Pipe-Run Extraction and Reconstruction from Point Clouds

Rongqi Qiu[1], Qian-Yi Zhou[2], and Ulrich Neumann[1,⋆]

[1] University of Southern California, USA
[2] Stanford University, USA

Abstract. This paper presents automatic methods to extract and reconstruct industrial site pipe-runs from large-scale point clouds. We observe three key characteristics in this modeling problem, namely, primitives, similarities, and joints. While *primitives* capture the dominant cylindric shapes, *similarities* reveal the inter-primitive relations intrinsic to industrial structures because of human design and construction. Statistical analysis over point normals discovers primitive similarities from raw data to guide primitive fitting, increasing robustness to data noise and incompleteness. Finally, *joints* are automatically detected to close gaps and propagate connectivity information. The resulting model is more than a collection of 3D triangles, as it contains semantic labels for pipes as well as their connectivity.

1 Introduction

3D digital models for industrial sites are crucial in many applications, including operator training, disaster simulations and response planning. As a dominant feature of industrial sites, pipe-runs are an important part of operations and maintenance. In older facilities, initial CAD models may not exist or are out of date, prompting the need for creating new models. While modern laser scanners can produce dense point clouds capturing the surface geometry, the automated transformation of point clouds to pipe-run models including cylinder geometry and accurate connectivity remains an open problem.

A popular strategy of 3D reconstruction from point scans is to simplify a triangular mesh that minimizes the geometric fitting error with respect to the input points (*e.g.*, [2,5,14]). However, pure data-driven methods (*e.g.*, Ball-Pivoting Algorithm [2]) have no capability to simplify or filter noisy input data and generate 3D meshes with rough surfaces and cracks that are faithful to the point scans (*e.g.*, Figure 2(d)). What is more, the resulting triangles contain no structural semantic or connectivity data.

Another strategy of reconstruction attempts to fit *primitives* (*e.g.*, cylinders, spheres and planes) to the raw data, capturing geometric information conveyed

⋆ The authors thank Suya You and anonymous reviewers for their valuable comments. This work is supported by Chevron, USA under the joint project, Center for Interactive Smart Oilfield Technologies and an Annenberg Graduate Fellowship from USC.

D. Fleet et al. (Eds.): ECCV 2014, Part III, LNCS 8691, pp. 17–30, 2014.

in the input points (*e.g.*, [11,24,23]). This strategy is well-suited to industrial sites since most parts of them are composed of primitive shapes ([12,20,21]). However, such methods rarely extract complete pipe-runs with accurate connectivity. Moreover, bottom-up primitive fitting adopted in these methods is non-robust due to sensitivity to noise and outliers (Figure 2(e)).

We present an automatic robust method to reconstruct pipe-runs in 3D point clouds (*e.g.*, Figure 1). As cylindric shapes are often the dominant geometry of interest in such sites, we focus on methods to reconstruct pipes and joints. In addition, we make use of global similarities because they are more stable than local features in the noisy and complex input data. Our method differs from prior uses of global regularities enforced as a post-process (*e.g.*, [18]). The effectiveness of post processing is bounded by the unreliable initial primitive detection (*e.g.*, Figure 2(f)). Instead, we introduce global similarities in the early detection stage to improve the primitive detection and fitting processes. Finally, to reliably capture primitive junctions and propagate connectivity between primitives, we detect *joints* between adjacent pipes. Our combined methods robustly reconstruct complete pipe networks from point clouds of industrial structures (*e.g.*, Figure 2(b)(c)).

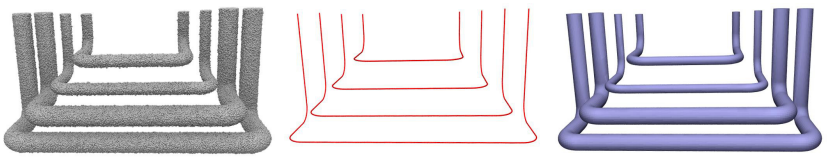

Fig. 1. Given a point cloud of industrial structures (a), our method automatically extracts a pipe axis network (b) and reconstructs pipe-runs (c)

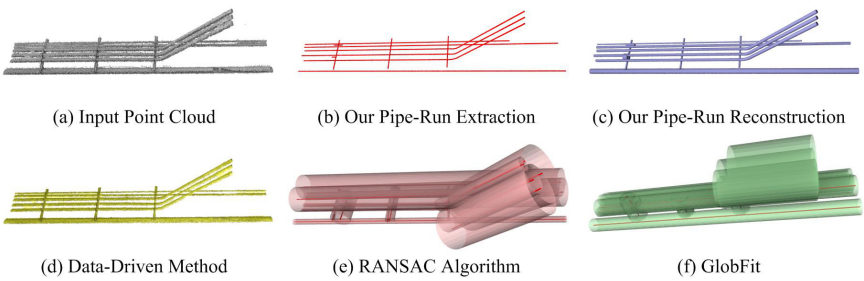

Fig. 2. Given a real-world scan of industrial structures (a), we show 3D reconstruction results from different approaches: (b) extracted pipe axis network by our approach, (c) reconstructed pipe-runs by our approach, (d) Ball-Pivoting Algorithm [2], (e) RANSAC algorithm [23], and (f) GlobFit [18]

2 Related Work

2.1 Modeling from Point Clouds

A general strategy of 3D reconstruction from point clouds is to simplify a triangular mesh model that minimizes the distance between the input points and the mesh surface. Research efforts following this strategy are usually known as *data-driven* reconstruction because they take the input data as *truth* and make a trade-off between the size of the mesh and the geometry fitting error [2,14]. Different heuristics are introduced to produce modeling preferences such as smoothness [3,17] and sharp feature [9]. Data-driven reconstruction has the advantage of generality and adaptation to a variety of input point clouds. However, none of these methods can produce semantic or connectivity information, which limits their application.

Introducing prior knowledge of object shapes can significantly reduce the solution space and thus simplifies the reconstruction problem. For instance, Schnabel *et al.*[23] present an efficient RANSAC approach to detect primitive shapes from point clouds. Hofer *et al.*[13] adopt line geometry for the recognition and reconstruction of 3D surfaces. Many methods of fitting primitives are designed for reverse engineering such as [1]. Another existing method employs Hough transform [15]. Even though efforts have been made to reduce their space and time complexity [21], memory size limits still make these methods impractical for large-scale input.

2.2 Global Similarity in Reconstruction

In addition to the prior knowledge of primitive shapes, higher level of knowledge representing the similarities and relations between primitive elements are also introduced and explored by Li *et al.*[18]. This work deals with small scale objects and produces primitives exhibiting global relations. However, as stated in Section 1, it relies heavily on fitting quality of primitives, thus loses accuracy when dealing with complicated industrial structures. Our approach, on the other hand, overcomes this drawback by introducing similarities among primitives in an early stage, thus it is more successful at handling large-scale industrial sites.

Top-down geometry reconstruction is studied by Chen and Chen [4]. It employs statistical models on point normals to detect planar regions. A similar strategy in our approach focuses on reconstructing the cylinders and joints dominating in industrial sites.

2.3 Pipe-Run Reconstruction

The problem of pipe-run reconstruction has drawn much attention in both academia and industry. For instance, Liu *et al.*[19] attempt to reconstruct pipeline plants by reducing the reconstruction problem into a set of circle detections. However, their work uses prior knowledge of specific scenes such as the ground plane. Their assumption that pipes are either orthogonal or parallel to the ground

is not general (*e.g.*, Figure 2). Fu *et al.*[10] adopt local RANSAC detection of pipes and uses standard elbow pieces to connect pipes. This method produces connected pipe-runs but only handles joints of some pre-defined size and shape. Commercial software (*e.g.*, [6]) is also available to interactively reconstruct pipe-runs. However, these products usually require substantial manual work. Our method, on the other hand, is fully automatic without any user intervention.

3 Overview

Our processing pipeline accepts point clouds as input and models pipe-runs in the scene. It is composed of the following steps (Figure 3).

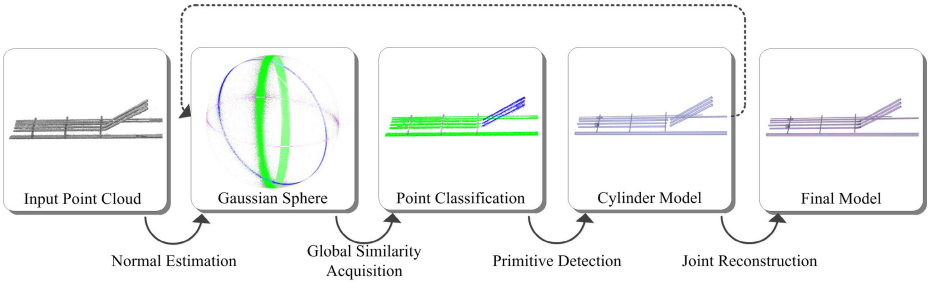

Fig. 3. Our modeling pipeline. Normals of input points are estimated and projected onto a Gaussian sphere, where patterns of great circles are detected to determine orientations of cylinders. Points within the same orientation are further separated to decide placement of different primitives. Orientation detection and placement extraction are iteratively performed until all primitives are detected. Joints between cylinders are then detected and generated to connect pipes into complete pipe-runs in the final models.

- **Global Similarity Acquisition:** We make a key observation that discovering global similarities is more robust than fitting primitives as the first stage. The reason is that global similarities appear more stable than local features in the noisy input point clouds. Thus, the first step of our approach analyzes the orientations of cylinders since they often exhibit similarities due to common design and construction practices. Statistical analysis on point normals is applied for extracting orientations of primitives.
- **Primitive Detection:** We use the global similarity information extracted from the last step to reduce the degrees of freedom of primitive fitting. This sequence significantly increases the robustness of primitive detection relative to the use of only local data for fitting. Points with the same orientation are projected onto an orthogonal plane used to detect 2D circles. Cylinders are detected within points contributing to the circle projections.
- **Joint Reconstruction:** The preceding stage typically extracts a large number of disconnected pieces of pipes. The next stage links them into a fully-connected pipe network. Three kinds of joints (*i.e.*, T-junctions, curved joints and boundary joints) are constructed to connect pipes smoothly.

4 Global Similarity Acquisition

While global similarities exist in any portions of an industrial scene point cloud, we observe that local sub-regions usually exhibit more consistent similarities than that in the whole scene. Therefore, we divide the whole scene into uniform cubic sub-volumes to process separately, and then seamlessly merge the results together in a later stage. This divide-and-conquer strategy is efficient for discovering global similarities. It also enables our system to handle arbitrary-sized input without encountering memory size limitations.

The most significant global similarities are orientations of cylinders. In a local region, cylinders tend to group into a few directions, denoted as *principal directions* in this paper. We detect these principal directions by adopting the following fact: if a point lies on a cylinder, its normal is perpendicular to the cylinder axis. Therefore, the point normals from cylinders of the same direction $\mathbf{d}$ will all be perpendicular to $\mathbf{d}$. When mapped onto a Gaussian sphere, they will distribute as the great circle that is perpendicular to $\mathbf{d}$, as illustrated in Figure 4(a). Therefore, we detect principal directions by mapping all the point normals onto the Gaussian sphere and detect great circle patterns. In the ideal case without noise, RANSAC [8] will reliably detect great circles. In particular, we randomly choose a pair of normals $(\mathbf{n}_i, \mathbf{n}_j)$, compute a perpendicular direction

$$\mathbf{d} = \mathbf{n}_i \times \mathbf{n}_j, \tag{1}$$

and validate the direction $\mathbf{d}$ with the number of points on the corresponding great circle, i.e., the size of set $\{\mathbf{p} | 1 - |\mathbf{n_p} \cdot \mathbf{d}| < \epsilon\}$. Once a principal direction is detected, we remove the points that contribute to that direction, then iteratively perform the detection process until all the principal directions have been found in this local region. In practice, however, the data is noisy and the points form thick rings around a great circle (e.g., green samples in Figure 4(a)). Thus a simple RANSAC becomes unstable and hard to converge. We adopt unsupervised clustering in the space of cylinder directions to solve this problem. In particular, we choose many random point pairs and compute cylinder direction candidates

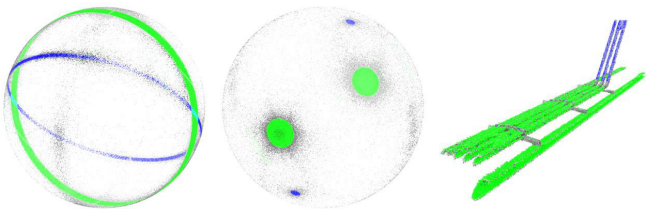

Fig. 4. The process of global similarity acquisition: point normals are projected onto a Gaussian sphere (a) where great circles are detected by transforming to a cylinder detection sphere (b); points are then segmented based on cylinder orientation (c)

that lie on a spherical map of potential cylinder direction (Figure 4(b)). The intuition is that even with noisy normal directions, the computed cylinder directions will lie close to the true principal direction. Therefore, we use Mean-shift [7] to detect modes on the spherical map of potential cylinder directions. In particular, starting from a random sample $\mathbf{x}$, it is iteratively updated with:

$$\mathbf{x} \leftarrow \frac{\sum_{\mathbf{x}_i \in N(\mathbf{x})} K(\| \mathbf{x} - \mathbf{x}_i \|)\mathbf{x}_i}{\sum_{\mathbf{x}_i \in N(\mathbf{x})} K(\| \mathbf{x} - \mathbf{x}_i \|)}, \tag{2}$$

where $K(\cdot)$ is Gaussian kernel function. After each iteration, the sample center $\mathbf{x}$ is not guaranteed to stay on the sphere. Therefore, we coerce it back onto the sphere. The centers of the modes are adopted as the principal directions in this local region.

Segmentation: A by-product of global similarity acquisition is the point segmentation based on cylinder orientations (Figure 4(c)). In particular, we identify points within a thick stripe on the Gaussian sphere as a category with the same cylinder orientation.

5 Primitive Detection

So far we have extracted principal directions of cylinders. In this section we show how global information helps extracting cylinder primitives. This task is accomplished in two steps: first, cylinder positions are discovered by mapping associated points on a plane and detecting circles; second, cylinder boundaries are determined.

5.1 Cylinder Position Calculation

We take points belonging to cylinders of the same direction $\mathbf{d}$. Intuitively, by mapping these points onto a plane that is perpendicular to the cylinder direction $\mathbf{d}$ (*e.g.*, Figure 5(a)), the projected points exhibit circular patterns (*e.g.*, Figure 5(b). Therefore, instead of fitting 3D cylinders, we detect circles on a 2D projection plane of direction $\mathbf{d}$.

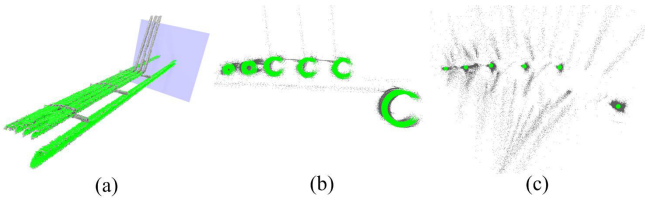

Fig. 5. The process of calculating cylinder positions: segmented points are projected to an orthogonal plane (a), where circular patterns are detected (b) by transforming to a circle-center map (c)

We also note that the normals of points can be projected onto the projection plane as 2D directions pointing either towards or outwards the cylinder center $\tilde{\mathbf{c}}$. Therefore, two projected points and normals are enough to determine a circle on the projection plane. Specifically, given two 2D points as $\tilde{\mathbf{p}}$, $\tilde{\mathbf{q}}$ and their 2D normals as $\tilde{\mathbf{n}}_{\mathbf{p}}$, $\tilde{\mathbf{n}}_{\mathbf{q}}$, the circle center $\tilde{\mathbf{c}}$ is determined by the intersection of two lines $L_1(u) = \tilde{\mathbf{p}} + u\tilde{\mathbf{n}}_{\mathbf{p}}$ and $L_2(v) = \tilde{\mathbf{q}} + v\tilde{\mathbf{n}}_{\mathbf{q}}$. The radius r is then calculated by $r = \frac{|\tilde{\mathbf{c}}\tilde{\mathbf{p}}|+|\tilde{\mathbf{c}}\tilde{\mathbf{q}}|}{2}$, if $|\tilde{\mathbf{c}}\tilde{\mathbf{p}}| \approx |\tilde{\mathbf{c}}\tilde{\mathbf{q}}|$.

However, in the presence of noise, projected cylinder points are distributed near circles (*e.g.*, Figure 5(b)). To address this issue, we choose many point pairs to get a collection of candidate circles. Our observation is that these candidate circles tend to form clusters (*e.g.*, Figure 5(c)) and the centers of clusters approximate cross-sections of cylinders. Mean-Shift algorithm [7] is adopted to detect these clusters, corresponding to cylinders and their associated points.

5.2 Cylinder Boundary Extraction

Given the positions of cylinders, the remaining parameters to be determined are the boundaries (*i.e.*, the start and end of the cylinder axis). End points are determined by the point coverage along cylinder surfaces. To be specific, we discretize the maximum discovered cylinder extents into small axial sections. The existence of each section is examined and the verified ones are reconstructed into contiguous cylinder pieces (*e.g.*, Figure 6(b)).

Pipe segment existence is based on two coverage tests. Axis coverage is a simple measure of point density per linear length of pipe. We set a minimum threshold for the total number of points along a valid pipe segment. Cross section coverage is a measure of the distribution of points around a pipe axis. We observe that plane normals can easily be mistaken for sides of cylinders by our algorithm. To avoid this, we require cylinders to exhibit more than 180-degrees of cross-section coverage.

In real-world scans, the gaps along pipes (*e.g.*, Figure 6(a)) are usually created with data incompleteness due to occlusions, data loss, and noise. To avoid over-segmenting a continuous pipe, we apply morphological operations of opening and closing to smooth the result. (*e.g.*, Figure 6(c)).

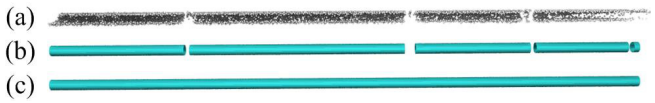

Fig. 6. Given incomplete scans of pipes (a), cylinder boundaries are determined by the point coverage along cylinder surfaces (b) and smoothed using morphological operations (c)

6 Joint Reconstruction

So far we have detected all the cylinder primitives in the scene. To reconstruct complete and fully-connected pipe-runs, we also need to link these cylinders together. We achieve this by introducing joints. Three categories of joints (*i.e.*, T-junctions, curved joints and boundary joints) are detected in our system (*e.g.*, Figure 7). T-junctions are extensions of one cylinder merging into another cylinder. Curved joints are elbows connecting two cylinders to allow a change of direction. Boundary joints are cylinder segments that fill small gaps between two cylinders. The gaps usually appear at the boundary of dividing sub-volumes.

6.1 Positions and Types of Joints

Our first thought was to detect joints as primitives like cylinders. However, it is not a trivial problem since joints are either two small (T-junctions and boundary joints) or have a complicated shape far from a simple primitive with a few parameters (curved joints). We observe that joints must connect two nearby cylinders. Therefore, we hypothesize all the possible joint locations and shapes and then select the most likely cases based on agreement with the point cloud data. Given the huge number of cylinders detected in the preceding stages, enumerating all candidate cylinder pairs would be impractical. Thus we employ several important heuristic criteria for joint positions (Figure 8(a)). Joint radius, gap distance (defined as the nearest distance between central lines), skew and angle are limited to reasonable ranges that are functions of the connecting pipe diameters. We thereby ensure that the connecting cylinders are nearby, similar-sized, co-planar and non-parallel for T-junctions and curved joints (or parallel for boundary joints).

To decide types of hypothetic joints, noting that all joints include at least one cylinder end, we examine extensions of cylinder ends. If an extension intersects with another cylinder, our hypothesis is a T-junction. If an extension intersects the extension of another cylinder, our hypothesis is a curved joint. If an extension coincides with another cylinder, our hypothesis is a boundary joint.

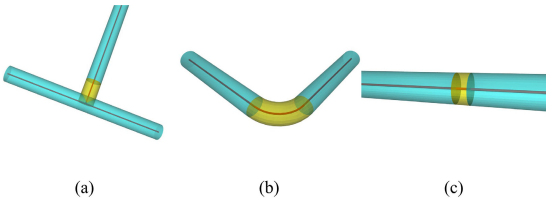

Fig. 7. Three types of joints included in our system: (a) T-junctions, (b) curved joints, and (c) boundary joints

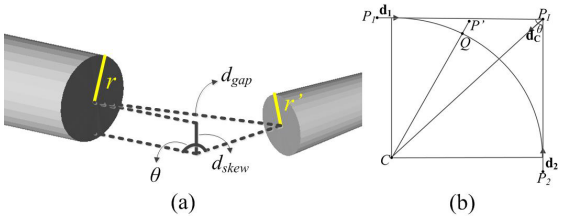

Fig. 8. (a) Four criteria of joint positions. (b) The process of calculating major radius of curved joints.

6.2 Reconstruction of Joints

The reconstruction of T-junctions and boundary joints is straight-forward, because all of their parameters has been determined. For T-junctions, the joint can be modeled by extending the end point of one cylinder into the axis of another cylinder. For boundary joints, we reconstruct a cylinder connecting two adjacent ones.

If two cylinders are connected with a curved joint, the only free parameter is the major radius. We determine the major radius of the optimal curved joint as the one with the most points lying on its surface among the range of possible major radius options. The intuition is that if we make every data point in the hypothetical joint volume vote for radius values such that the joint surfaces touch it, the value with most votes would be the optimal radius. The following portion shows how to calculate the major radius for a given data point.

As shown in Figure 8(b), assume the end points of two cylinders are P_1 and P_2 with corresponding direction $\mathbf{d_1}$ and $\mathbf{d_2}$. Since the two cylinders are near co-planar, we first force them onto the same plane, and then calculate the intersection of their axes as P_I. The goal is to find the circle center C and major radius R, so that the circle is tangent to both P_1P_I and P_2P_I. It can be shown that C must lie on the interior bisector of $\angle P_1P_IP_2$ (denoted as θ). If $\mathbf{d}_C$ is the unit direction of this bisector, then C and R can be expressed as:

$$\begin{aligned} C &= P_I + |CP_I| \cdot \mathbf{d}_C, \\ R &= |CP_I| \cdot \sin\frac{\theta}{2}. \end{aligned} \tag{3}$$

For every data point P, we first need to decide which point (denoted as Q) on circle C is the circle center of the cross section that contains P. It can be shown that the line connecting projected point P' and C intersects circle C at Q. Since $|P'Q|$ is known ($|P'Q| = \sqrt{r^2 - d^2}$, where r is cylinder radius and d is projection distance of P), $|CP'|$ can be expressed by:

$$|CP'| = R \pm |P'Q|. \tag{4}$$

Bringing Equation 3 into Equation 4 gives a quadratic equation of R, where R can be solved.

7 Experimental Results

To evaluate our method, we test the algorithm on both synthetic datasets and real-world datasets. Synthetic data is created by subsampling CAD models. Various levels of noise and outliers are manually added to test robustness of different methods. We also perform experiments on real-world scans of a large industrial site containing a variety of pipes, joints and other structures. Our approach is compared to both sequential RANSAC [23] and GlobFit [18] as post-application of global similarities. For fair comparisons, we use the same set of parameters throughout all experiments within each method.

7.1 Synthetic Dataset Experiments

Figure 9 shows that the RANSAC method [23] neglects global relationships of primitives and gets lost in local errors. GlobFit [18] takes global similarities into consideration, but is still bounded by the effects of initial RANSAC. Our method, on the other hand, discovers global similarities before fitting primitives and produces clean and accurate models.

Noise Experiments. To test the robustness of our algorithm, we add various levels of noise to the input. Two types of noise are tested, *i.e.*, surface noise and background noise. Surface noise (K_S in Figure 10) is random Gaussian

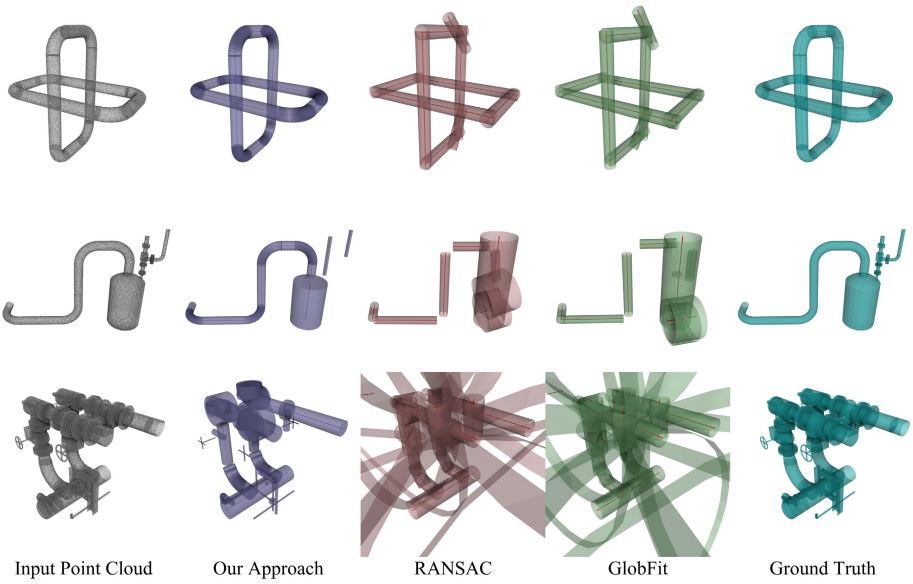

Fig. 9. Experiments on synthetic datasets. From left to right: input point cloud, our result, RANSAC [23] result, GlobFit [18] result and ground truth.

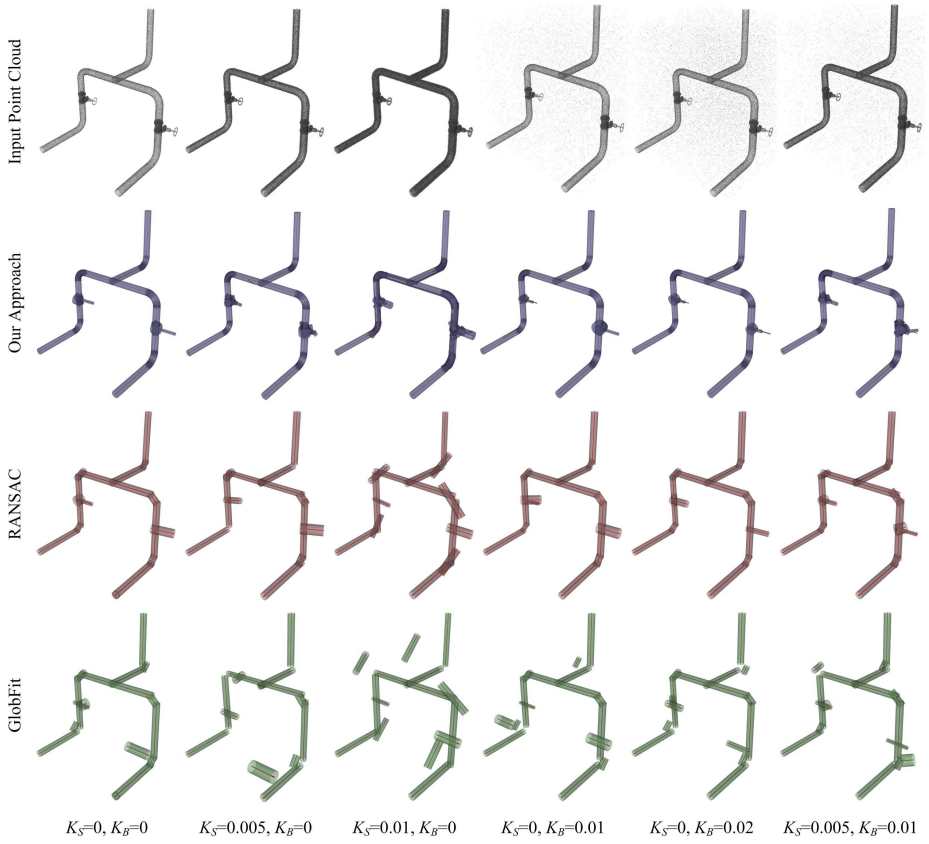

Fig. 10. Reconstruction under various noise levels. From top to bottom: input point cloud, our result, RANSAC [23] result and GlobFit [18] result. K_S stands for surface noise and K_B stands for background noise.

noise in the direction of surface normals. Background noise (K_B in Figure 10) is uniformly random noise in 3D space. Both types of noise are to simulate data problems encountered in real world scans such as scanner hardware noise or registration error. As shown in Figure 10, our results remain stable as the noise level increases, while RANSAC [23] results are dramatically affected by noise, even if global similarities are enforced as a post-process using GlobFit [18].

7.2 Real-World Dataset Experiments

Figure 11(b) shows models reconstructed from 381M LiDAR points (Figure 11(a)) in a 25 meters × 25 meters × 10 meters scene. In pre-processing, we employ a voxel-grid method [22] to make the surface point density more uniform. We observe that planar objects may cause problems for our method, so we pre-filter points in planar areas (occupy 49.5% of points) using a method presented in [16]. Our method suc-

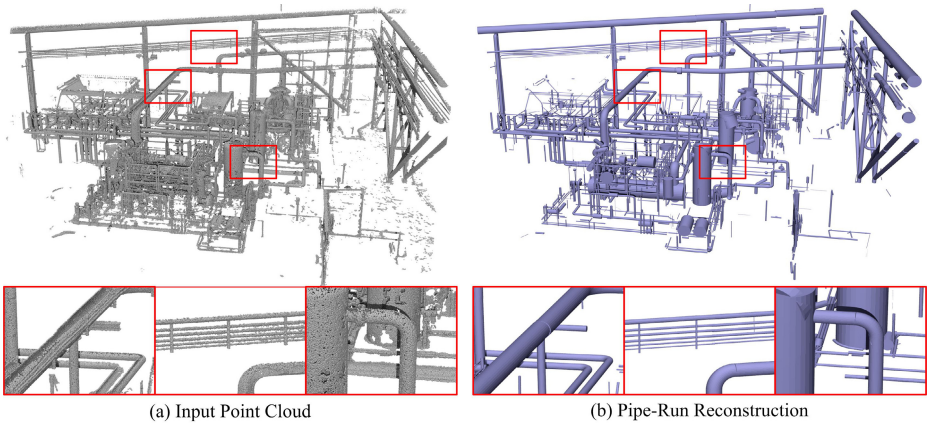

Fig. 11. Experiments on a real-world dataset: (a) input point cloud (1/40 subsampled); (b) pipe-run reconstruction

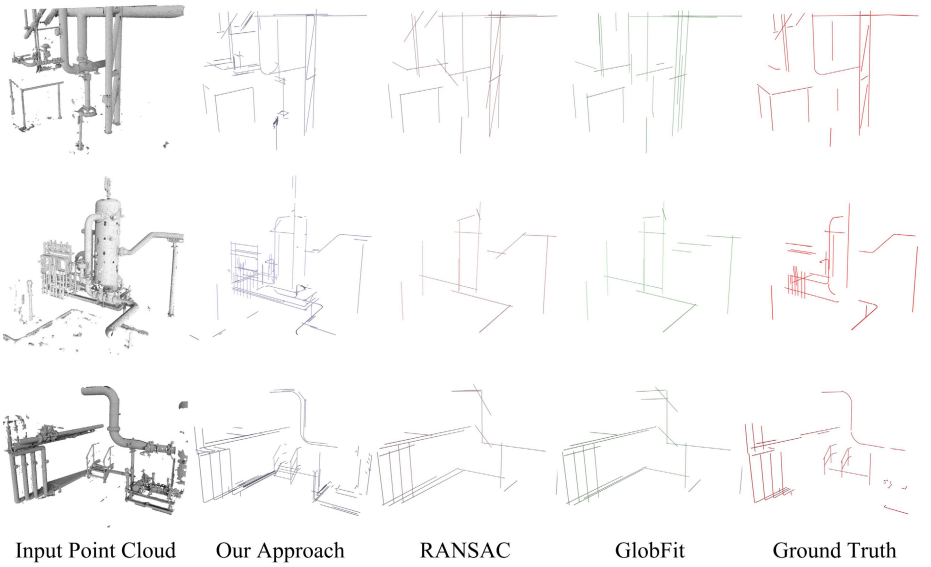

Fig. 12. Extracted pipe axis networks on real-world datasets. From left to right: input point cloud, our result, RANSAC [23] result, GlobFit [18] result and ground truth.

cessfully extracts pipe-run networks within 24 hours on a consumer-level desktop (Intel Core i7-3610QM 2.30GHz CPU with 6GB RAM). A detailed comparison with commercial software [6] is included in the supplementary material.

To quantitatively evaluate pipe-run extraction, we manually mark pipe networks of three sub-volumes to be compared with automatically extracted pipe-runs

Table 1. Quantitative evaluation of results shown in Figure 12

Models in Figure 12	First row			Second row			Third row		
	RANSAC [23]	GlobFit [18]	Our approach	RANSAC [23]	GlobFit [18]	Our approach	RANSAC [23]	GlobFit [18]	Our approach
Precision	0.804	0.603	0.603	0.745	0.714	0.481	0.629	0.594	0.402
Recall	0.938	0.748	0.953	0.486	0.476	0.986	0.630	0.625	0.915

(Figure 12). The pipe axes are sampled by an interval of 1cm so that each sample represents 1cm of pipe. These sampled points are compared based on proximity to calculate precision and recall for our method, RANSAC [23] and GlobFit [18] (Table 1). Our method achieves scalability by fully utilizing global similarities. Small cylinders are successfully captured (*e.g.*, the handrails and ladders in Figure 12, third row) because their orientations fall into principal directions in the scene and the detected orientations become a strong constraint in cylinder fitting.

8 Conclusion

We describe a novel robust processing pipeline to automatically extract pipe-runs from large-scale 3D point clouds. Our approach introduces the global information in an early stage by applying unsupervised analysis on point normals, and treats orientation detection and placement extraction as two separate sub-problems, thus avoiding degradation by local data errors. Joints are detected and modeled to recover connectivity information and smoothly connect cylinders.

References

1. Benkö, P., Martin, R.R., Várady, T.: Algorithms for reverse engineering boundary representation models. CAD (2001)
2. Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., Taubin, G.: The ball-pivoting algorithm for surface reconstruction. IEEE Transactions on Visualization and Computer Graphics 5(4), 349–359 (1999)
3. Carr, J.C., Beatson, R.K., Cherrie, J.B., Mitchell, T.J., Fright, W.R., McCallum, B.C., Evans, T.R.: Reconstruction and representation of 3d objects with radial basis functions. In: ACM SIGGRAPH (2001)
4. Chen, J., Chen, B.: Architectural modeling from sparsely scanned range data. International Journal of Computer Vision 78(2-3), 223–236 (2008)
5. Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G.: Meshlab: An open-source mesh processing tool. In: Eurographics Italian Chapter Conference (2008)
6. ClearEdge3D: Edgewise plant (December 2012), `http://www.clearedge3d.com/`, `http://www.clearedge3d.com/Products.aspx?show=EdgeWisePlant`
7. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(5), 603–619 (2002)

8. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6), 381–395 (1981)
9. Fleishman, S., Cohen-Or, D., Silva, C.: Robust moving least-squares fitting with sharp features. ACM Transactions on Graphics (TOG) 24, 544–552 (2005)
10. Fu, Y., Zhu, X., Yang, J., Yuan, Z.: Pipe reconstruction from unorganized point cloud data. U.S. Patent Application 12/849, 647 (2010)
11. Gal, R., Shamir, A., Hassner, T., Pauly, M., Cohen-Or, D.: Surface reconstruction using local shape priors. In: Symposium on Geometry Processing (2007)
12. Hakala, D., Hillyard, R., Malraison, P., Nource, B.: Natural quadrics in mechanical design. Proc-AUTOFACT West 1, 17–20 (1981)
13. Hofer, M., Odehnal, B., Pottmann, H., Steiner, T., Wallner, J.: 3d shape recognition and reconstruction based on line element geometry. In: ICCV (2005)
14. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. In: ACM SIGGRAPH (1992)
15. Hough, P.: Method and means for recognizing complex patterns. U.S. Patent No. 3,069, 654 (1962)
16. Huang, J., You, S.: Detecting objects in scene point cloud: A combinational approach. In: 2013 International Conference on 3DTV-Conference (2013)
17. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Symposium on Geometry Processing (2006)
18. Li, Y., Wu, X., Chrysathou, Y., Sharf, A., Cohen-Or, D., Mitra, N.: Globfit: Consistently fitting primitives by discovering global relations. ACM Transactions on Graphics (TOG) 30, 52 (2011)
19. Liu, Y.J., Zhang, J.B., Hou, J.C., Ren, J.C., Tang, W.Q.: Cylinder Detection in Large-Scale Point Cloud of Pipeline Plant. IEEE Transactions on Visualization and Computer Graphics, 1–8 (April 2013)
20. Petitjean, S.: A survey of methods for recovering quadrics in triangle meshes. ACM Computing Surveys (CSUR) 34(2), 211–262 (2002)
21. Rabbani, T., Van Den Heuvel, F.: Efficient hough transform for automatic detection of cylinders in point clouds. ISPRS WG III/3, III/4 3, 60–65 (2005)
22. Rusu, R., Cousins, S.: 3d is here: Point cloud library (pcl). In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–4. IEEE (2011)
23. Schnabel, R., Wahl, R., Klein, R.: Efficient ransac for point-cloud shape detection. In: Computer Graphics Forum, vol. 26, pp. 214–226. Wiley Online Library (2007)
24. Schnabel, R., Degener, P., Klein, R.: Completion and reconstruction with primitive shapes. Computer Graphics Forum (2009)