods for Sarah (row 1), Loot (row 2) with occlusions marked in circle, with a representative cube magnified.

III. Experiment Analysis

A. Implementation

The proposed method is implemented with C++ on a computer with Intel(R) Xeon(R) CPU E5-1620 v2 @ 3.70GHz, 16GB RAM and NVIDIA TITAN V 12 GB.

B. Experimental Setup

The proposed PLGP method is evaluated on several point cloud datasets coming from Stanford [16], JPEG Pleno [17], Paris-Lille-3D dataset [18] and Visual Computing Lab [19]. We conduct experiments to demonstrate its effectiveness with three classical competing works for lidar point cloud inpainting, including Doria *et al.* [3], Sahay *et al*. [5] and Hu *et al.* [8]. For comparison, F-score, consumed-time and the geometric distortion metrics GPSNR in [20] are applied as the quantified metrics to evaluate the inpainted results. F-score is a comprehensive indicator relates to recall and precision, which helps compare the similarity between the generated point clouds and groundtruth in 3D reconstruction. GPSNR is usually used to measure the errors between two point cloud sets. A higher GPSNR or a higher F-score means a lower difference between two point cloud sets.

C. Experimental Results

Subjective evaluation. Fig. 4 describes two representative comparisons of the inpainted results for *Sarah* and *Loot*, in which, row 1 delineates a one-sided inpainting example and row 2 presents a double-sided one. In the case of *Sarah*, Doria *et al.* [3] shows a clear-texture result, even covers the entire area around the hole, because it fills in the occluded region with image-based pixels in the 2D range image, which inevitably produces dislocation points and carries back the alternate texture structure in the transformation from a 2D environment to 3D point clouds. Sahay *et al.* [5] demonstrates an unsmooth inpainted voxel-patch that extends to the perimeter of the occlusion, which reveals its imprecise occluded-boundaries detection and unmatched patch-filling, since it projects the point clouds into a depth image, retrieves a similar one from an online depth database via dictionary learning for inpainting. Hu *et al.* [8] exhibits impressive visual effect in the one-sided inpainting, but the contents they fill in the occlusion look incoherent with stripy geometry around the hole, which may be due to the whole-patch filling that searched from elsewhere. In the case of *Loot*, the results of Doria *et al.* and Sahay *et al.* generate artificial contours, since they attempt to connect the hole's boundary with planar structures in a 2D depth image without smoothing when back to the 3D space. The result in [8] presents lower consistency with the existing points and lower smooth in the local region when compared to the ground truth, indicating spatial inconsistency to some extent. In comparison, our results shown in row 1 and row 2 illustrate that the proposed PLGP is able to inpaint occluded holes with appropriate geometric structure and smoothness over the missing regions since it propagates both the valid point clouds and structure along the local geometry of the occluded boundary. During the inpainting, PLGP does not transform the input point clouds into a 2D range image to perform the detection and inpainting, hence the biggest highlight of PLGP is its low time consumption and less dislocation points in the inpainted results.

Objective evaluation. Table I, Table II and Table III illustrate the inpainted values of GPSNR, F-score and consumed-time respectively in some test point cloud sets. We find that our proposed PLGP achieves obviously better performance on the consumed-time and shows competitive potential significantly on the indicators of GPSNR, F-score. Specifically, in Table I we achieve 23.45 dB gain in GPSNR on average over Doria *et al.* [3], 17.73 dB over Sahay *et al.* [5] and 6.38 dB over Hu *et al.* [8]. In Table II, the performances of our PLGP have

TABLE I
PERFORMANCE COMPARISON IN THE METRIC OF GPSNR (DB)

	Doria *et al.* [3]	Sahay *et al.* [5]	Hu *et al.* [8]	**PLGP**
Andrew	47.6824	29.7906	51.2801	**58.3861**
David	34.8201	22.0725	46.6460	**49.8252**
Philips	25.8533	48.6973	47.6706	**60.0475**
Ricardo	14.3950	39.2636	41.6405	**46.8720**
Sarah	32.5954	44.1576	53.4859	**57.4772**

TABLE II
PERFORMANCE COMPARISON IN THE METRIC OF F-SCORE

	Doria *et al.* [3]	Sahay *et al.* [5]	Hu *et al.* [8]	**PLGP**
Andrew	0.785	0.733	0.816	**0.887**
David	0.590	0.684	0.785	**0.806**
Philips	0.528	0.803	0.768	**0.892**
Ricardo	0.421	0.752	0.612	**0.795**
Sarah	0.583	0.764	0.824	**0.873**

TABLE III
PERFORMANCE COMPARISON IN THE METRIC OF CONSUMED-TIME (S)

	Doria *et al.* [3]	Sahay *et al.* [5]	Hu *et al.* [8]	**PLGP**
Andrew	358.65	288.24	213.29	**146.52**
David	407.57	353.10	259.74	**163.30**
Philips	374.09	265.38	235.65	**98.39**
Ricardo	206.54	142.62	161.31	**138.68**
Sarah	335.68	208.36	226.91	**70.58**

improved in F-score by an average of 47.47% over [3], 24.80% over [5] and 12.67% over [8]. In Table III, the proposed PLGP takes much lower consumed-time, obtaining a 61% lower on average than [3], 47% than [5], 42% than [8]. In the point cloud set of *Philips* that contains more than 300,000 points, PLGP indicates the best inpainted performance with the maximum values in GPSNR and F-score while it only takes very short time compared with other methods. In another case of *Sarah,* It also gets obviously better result with pleased values in the three metrics. Objective indicators of the comparisons exhibit our proposal's competitive ability in point clouds inpainting.

IV. CONCLUSION

In this paper, we propose a novel and efficient inpainting approach for point clouds called PLGP. The key idea is to conduct the occluded hole detection directly in the 3D space, and iteratively applies the natural correlation of local geometric around the hole to quickly search for appropriate match and propagate it into the missing regions along the local detailed structure. It can largely facilitate the inpainting of defect point cloud surface in many conditions and maintains spatial consistency, both in objective and subjective quality. The big contribution of PLGP is its lower consumed-time while ensuring competitive inpainted results. In comparison to the three classical existing methods, our proposed work presents the best performance in the widely-used four test databases.

REFERENCES

[1] C. Dinesh, I. V. Bajić and G. Cheung, "Adaptive Nonrigid Inpainting of Three-Dimensional Point Cloud Geometry," *IEEE Signal Processing Letters*, vol. 25, no. 6, pp. 878-882, Jun. 2018.

[2] Z. Fu, W. Hu and Z. Guo, "3d Dynamic Point Cloud Inpainting Via Temporal Consistency on Graphs," in *IEEE International Conference on Multimedia and Expo (ICME)*, London, UK, 2020, pp. 1-6.

[3] D. Doria and R. J. Radke, "Filling large holes in LiDAR data by inpainting depth gradients," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Providence, RI, 2012, pp. 65-72.

[4] M. Pauly, N. J. Mitra, J. Giesen, M. H. Gross, and L. J. Guibas, "Example-based 3D scan completion", in *Proc. Eurographics Symp. On Geometry Processing*, Vienna, Austria, 2005, pp. 23-32.

[5] P. Sahay and A. N. Rajagopalan, "Geometric inpainting of 3D structures," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Boston, MA, 2015, pp. 1-7.

[6] J. Wang and M. M. Oliveira, "Filling holes on locally smooth surfaces reconstructed from point clouds," *Image & Vision Computing*, vol. 25, no. 1, pp. 103-113, Jan. 2007.

[7] Y. Quinsat and C. Lartigue, "Filling holes in digitized point cloud using a morphing-based approach to preserve volume characteristics," *International Journal of Advanced Manufacturing Technology*, vol. 81, no. 1-4, pp. 411-421, Oct. 2015.

[8] W. Hu, Z. Fu and Z. Guo, "Local Frequency Interpretation and Non-Local Self-Similarity on Graph for Point Cloud Inpainting," *IEEE Trans. on Image Processing*, vol. 28, no. 8, pp. 4087-4100, Aug. 2019.

[9] Z. Fu, W. Hu, and Z. Guo, "Point cloud inpainting on graphs from non-local self-similarity, " *IEEE International Conference on Image Processing (ICIP)*, Athens, Greece, 2018.

[10] P. Lai, Y. Huang and S. Chien, "Surface-based background completion in 3D scene," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Washington, DC, 2016, pp. 1218-1222.

[11] H. Sun, X. Liu, Q. Deng, W. Jiang, S. Luo and Y. Ha, "Efficient FPGA Implementation of K-Nearest-Neighbor Search Algorithm for 3D LIDAR Localization and Mapping in Smart Vehicles," *IEEE Trans. on Circuits and Systems II: Express Briefs*, vol. 67, no. 9, pp. 1644-1648, Sept. 2020.

[12] J. Tachella, Y. Altmann, S. McLaughlin and J. -Y. Tourneret, "Real-Time 3D Color Imaging with Single-Photon Lidar Data," in *IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, Le gosier, Guadeloupe, 2019, pp. 206-210.

[13] N. Aranjuelo, G. Engels, L. Unzueta, I. Arganda-Carreras and O. Otaegui, "Robust 3D Object Detection from LiDAR Point Cloud Data with Spatial Information Aggregation, " in *International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO)*, Burgos, Spain, 2020, pp. 813-823.

[14] Z. Cai, C. Wang, C. Wen and J. Li, "Occluded Boundary Detection for Small-Footprint Groundborne LIDAR Point Cloud Guided by Last Echo," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 11, pp. 2272-2276, Nov. 2015.

[15] P. Goyal, J. S. Challa, D. Kumar, et al. Grid-R-tree: a data structure for efficient neighborhood and nearest neighbor queries in data mining. *International Journal of Data Science and Analytics*, vol. 10, no. 10, pp. 25-47, Apr. 2020.

[16] M. Levoy, J. Gerth, B. Curless, and K. Pull, "The Stanford 3D scanning repository," [Online] https://graphics.stanford.edu/data/3Dscanrep/.

[17] https://jpeg.org/plenodb/.

[18] https://npm3d.fr/paris-lille-3d.

[19] http://vcl.iti.gr/dataset/reconstruction/.

[20] D. Tian, H. Ochimizu, C. Feng, R. Cohen and A. Vetro, "Geometric distortion metrics for point cloud compression," in *IEEE International Conference on Image Processing (ICIP)*, Beijing, 2017, pp. 3460-3464.