Pattern Recognition 107 (2020) 107443

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Cylinders extraction in non-oriented point clouds as a clustering problem

Filippo Bergamasco[1,*], Mara Pistellato[1], Andrea Albarelli, Andrea Torsello

DAIS, Ca'Foscari University of Venice, 155 Via Torino, Venice, Italy

ARTICLE INFO

Article history:
Received 19 August 2019
Revised 5 May 2020
Accepted 10 May 2020
Available online 27 May 2020

Keywords:
Cylinder extraction
Dual quaternions
Point clouds
Industrial inspection
Game theory

ABSTRACT

Finding geometric primitives in 3D point clouds is a fundamental task in many engineering applications such as robotics, autonomous-vehicles and automated industrial inspection. Among all solid shapes, cylinders are frequently found in a variety of scenes, comprising natural or man-made objects. Despite their ubiquitous presence, automated extraction and fitting can become challenging if performed "in-the-wild", when the number of primitives is unknown or the point cloud is noisy and not oriented.

In this paper we pose the problem of extracting multiple cylinders in a scene by means of a Game-Theoretic inlier selection process exploiting the geometrical relations between pairs of axis candidates. First, we formulate the similarity between two possible cylinders considering the rigid motion aligning the two axes to the same line. This motion is represented with a unitary dual-quaternion so that the distance between two cylinders is induced by the length of the shortest geodesic path in SE(3). Then, a Game-Theoretical process exploits such similarity function to extract sets of primitives maximizing their inner mutual consensus. The outcome of the evolutionary process consists in a probability distribution over the sets of candidates (ie axes), which in turn is used to directly estimate the final cylinder parameters. An extensive experimental section shows that the proposed algorithm offers a high resilience to noise, since the process inherently discards inconsistent data. Compared to other methods, it does not need point normals and does not require a fine tuning of multiple parameters.

1. Introduction

The availability of depth data and their integration with classical imaging techniques allows the fulfilment of several tasks which were challenging, or even infeasible, in the past. For this reason, 3D data are widely employed in a variety of fields, ranging from cultural heritage [1] to industrial inspection [2,3]. Many practical applications employing point cloud data require the extraction and identification of one or more well-known objects included in the scene: some examples are human pose estimation [4], object segmentation [5,6] and model fitting [7]. Despite the high quality offered by modern acquisition devices, in some situations 3D data captured from a scene are far from being perfect. This can be due to limitations of the working environment (light conditions, occlusions, etc.) or to the nature of the scanned objects themselves. For instance, the scanning process of objects presenting shiny metallic areas could easily lead to an incomplete or noisy 3D surface [8]. Furthermore, many times the acquired scene is complex and contains several elements which are to be filtered out as they are not relevant for the subsequent analysis. For these reasons any practical shape extraction algorithm must provide a reliable method excluding outliers in addition to a good parameter estimation.

The fitting and extraction of geometrical primitives (planes, circles, spheres, cylinders, pyramids) is a widely covered topic in the literature [9–11], being an essential task in several scenarios as industrial automation and inspection [12,13], reverse engineering [14], scene segmentation [15], marker detection [16] or camera calibration [17]. In particular we are interested on cylindrical shapes, often found in many applications including natural landscape analysis [18], automated pipe-run reconstruction [19], and industrial quality inspection [20].

In general, when dealing with primitives identification, we can distinguish two conceptually different tasks, namely *fitting* and *extraction*, which differ in the type of problem and how it is addressed. In the case of fitting the point cloud has already been segmented, so a subset of 3D points is associated to a possible cylindrical shape. So, the objective of a fitting algorithm is to compute the best parameters according to the given data, typically mini-

* Corresponding author.
E-mail addresses: filippo.bergamasco@unive.it (F. Bergamasco), mara.pistellato@unive.it (M. Pistellato), albarelli@unive.it (A. Albarelli), andrea.torsello@unive.it (A. Torsello).
[1] Equal contribution.

https://doi.org/10.1016/j.patcog.2020.107443
0031-3203/

mizing the average error with respect to the theoretical surface. On the other hand, the goal of extraction methods is to detect the primitives included in a complex point cloud and then devise their parameters. These approaches are often employed when the whole scene is captured, making the task more challenging than a simple fitting since data could be incomplete, noisy or include clutter. Usually there are no assumptions on the number of primitives in the scene, neither on their orientations. Moreover, the presence of point normals is not always ensured and their quality may depend on the acquisition device or the used reconstruction algorithm. Nevertheless, the majority of approaches in the literature assume to either have point normals or to compute a good approximation of them using PCA.

In this manuscript we propose a cylinder extraction technique that does not require point normals and considers cylinder extraction as a clustering problem. We first formulate a cylinder similarity function expressing the compatibility between two possible candidates, and a simple yet effective candidate extraction technique based on geometrical properties of cylindrical primitives. These two ingredients are combined in a clustering process based on evolutionary game theory, that ensures the extraction of the subset of candidates with best mutual support, exhibiting excellent robustness to outliers. In particular, we extended our seminal work described in [21] by: (i) formulating the approach as a clustering problem and adding a more in-depth discussion of the cylinder similarity function; (ii) proposing a new technique exploiting manifold mean to provide a better parameter estimation after the inlier selection process, and (iii) expanding the experimental section including an extensive study of the method's performances with different kinds of noise.

Our approach offers three main advantages: first, it does not require point normals nor any prior knowledge of the scene. Second, the game-theoretic inlier selection process exhibits high resilience with respect to outliers without an initial tuning of several parameters. Third, we achieve good estimation accuracy even if most of the cylindrical shapes are occluded or covered by clutter. For these reasons, our method is suitable in applications for which the fitting accuracy is at a paramount importance like robot grasping or industrial quality inspection.

2. Related work

Cylindrical shapes are widely present in both man-made and natural environments, therefore the literature counts a significant number of specialised approaches to detect cylinders from 3D data and reliably estimate their parameters. In this section we briefly cover a number of state-of-the-art cylinder fitting and extraction techniques pertinent with the proposed method. Among these solutions we can distinguish two different approaches: the ones requiring oriented point clouds (i.e. each point is associated with a normal vector) and the ones which only uses 3D points locations. While surface normals significantly reduce the parameter search space and the problem complexity, their availability in acquired data is not always ensured and the reliability of computed normals strongly depends on the quality of the point cloud itself. We stress that methods which do not use point normals are naturally more stable, especially on noisy point clouds. As previously mentioned, we can distinguish between two conceptually different tasks, that are fitting and extraction.

The aim of fitting methods is to compute the best set of parameters for a cylindrical model (axis, location and radius) to suit the given point cloud. Usually this kind of task is performed after the acquired data has been segmented, so that we can associate a subset of 3D points with a geometrical primitive to be fitted. The parameter estimation is usually carried out through a linear or non-linear least-squares minimization of the points distance from the fitted surface [22]. Since these techniques assume a Gaussian distribution of noise, they exhibit high instability with respect to outliers. Consequently, the approach works if the segmentation is almost exact (so there is no clutter) or when the scene includes only a single cylindrical shape acquired with a good degree of accuracy. In [23] the authors propose a Maximum Likelihood Estimation assuming a Gaussian mixture distribution for errors, and they apply the method to a terrestrial laser scanning application. The algorithm proposed in [24] extracts a single cylinder from noisy and possibly incomplete data samples. It adopts robust Principal Component Analysis and regression to reliably compute the parameters of a single cylinder. The approach described in [25] proposes algebraic methods to efficiently compute cylinder parameters from a minimal set of points, in both oriented and non-oriented cases. In [26] a fitting method for laser scanner is proposed: an ellipse is fitted for each line acquisition, then cylindrical parameters are devised from the set of 3D ellipses. The method does not require point normals but it is designed to work on a scene containing a single cylinder in a specific industrial scenario, therefore this approach can not be generalised to other contexts.

Extraction algorithms are applied when capturing more complex scenes, including a number of primitives to be detected and other elements that must be automatically discarded. In the literature, the two major classes for extraction algorithms consist in RANSAC and Hough-based.

Thanks to their robustness, RANSAC-based methods [27,28] are widely employed for primitive extraction and fitting [29]. They directly exploit acquired data to exclude outliers and devise a good set of parameters. The work described in [30] splits the extraction task in two phases, both exploiting random sampling techniques. First, the set of points is filtered extracting a possible cylinder direction from the Gaussian image of the whole point cloud, then its size and location are extracted in the same way. Note that the first step of this method highly relies on point normals to extract the cylinder orientation. In [31] the authors propose a shape detection algorithm operating in unorganized point clouds where the point normals have been precomputed. They formulate a method which works on large point clouds, exploiting points spatial proximity to devise their belonging to a shape, and then apply a hierarchical sampling strategy. Despite the advantages offered by RANSAC-based approaches, their non-deterministic nature makes them impractical when dealing with large-scale point clouds. Moreover, data exhibiting severe noise levels could be problematic for shape detection also because the performances are highly influenced by point normal accuracy.

Another class of approaches for primitive extraction is Hough-based methods. As in the well-known Hough transform [32], the generalised method uses specifically designed voting spaces to extract the primitives in the scene. In the case of cylinders the five parameters needed translate in a 5-dimensional voting space, resulting infeasible. To overcome this problem, the approach in [33] proposes a sequential Hough transform divided into two steps in order to reduce the dimensionality of the parameter space. This method first exploits normals in a 2-dimensional accumulator to estimate the axis rotation, then it estimates radius and location in a 3D accumulator.

Some works present variations of Hough-based technique for some specific applications. For example, the authors in [18] propose a fallen tree detection for terrestrial laser scanning. In particular, they make some assumptions on the position of the cylinders and apply some filtering to have an initial prior over data, then they propose a cylinder extraction based on [33] and a final refinement to obtain single tree stems. Another popular application is pipe-run reconstruction: in [34] the authors propose a prior segmentation and an area-based adaptive Hough transform to reduce time and space complexity. Hough-based methods are

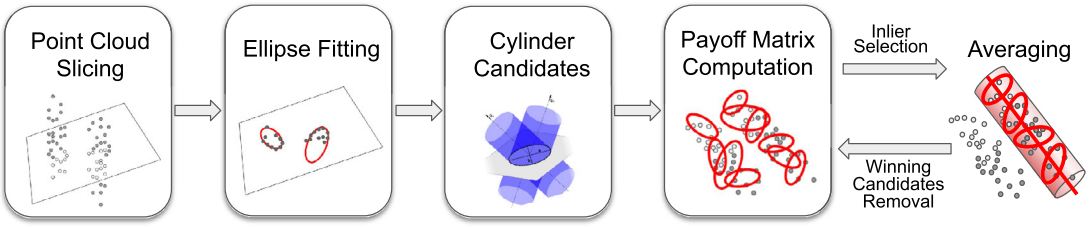

Fig. 1. Flowchart of the proposed method: first, the point cloud is sliced with virtual random planes, then all 2D ellipses are fitted: each of them produces two possible axis candidates for the cylinder. Finally, an inlier selection process based on replicator dynamic performs the primitive extraction. Multiple cylinders are extracteds by updating the payoff matrix and running replicator dynamics again.

usually time-consuming for large amount of data and the voting space discretization could lead to poor performances. Moreover, in absence of point normals the parameter space dimensionality makes the approach infeasible so surface normals are always required and their precision highly influences the final outcome.

In addition to these two methods, several solutions combine RANSAC, Hough and other approaches within a pipeline, or exploit some prior knowledge of the scene to get better results. The work in [35] proposes a pipeline that detects connected components and circles to validate and extract multiple cylindrical shapes. In [36] the authors propose a cylinder detection algorithm for robot grasping applications. They assume the objects lying on a planar surface and perform an initial image classification using a CNN, then they propose an extension of the two-step Hough transform employing a randomized sampling scheme and finally use surface curvature to filter the results. Another cylinder fitting for robot grasping is proposed in [37], where the authors propose a significant data preprocessing phase, followed by a RANSAC model fitting employing a custom validation and a Hough-based voting scheme. The method proposed in [38] is composed by several steps: first, they select points belonging to potential cylindrical surface by analysing the curvature values, then for each of them a fitting procedure is run starting from its neighbourhood. The fitting algorithm is applied several times so that at each iteration additional inlier points are appended to the current cylindrical model; moreover a validation procedure determines the reliability of the extracted cylinder. Finally the set of cylinders is obtained through mean shift clustering.

Many extraction algorithms are designed to solve specific tasks: a widely described application in the literature is pipe-run reconstruction from large-scale 3D scans of huge plants. In such scenarios cylinder extraction is often an initial task in a process finalised in recovering connections, curves and junctions of the pipes to reconstruct the whole structure of the scanned plant. These cases are usually more complex and solved with method comprising several heuristics and filtering steps. The method described in [39] does not need normals and employs a RANSAC-based approach and PCA to distinguish between straight and curves section of pipes, while the work presented in [40] proposes a normal-based region-growing approach to detect the position of different elements in piping systems. Most of the times these approaches are based on strong structural assumptions like a prior over the cylinder orientation and location. In [41] normals are not exploited but the authors assume to observe only horizontal or vertical pipes, so the cylinder orientations are significantly constrained. The approach proposed in [20] exploits an initial registration of the CAD model to the point cloud in order to have some prior information on the acquired data. Another common assumption consists in observing only objects lying on a planar surface, like in robot grasping [36,37].

The majority of described methods require normal or curvature pre-computation, which can be very sensitive to noise and outliers. Moreover, they need a specific tuning step for several thresholds and parameters involved in the process. An accurate parameter calibration is a time-consuming task, and could be an advantage if working with stable conditions, but could be a serious limitation when working with heterogeneous scenes and scanning devices.

The extraction algorithm we present in this paper is partially inspired by [26], and improves the concept to extract multiple cylinders from an unknown configuration with several sources of noise. Fig. 1 displays a schematic representation of the basic steps performed by the proposed algorithm. The method iteratively slices the scene with randomly generated virtual planes to compute a set of 2D ellipses, each of which generates two cylinder candidates. Then, a specifically designed cylinder similarity function is employed in a clustering process based on game theory, which ensures the selection of the subset with the best mutual consensus.

3. The proposed method

Our goal is to extract cylindrical shapes from non-oriented point clouds. We simplify the operation by considering only cylinders with an infinite extent along the axis, thus reducing their parametrization to a 3D line in space, called *axis*, and a radius. In practice, this is not a limiting assumption since (i) co-axial cylinders with different radii are rare in common scenes, and (ii) it is simple to estimate the cylinder height once the 3D points associated with its shape are identified in the cloud.

We parametrize a cylinder ζ with the triplet $(p, \vec{v}, r)$, in which $(p \in \mathbb{R}^3,\ \vec{v} \in \mathbb{R}^3)$ and $r \in \mathbb{R}$ describe its axis and radius respectively. We pose the additional constraints of $\|v\| = 1$ and $\langle p, \vec{v} \rangle = 0$, resulting in only 5 degrees of freedom out of the 7 parameters used for the parametrization.

Our method considers the cylinder extraction as a clustering problem. We suppose to have a set of *cylinder candidates* representing possible alternative cylindrical shapes observed in the scene. We designed a robust Game-theoretic process so that candidates coherent with a common cylindrical model are grouped together and weighted according to their mutual fitness. Therefore, by repeating this selection until all candidates are grouped, we robustly extract a sequence of cylinders ordered by their overall consistency with the relative candidate cluster. To support this principle, in Section 3.1 we formulate a distance function among pairs of candidates that properly accounts for the 3D displacement of their relative axes. Then, in Section 3.2 we propose a possible way to generate robust candidates and in 3.3 we discuss the details of the Game-theoretical process.

3.1. Cylinder distance function

Cylinder candidates are clustered together by maximizing a similarity among elements of the same cluster. The non-trivial aspect is that, a part from the relative radii, a well-defined function must deem the combined contribution of axes positions and orien-

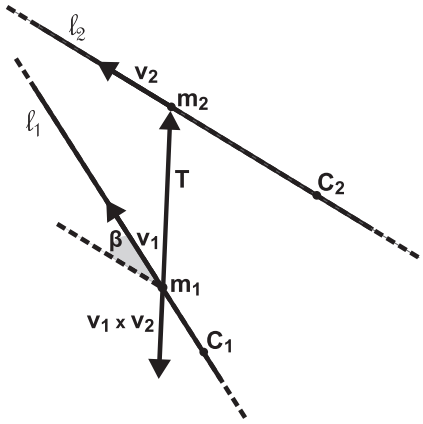

Fig. 2. The defined screw motion between lines l_1 and l_2.

tations. Therefore, defining a similarity between two cylinders boils down in defining a distance between 3D lines (ie. axes) in space.

Let $l_1 = (c_1, \vec{v}_1)$ and $l_2 = (c_2, \vec{v}_2)$ be two 3D lines defined respectively by a point c and a unitary-norm direction $\vec{v}$. Following [42], we construct a rigid motion $(\bar{\mathbf{R}}, \bar{T})$ mapping line l_1 to l_2 or, more formally, the roto-translation such that $\forall p \in l_1, \exists\, k \in \mathbb{R}.\ \bar{\mathbf{R}}p + \bar{T} = c_2 + k\vec{v}_2$. Since every rotation around $\vec{v}$ and every translation along $\vec{v}$ transform a line in itself, there exist infinitely many $(\bar{\mathbf{R}}, \bar{T})$ mapping l_1 in l_2. Among those, we can express the one exhibiting the shortest translation length and smaller rotation angle taking advantage of the kinematic notion of a screw. Chasles' theorem [43] states that every rigid motion can be described by means of a translation along a unique line (called screw axis) followed (or preceded) by a rotation about the same axis. The motion described in the theorem is called *screw* and, since its discovery, is one of the most convenient ways to describe spatial movements. In our setting, we can directly describe $(\bar{\mathbf{R}}, \bar{T})$ in terms of a screw motion, as sketched in Fig. 2. Let $m_1 = c_1 + t_1\vec{v}_1$ (for some scalar t_1) be the point lying on l_1 closest to the line l_2. Similarly, let $m_2 = c_2 + t_2\vec{v}_2$ be the point of l_2 closest to line l_1. By construction, the vector $T = m_2 - m_1$ is orthogonal to both l_1 and l_2 and its length is the minimum distance between all the points of the two lines. This implies that T has the same orientation (but possibly different magnitude and sense) of the vector $\vec{v}_\perp = \vec{v}_1 \times \vec{v}_2$. So, the rotation with angle $\beta = \arcsin(\|\vec{v}_\perp\|)$ around the axis defined by point m_1 and vector $\vec{s} = \vec{v}_\perp / \|\vec{v}_\perp\|$ will let l_1 be parallel to l_2. Since m_1 lies on the rotation axis, its position will not change after the transformation, hence the distance between the two lines will remain equal to $\|T\|$. Trivially, a final translation along T will let l_1 coincide with l_2. To summarize, the minimum screw motion between the two lines is the one with axis $(m_1, \vec{s})$, translation length $\|T\|$ and rotation angle β.

This modelling is convenient for our purposes because we can relate the parameters of a screw-motion with the scalar and vector part of a unitary dual quaternion $\hat{\mathbf{q}}$ (refer to Appendix A for more details about dual quaternions).

The distance is then defined as following

$$d(l_1, l_2) = \min(\|\log(\hat{\mathbf{q}})\|, \|\log(-\hat{\mathbf{q}})\|). \tag{1}$$

Given such axes distance, we can easily derive a similarity function between a couple of cylinders $\zeta_1 = (p_1, \vec{v}_1, r_1)$ and $\zeta_2 = (p_2, \vec{v}_2, r_2)$.

$$\pi(\zeta_1, \zeta_2) = \begin{cases} e^{-\frac{d\left((c_1,\vec{v}_1),(c_2,\vec{v}_2)\right)}{\lambda}} & \text{if} \quad Ad(\zeta_1, \zeta_2) < r_1 + r_2 \\ & \quad \wedge \ \frac{min(r_1,r_2)}{max(r_1,r_2)} > p \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $Ad(\zeta_1, \zeta_2) = \|T\|$ is the minimum distance between two axes $(c_1, \vec{v}_1)$ and $(c_2, \vec{v}_2)$ as defined previously, and $\lambda > 0$ is a free parameter[2]. The value $p \in [0, 1]$ is the minimum ratio between the radii r_1 and r_2 required for two primitives to be compatible. The similarity function takes values between 0 and 1 and is inversely proportional to the axis distance defined in (1), thanks to the negative exponential with factor λ. Note that a cylinder is fully compatible with itself (the ray distance is equal to zero, so the similarity will be one) and not compatible at all if the axes relative position and the radii are not consistent with each other.

We will employ such definition of similarity in a clustering process based on Game-Theoretical framework, which allows the extraction of the more coherent subset of primitives among all candidates.

3.2. Candidate extraction

There are several ways to obtain possible cylinder candidates from a 3D point cloud. The choice is influenced by the acquisition device, that could introduce some limitations due to its functioning and level of accuracy. For example, data could not be uniformly distributed in space and point normals may be available or not. In this approach we do not assume points to have a specific structure, neither to have normals. In this way the problem is kept as general as possible, and the method can be applied in a variety of practical applications.

Our candidate extraction is based on random slicing planes over the scene: we randomly generate N planes and compute the intersection between them and the point cloud. In this way we obtain N sets of 2-dimensional scattered points used to fit all possible ellipses, which in turn are used to compute the cylinder candidates.

To simplify the discussion, let's assume to have acquired a scene containing one single cylindrical shape and many other points resulting from other objects or noise. The intersection between a cylinder ζ and a plane P (whose normal $\vec{n}$ is not orthogonal with $\vec{v}$) results in an ellipse E with the following properties (see Fig. 4, Left):

- the minor semi-axis b is equal to the cylinder's radius r;
- the ellipse center c lies on the cylinder's axis;
- the ratio between the major semi-axis a and minor semi-axis b is a function of the angle α between $\vec{v}$ and $\vec{n}$, according to:

$$\cos\alpha = \frac{b}{a}. \tag{3}$$

If 3D data is dense enough, a subset S of points lying sufficiently close to the plane P can be extracted, generating a set of 2D points. Fig. 3 shows a simple example of this extraction process, starting with a point cloud and a slicing plane. According to what said before, some of these points will lie on the ellipse corresponding to the intersection between plane and cylinder, while some others will be distributed without any particular pattern (ie. they are outliers with respect to the elliptical shape). Therefore, any sufficiently robust fitting operation that extracts the elliptical shape E out of S will give clues on the unknown cylinder enclosed in the scene.

In particular, we already observed that the minor axis of E corresponds to the radius of the cylinder, while the center c defines the position of the axis. The ratio between semi-axes a and b will determine the axis direction $\vec{v}$ up to two equally-possible configurations (see Fig. 4, Right). Due to this intrinsic ambiguity, at least two plane slices are needed to fully recover a cylinder.

[2] In all our tests we fixed $\lambda = 5$.

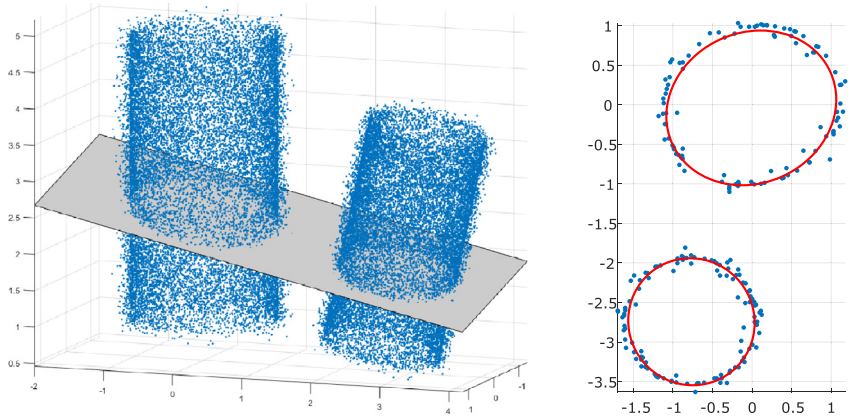

Fig. 3. Left: an example of point cloud containing two cylinders and a random slicing plane. Right: scattered points lying on the planar slice and the corresponding fitted ellipses. Each ellipse will be associated with two possible axes so this plane will generate four cylinder candidates.

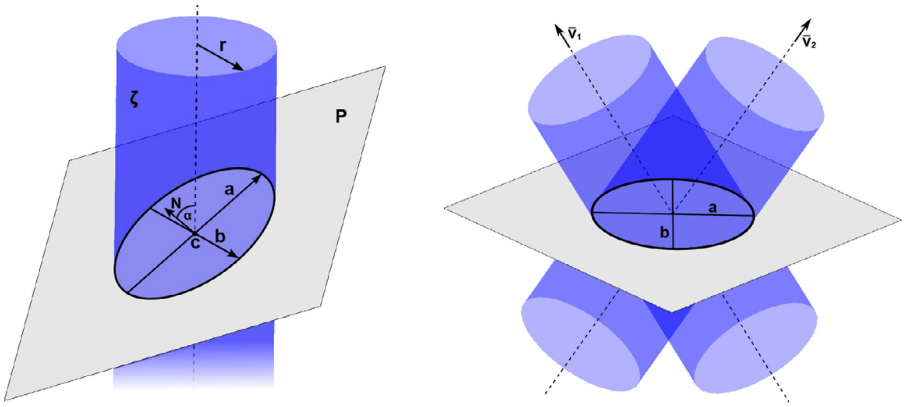

Fig. 4. Left: Intersection between a cylinder ζ and a plane P. Right: The two possible cylinders deriving from an ellipse in 3D space. They have radius and center point in common but differ in the axis direction (denoted as $\vec{v}_1$ and $\vec{v}_2$).

In [26] the authors propose to use multiple slices to collect only the centers $(c_1 \dots c_n)$ and the minor axes $(b_1 \dots b_n)$ of each fitted ellipses. With this method, the cylinder radius is recovered by averaging each $b_1 \dots b_n$ and the axis $(p, \vec{v})$ by fitting a 3D line to $c_1 \dots c_n$ in a least-squares sense. The approach has three drawbacks. First, it totally discards information given by each ellipse regarding the possible orientation of cylinder axis (due to the already mentioned ambiguity). Second, it can only handle a single cylinder in the data. Third, it can badly suffer from unlucky plane slices (like areas with few scene content), since all values are averaged without weighting their reliability. This aspect is partially relevant though, since the subsequent non-linear optimization may still be able to recover the correct shape.

Our approach follows a similar strategy: we slice the scene with n different random planes. From each 2D scattered point set obtained from the i-th slice, we robustly extract all the $j = 1 \dots N_i$ possible ellipses using RANSAC and the ellipse model estimation method described in [44]. When considered in 3D space, each extracted ellipse $E_{i,j}$ is defined by its major and minor semi-axes vectors $\vec{a}_{i,j} \in \mathbb{R}^3$ and $\vec{b}_{i,j} \in \mathbb{R}^3$ and its center $c_{i,j} \in \mathbb{R}^3$. By construction, $\vec{a}_{i,j}$ and $\vec{b}_{i,j}$ are both defined up to a sign and orthogonal to the normal $\vec{n}_i$ of the i-th plane.

From each $E_{i,j}$, we compute the two possible cylinder candidates as the tuples $(r, c, \vec{v} = \mathbf{R}\vec{n}_i)_{i,j}$ and $(r, c, \vec{v} = \mathbf{R}^T\vec{n}_i)_{i,j}$, where:

$$r = \| \vec{b}_{i,j} \|$$
$$c = c_{i,j}$$

$$\mathbf{R} = \mathbf{I} + \sqrt{1 - \frac{\| \vec{b}_{i,j} \|^2}{\| \vec{a}_{i,j} \|^2}} \left[\frac{\vec{b}_{i,j}}{\| \vec{b}_{i,j} \|} \right]_\times + \left(1 - \frac{\| \vec{b}_{i,j} \|}{\| \vec{a}_{i,j} \|} \right) \left[\frac{\vec{b}_{i,j}}{\| \vec{b}_{i,j} \|} \right]_\times^2 . \quad (4)$$

In the tuples, r and c denote the cylinder radius and axis point shared by the two candidates and $\mathbf{R}\vec{n}_i$, $\mathbf{R}^T\vec{n}_i$ are two possible axis orientations obtained by rotating the plane normal $\vec{n}_i$ around the minor-axis of the ellipse $E_{i,j}$ with an angle of $\pm \arccos \frac{\|\vec{b}_{i,j}\|}{\|\vec{a}_{i,j}\|}$[3].

Note that we can extract several ellipses from a single plane slice (each one originating two possible axes), but all cylinder candidates $\xi = (r, c, \vec{v})_{i,j}$ generated from slice i are not compatible by construction and hence should be assigned to different clusters. Indeed, it is simple to observe that distinct coplanar ellipses can only be generated by distinct cylinders and the two axes coming from the same ellipse are mutually exclusive.

Thanks to this property we can reformulate the similarity function, taking into account the plane which generated each candidate. Therefore, the new similarity between two candidates $\xi_1 =$

[3] Matrix $\mathbf{R}$ is obtained by the Rodrigues' formula. The symbol $[.]_\times$ denotes the skew-symmetric matrix of the cross-product.

$(r_1, c_1, \vec{v}_1^k)_{i_1,j_1}$ and $\xi_2 = (r_2, c_2, \vec{v}_2^h)_{i_2,j_2}$ is the following

$$\pi'(\xi_1, \xi_2) = \begin{cases} 1 & \text{if } \xi_1 = \xi_2 \\ \pi(\xi_1, \xi_2) & \text{if } i_1 \neq i_2 \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where $\pi(\xi_1, \xi_2)$ is the cylinder similarity function as defined in (2). The additional condition is added in order to explicitly exclude primitives generated from coplanar ellipses: in this way all pairs of candidates coming from the same slicing plane will have zero similarity. This formulation is particularly convenient in the clustering approach explained in the next section since the evolutionary process used to extract sets of coherent observations will inherently exclude incompatible candidates from the final result.

3.3. Inlier selection process

In the proposed approach the extraction of a subset of candidates containing elements with a high mutual support is carried out in the context of Evolutionary Game Theory [45]. We consider a scenario in which we have an initial population of candidates, where each distinct couple of individuals play a game one versus the other on the basis of pre-programmed strategies. Then, a selection process brings some elements which match well together to thrive, while driving the "unfitting" ones to extinction.

In our specific case, we consider all the $M = \sum_{i=1}^{n} N_i$ cylinder candidates $\xi_1 \dots \xi_M$ as the individuals, that we call *hypotheses*. Rather than the singular characteristics of each of them, we are interested in expressing how well two hypotheses play together. In other words, we need to define their behaviour when they are put together in the final population. We define such fitness through a *payoff-function*, which in our case is formulated as in (5). This function can be conveniently represented as an $M \times M$ symmetric matrix Π in which an element at row i and column j is defined as follows:

$$\Pi(i, j) = \pi'(\xi_i, \xi_j). \quad (6)$$

We then consider a discrete probability distribution $\mathbf{x} = (x_1, x_2, \dots, x_M)$ over the set of hypotheses (the entire population). The problem of finding a coherent set of candidates can be defined formally as finding a particular distribution in the standard simplex $\mathbf{x} \in \Delta^M$ maximizing the average payoff $\mathbf{x}^T \Pi \mathbf{x}$.

One possible solution to the problem employs an evolutionary process starting from a uniform distribution $\mathbf{x}^{(0)}$ and making it evolve through the well-known *discrete-time replicator dynamic* [46]

$$x_i^{(t+1)} = x_i^{(t)} \frac{(\Pi \mathbf{x}^{(t)})_i}{\mathbf{x}^{(t)^T} \Pi \mathbf{x}^{(t)}}. \quad (7)$$

Eq. (7) belongs to a class of evolutionary dynamics called *Payoff-Monotonic Dynamics* that are guaranteed to converge to a solution $\hat{\mathbf{x}}$ in which the set of hypotheses with associated probability greater than 0 (namely the *support* of a population) cannot include any strategy with mutual payoff equal to 0. In other words, the candidates with associated non-zero probability in $\hat{\mathbf{x}}$ will only be the ones generated from different planes and with the distance between the two axes greater than the sum of the two radii. As a result, only the most consistent cylindrical model will emerge from the whole set of candidates after the inlier selection process.

3.4. Averaging the winning candidates

At convergence, a probability distribution over the candidates $\xi_1 \dots \xi_M$ is obtained, but we are still left with the problem of estimating the resulting cylinder parameters out of it. If we just consider the support of the population, we might use the same approach of [26] to just fit the cylinder axis line to the centers of the winning candidates. However, we now have the chance to produce a better estimation by weighting the contribution of each candidate considering the probability distribution $\hat{\mathbf{x}}$.

The estimation of the final cylinder $\zeta = (p, \vec{v}, r)$ is divided in two parts. The radius r is estimated as the weighted average of the candidate radii (8)

$$r = \sum_{i=1}^{M} \hat{x}_i r_i \quad (8)$$

where r_i is the radius of ξ_i. Then, the axis $(p, \vec{v})$ is averaged using the same approach of [42], posing the problem of 3D lines interpolation in terms of rigid motions blending. The general approach is as following:

1. An initial axis estimate $(p', \vec{v}')$ is computed by linearly interpolating the axis point p' and direction $\vec{v}'$ independently, according to the weights given in $\hat{\mathbf{x}}$
2. All the screw motions between $(p', \vec{v}')$ and the cylinder candidates $\xi_1 \dots \xi_M$ are computed according to (A.2)
3. The screw motions are averaged using the *Dual-quatertions Iterative Blending* [47] to obtain an average motion $\mathbf{q}^*$
4. $\mathbf{q}^*$ is applied to $(p', \vec{v}')$ to get the final cylinder axis.

3.5. Extracting multiple cylinders

The process described so far can extract only one cylinder at a time. To extract multiple cylinders, the two operations are simply repeated by ensuring that the candidates resulting from one run of the game theoretical process cannot be selected anymore. Specifically, after the extraction of each cylinder, we manually modify the payoff matrix Π according to the current population support.

Let $k_1, k_2, \dots, k_h$ be the set of indices corresponding to the winning strategies. This means that they are non-zero elements in the final distribution $\hat{\mathbf{x}}$ (ie. $\hat{x}_K \neq 0 \ \forall K \in k_1 \dots k_h$). The payoff matrix Π is then modified so that all rows and columns at indices $k_1 \dots k_h$ are set to zero (they are considered fully incompatible with any other candidate in the population). After that, the initial population $\mathbf{x}^0$ is reset to the uniform distribution and the inlier selection process (7) is run again until convergence. Since we are using a payoff-monotonic dynamic, all the previous candidates indexed by $k_1 \dots k_h$ cannot be present again in the next set of winning candidates. The whole process is repeated until all cylinders are extracted from the scene.

4. Experimental evaluation

In this section we present a set of experiments to assess the correctness and robustness of the proposed technique in several situations. Since this approach is designed to work with data coming from real scans, we tested it against both synthetic and real-world data. The synthetic setup allows us to estimate the behaviour of our cylinder fitting algorithm while adjusting some external factors such as noise levels, outlier quantity and occlusions. Moreover, we used such controlled environment to compare the proposed technique with other fitting approaches. Finally, at the end of this section, we use real-world data to test the extraction ability in some real-world situations.

Synthetic generated data consist in a point cloud lying on a unitary-radius cylinder, centred at the origin and with its axis parallel to the z-axis. Such points are perturbed with two different types of noise: zero-mean Gaussian additive noise and salt and pepper noise. The former is controlled by its standard deviation σ, while the latter is regulated by p, that is the probability of a point to be an outlier, with σ_o (such that $\sigma_o > \sigma$) being the outliers' standard deviation. We also introduced an additional parameter θ, denoting the angle underlying the portion of observed surface

(when $\theta = 360^\circ$ the cylinder surface is complete). This parameter is used to simulate a range-map acquisition, since auto-occlusions usually limit the reconstruction to a portion of the cylinder surface. We measured the algorithm precision in terms of three quantities, that are: radius relative error (in percentage), the angle and relative distance between the fitted axis and the ground truth.

4.1. Sensitivity analysis

One of the strengths of our approach is that it only depends on the number of slicing planes used to generate axis candidates. In fact, other possible parameters (like RANSAC thresholds for ellipse fitting) depend only on the scale and on the density of data and are fixed at the beginning of the process. The number of random planes has a direct impact on the algorithm's performances, so in the first group of experiments we evaluated our fitting approach varying the nature of the slicing planes against different kinds of data. In particular, we are interested in analysing the effects of two factors: the number of planes involved and their inclination with respect to the actual cylinder's axis.

In the first experiment we generated N planes with random inclination and position, and run our algorithm on the same randomly-perturbed data increasing the value N from 10 to 90. We used synthetically generated data as described before, testing several coverage angles θ and simulating different outlier conditions. In Fig. 5 the error values are plotted against the increasing number of slicing planes (on x-axis, from 10 to 90), each curve denotes a different surface coverage angle. We measured the axis angle error (in degrees, first row), the axis translation error (second row) and the radius relative error (third row). The columns show two setups with different amounts of inliers, while keeping the same level of additive Gaussian noise. Each configuration was tested 100 times and the error bars display the mean standard error.

As expected, a higher coverage angle implies a better ellipse fitting, producing a set of more reliable axis candidates. For this reason, in the few outliers case, the fitting precision increases for angles greater than 90° and become essentially the same for angles larger than 135°, where the curves overlap. Adding more outliers (2nd column) basically produces the same behaviour: for greater values of θ we observe a very good cylinder fitting, that is comparable to the low-noise scenario. For smaller angles (50° and 90°) the fitting can be improved by adding more planes, but still the axis inclination error results the most critical parameter to resolve. Overall, we observe a slight improvement when we slice the scene with a higher number of planes, but the choice of this parameter is not so critical in terms of fitting precision. The number of planes affects the robustness of the proposed algorithm, and thus should be adjusted depending on the features of the acquired scene. In the case of low or medium amount of outliers we can safely set the parameter N from 30 to 60 to obtain acceptable outcomes. The results obtained on partially occluded surfaces are a good indication that, with a suitable number of planes, the algorithm offers good results with typical scanner range data. Of course, this parameter depends on the kind of application: if the point cloud contains several cylindrical primitives the number of planes should be kept relatively high in order to capture all the elements to extract.

In the previous experiments we generated several planes with a random center and normal in order to test the general case in which locations and orientations of the primitives to be extracted are unknown. Nevertheless, there are scenarios in which an approximate orientation of the objects to be detected is known. In such cases, it makes sense to exploit such prior information to extract better candidates.

For this reason we analysed the performances of our algorithm varying planes inclinations with respect to the cylinder axis. We fixed the number of slicing planes to 20, and $\sigma_o = 0.05$, with $p = .1$; then we performed two sets of experiments, for low ($\sigma = 0.001$) and very high ($\sigma = 0.1$) levels of additive Gaussian noise. After generating a randomly perturbed cylinder, we run the fitting algorithm using planes with a random orientation which have been restricted to a small interval (from 0° to 10°, from 10° to 20° and so on). The experiment was repeated 100 times for each configuration. In Fig. 6 we show the three measured errors (one for each row) as boxplots for each different plane orientation interval. The first column displays results in the low noise setup, while the second column the results for very high noisy conditions. Note that the angle values of intervals refer to the relative angle between the plane normal vector and the cylinder's axis, so a small angle implies the plane to be almost orthogonal to the cylinder's axis. In practical terms, a larger plane angle stretches out the ellipses to be fitted, increasing their eccentricity, while an angle close to zero implies more circular sections. Results show that in both cases (low and high noise) a larger angle improves the cylinder fitting precision, especially in terms of axis localization (rotation and translation). This happens because the ellipse fitting becomes less ambiguous in the case of higher eccentricity of sampled data, therefore the computation of major and minor axes is more accurate. In fact, if we sample noisy points from a circle, a lot of ellipses with small eccentricity are equally acceptable results, but in practice they make the axis fitting more unstable. Conversely, the cylinder's radius is less affected by the planes inclination, but still we can observe small improvements for higher angles. This observation should be considered especially when extracting cylinders from a scene where some prior on their position is known and a high precision is required, even with noisy point clouds.

4.2. Comparisons

In the following set of experiments we focused on the performances of the proposed algorithm when compared to other cylinder fitting techniques for different noise conditions. We generated the usual synthetic scene containing a unitary-radius cylinder to be fitted, and we applied our method *GT* against *MSAC* [28], *Line fit* [26], *Tran* et al. [38] and *Jin* et al. [39].

The *MSAC* approach consists in a sample consensus technique requiring point normals to estimate the best cylinder fitting from the given point cloud. If vertex normals are not present, they are estimated by locally fitting a plane in each point's neighbourhood. We run the algorithm setting the maximum number of iterations equal to 5000; moreover, after some preliminary tests, we set the maximum inlier distance to 0.03 (which is 3% of the radius), so that MSAC approach works well with our data. *Line fit* does not need point normals and exploits the fitted ellipses as cylinder sections: the cylinder's axis is computed fitting a line through the centres and the final radius is the average of all radii values. After that, the cylinder is refined by fitting through a non-linear optimization. To have fair comparisons, we used the same extracted ellipses as input to both our and the described techniques. *Tran* et al. method proposes a cylinder fitting algorithm using a growing approach. In each iteration it exploits points normals to estimate the axis direction, then fits a circle to the points projected on the orthogonal plane to devise radius and axis location. After this process, the collected cylindrical shapes are validated and clustered via mean shift algorithm. In all the experiments we fixed the algorithm parameters as suggested in the original paper, namely: $\alpha = 50$, $\beta = 0.95$, with 10 iterations. Point normals were computed as described in [48], choosing 50 neighbours. The method proposed by Jin et al. [39] requires no normals for cylinder estimation: it starts from small neighbourhoods of points to fit a sphere in each group, then all fitted spheres are used to devise axis and radius of the detected cylinder. The work in [39] is specifically designed to detect pipes in 3D scans of large-scale plants, and

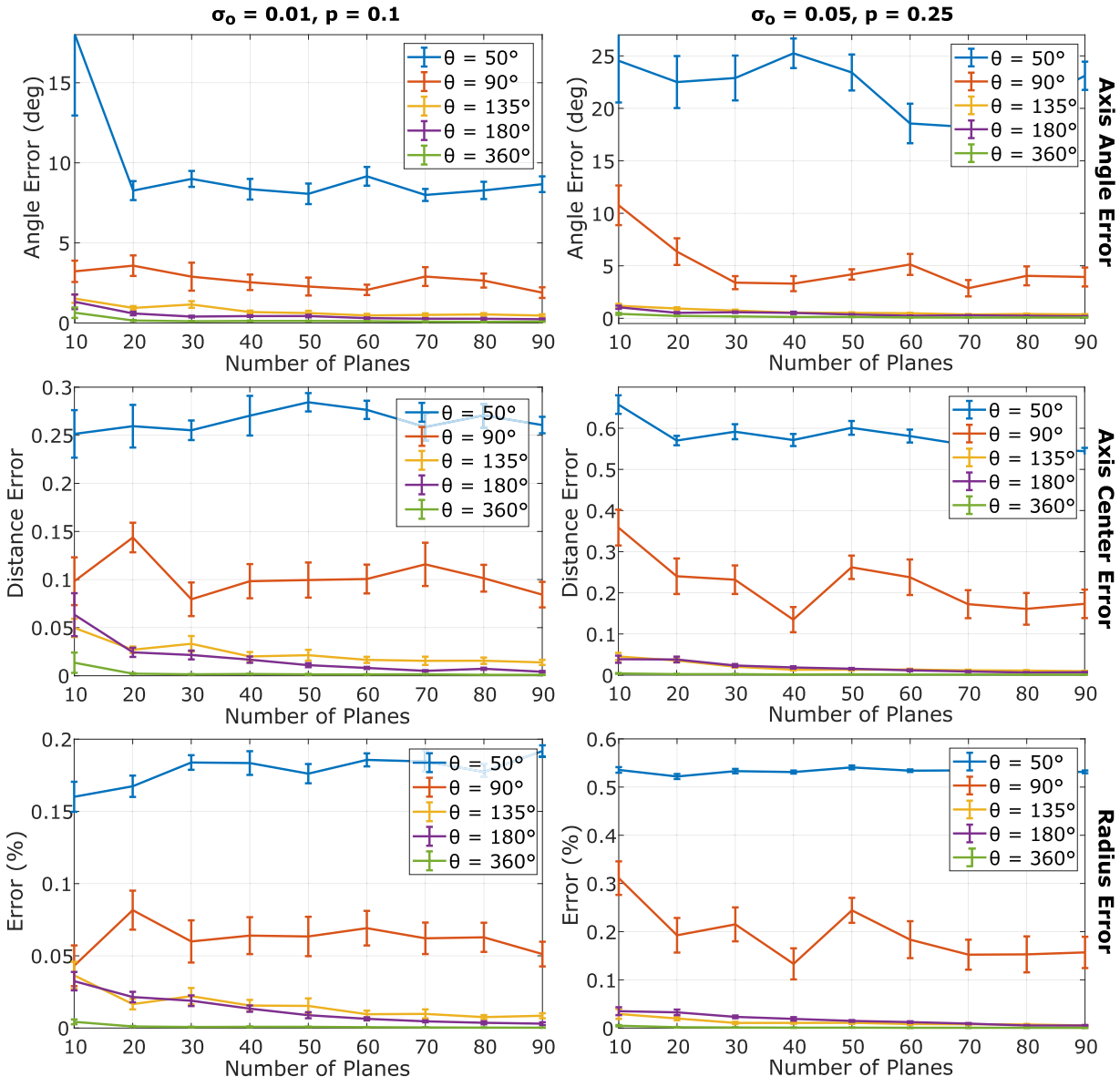

Fig. 5. Angle, center and radius errors varying the number of randomly generated planes. Each curve corresponds to a different surface coverage (the angle θ). Results are shown for low (1st column) and high (2nd column) outlier presence.

includes additional methods to compute the cylinder length and recognise curved regions. Despite the target application is different from ours, we chose to compare this technique because it is one of the few methods not requiring point normals and that can be adapted in a cylinder extraction context, ignoring the pipe routing aspect.

In the first test we gradually increased the standard deviation of additive Gaussian noise (from 0.001 to 0.05) and run all the fitting algorithms on perturbed data using 20 slicing planes. The outlier level was set to a medium level of $\sigma_o = 0.04$ and $p = .3$. Fig. 7 shows in different curves the average of error values for each fitting technique, increasing the level of noise (on x-axis). As usual, the bars denote standard error values and each experiment has been repeated 100 times to compute the average. Each plot displays a different measured error, namely: (from left to right) axis angle, centre displacement and percentage of radius error. Note that a logarithmic scale has been applied to y-axis to improve the visualization. In the case of axis angle error, *GT* offers more stable results and a better axis orientation precision with respect to other approaches. In particular, our method and *line fit* exhibit a similar behaviour, but line fit is characterized with slightly higher errors and variability. *Tran* et al. shows significant instability and poor results in most of the cases, especially in the axis parameter estimation. Its performances for angle and radius errors are close to our technique for σ larger than 0.03. *MSAC* algorithm is unstable in all the scenarios and in general it exhibits higher errors. The errors observed for Jin et al. method show invariance with respect to increasing Gaussian noise, but its overall precision can be compared with MSAC (for angle and radius) and line fit (for the axis center). In general, our method exhibits more accurate results in the case of Gaussian additive noise.

In the following set of experiments we compared the fitting performances for different outlier cases. Fig. 8 shows the behaviour of the five methods increasing the outliers frequency (left column) and magnitude (right column). We fixed Gaussian noise to $\sigma = 0.001$ and 20 random slicing planes to measure the different fitting errors as in the previous test (from top to bottom: axis angle, axis distance and radius error). We also changed *Tran* et al.

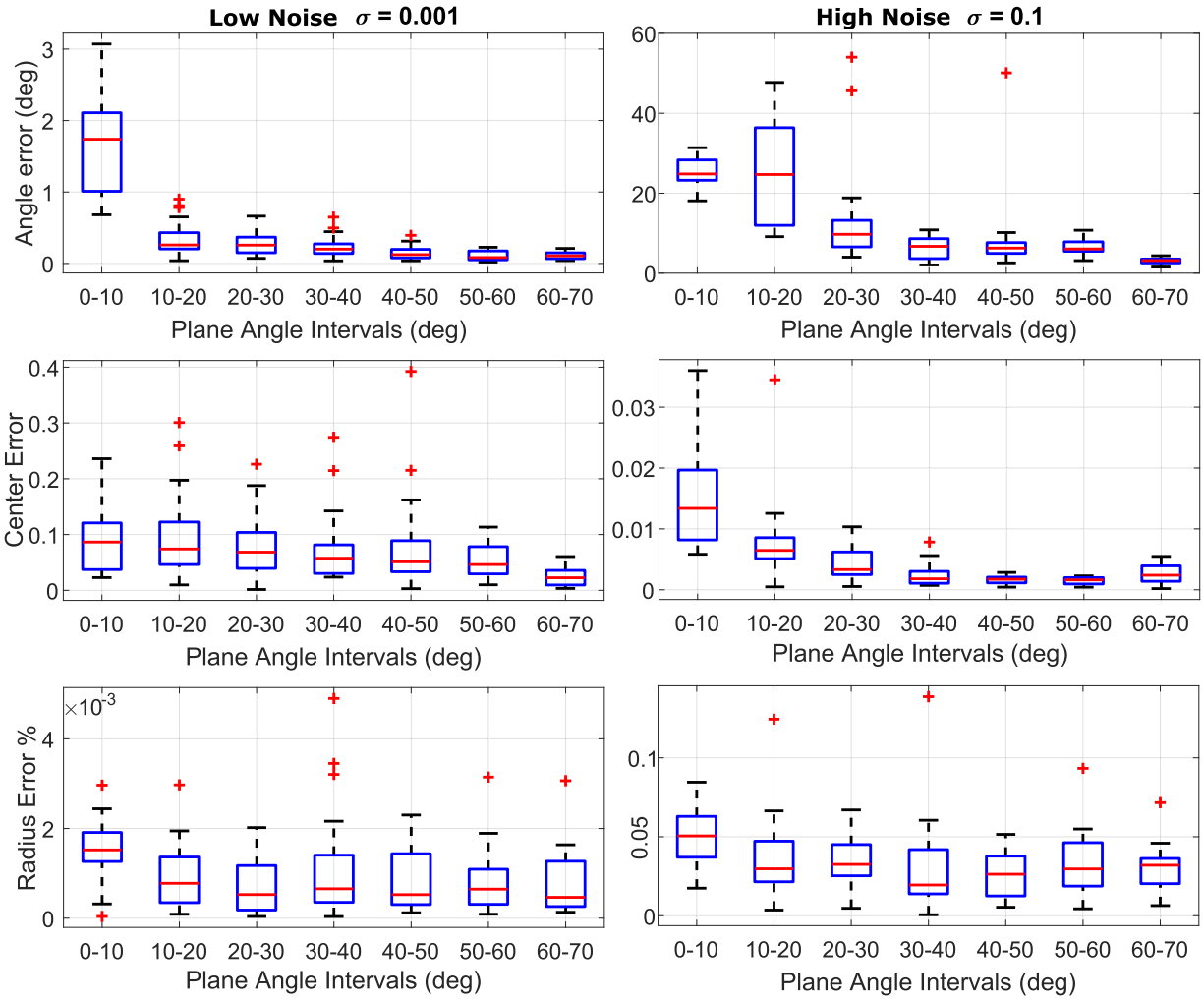

Fig. 6. Fitting performances using planes at different inclinations, in the case of low or high noise levels (respectively 1st and 2nd column). The error is splitted into axis angle, center displacement and radius error. Each boxplot represents an orientation interval in terms of angle between the cylinder axis and plane normals.

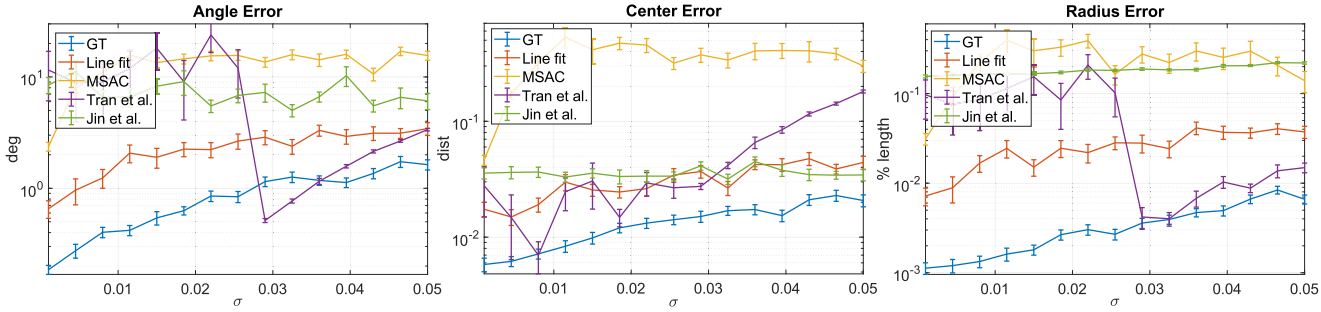

Fig. 7. Angle, center and radius error means (displayed in logarithmic scale) increasing the standard deviation of additive, zero-mean Gaussian noise. We tested five different cylinder fitting techniques, each one denoted by a curve.

parameters to adapt to the noisy acquisition: we used 100 points for normal computation and 200 neighbourhood points at the beginning of the fitting algorithm. Data were perturbed 100 times for each configuration to compute the mean and standard errors, shown as error bars. The leftmost column displays the results with a fixed $\sigma_o = 0.1$, increasing the value of p (ie. the probability of a point to be an outlier). In the rightmost plots (shown in logarithmic scale) we fixed the outlier probability $p = .4$ and gradually increased the outliers standard deviation σ_o from 0.05 to 0.2. In both cases *GT* exhibits more accurate and stable results with respect to other techniques, even with a severe outlier component. Both the techniques requiring point normals (*MSAC* and *Tran* et al.) display unstable results characterized by higher error values caused by wrong normal estimations introduced by outlier data. In particular, *Tran* et al. and *MSAC* have large variability and error when compared to *GT* and *line fit*. The technique proposed by Jin et al. is not altered by outlier changes but exhibits higher errors, especially in radius estimation.

We also simulated a typical scenario with a partially occluded object. Indeed, this situation is common in many applications since most of the times the acquired data is a range-map or a laser scan line. We run the algorithms increasing the cylinder's surface cov-

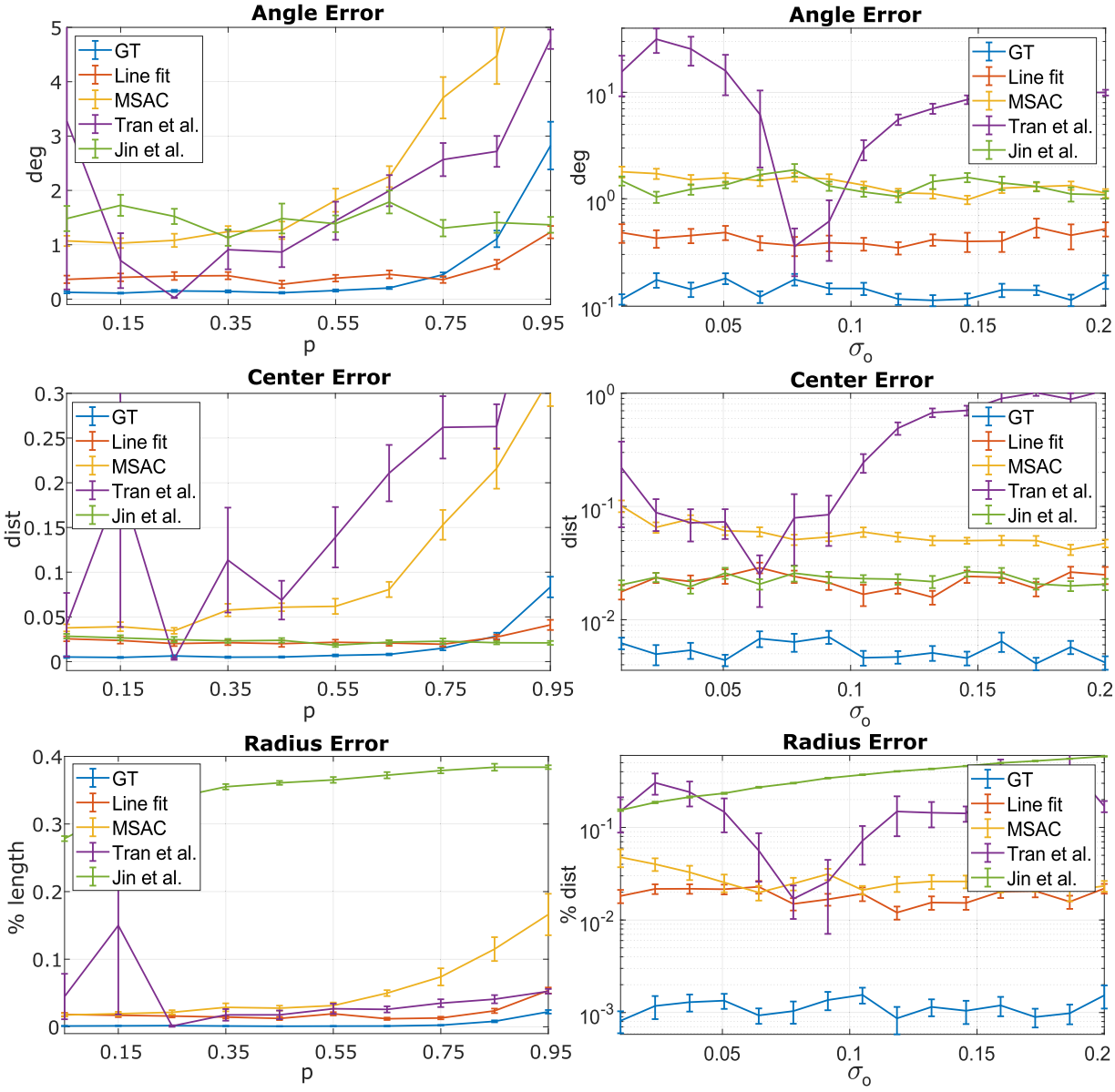

Fig. 8. Angle, center and radius error means varying the salt and pepper noise parameters. In the first column we increased the outlier probability p and keep $\sigma_o = 0.1$. In the second column (plotted with log scale) the outliers standard deviation σ_o was gradually increased, keeping $p = .4$. We tested five different cylinder fitting techniques, each one denoted by a curve.

erage angle θ, plotting the resulting errors in Fig. 9. Note that we displayed only our technique and the line fitting approach because *MSAC, Tran* et al. and *Jin* et al. errors were so high to hinder a good visualization of the first two. The additive Gaussian noise standard deviation was set to $\sigma = 0.005$, while the salt and pepper noise was characterized by a standard deviation $\sigma_o = 0.01$ and an outlier probability $p = .1$. We also increased the number of slicing planes (40, with any random inclination) to have more stable results. As we already noticed, the cylinder's axis angle is the most critical parameter to be computed: the leftmost plot of Fig. 9 shows that the proposed algorithm offers a better axis estimation with respect to the simple line fitting technique. In terms of axis dislocation and radius error, the proposed method and line fitting offer similar results.

Finally, we tested the execution time of the different techniques against data changes. We generated the usual synthetic scene and increased both additive Gaussian noise and outlier ratio: Fig. 10 displays the results. The leftmost plot shows execution time increasing the standard deviation σ with a fixed outlier amount ($p = .1$, $\sigma_o = 0.05$), while in the rightmost plot we increased the outlier probability p and kept fixed $\sigma = 0.01$ and $\sigma_o = 0.05$. Note that we did not compare the execution time varying the cylinder coverage angle θ since the other methods are not viable in such scenarios, as previously discussed (see Fig. 9 description). Execution times of our technique and line fit only differ in the game-theoretical inlier selection, since line fit approach exploits the same fitted ellipses to get cylinder parameters. Other approaches employ a constant amount of time, depending on the number of input data points.

In conclusion, we observed that our method gives more stable results and lower error in terms of both axis and radius estimation. The accuracy is similar to *line fit* for some cases, becoming more significant in presence of a large amount of outliers where the performance is improved. We also observed that the axis angle is the most critical element to be estimated, and our approach allows for the best candidates selection to have an accurate estimation and filter out outlier data.

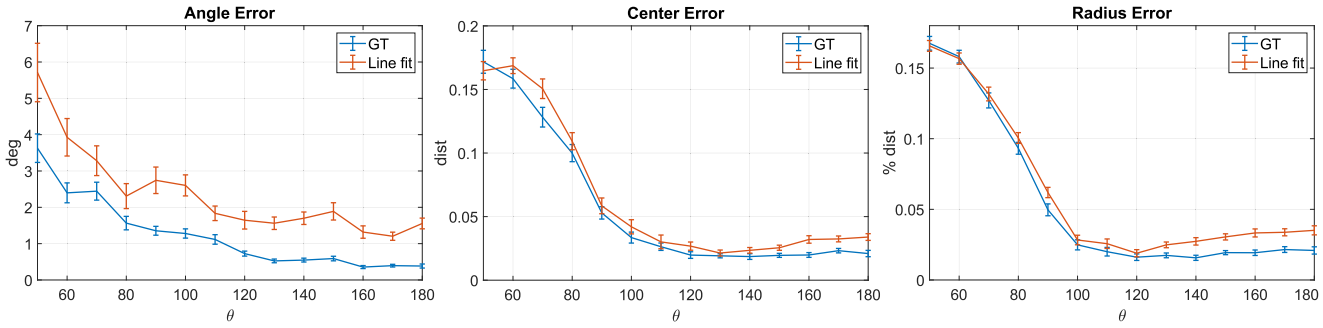

Fig. 9. Angle, center and radius error means increasing the cylinder coverage angle θ. Note that the other fitting techniques were excluded from the plot because of their high values.

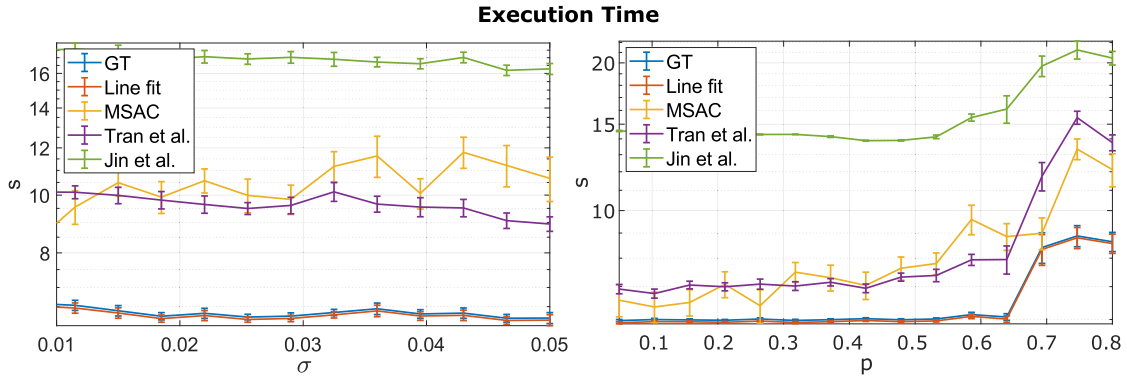

Fig. 10. Execution time of different fitting methods increasing Gaussian noise and outlier probability.

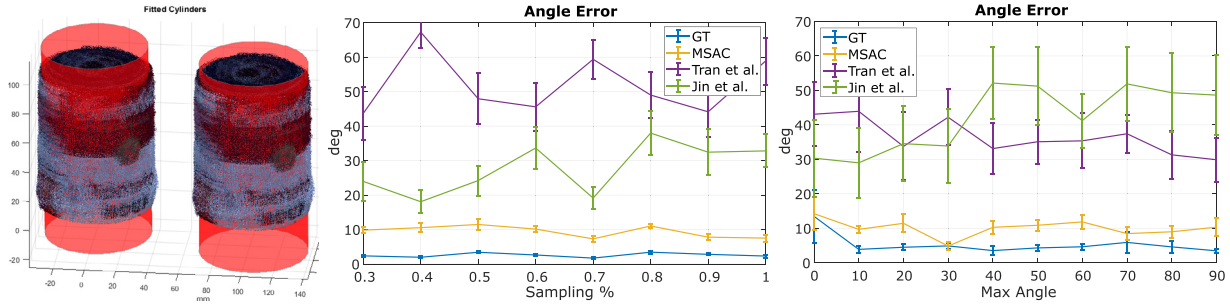

Fig. 11. Left: an example of the generated scene with two scans of tomato soup object. Red cylinders have been fitted by our method. Center and right: relative angle means of the two fitted axes from the scene. Bars represent the standard error.

4.3. Real-world experiments

In this experimental section we tested the multiple cylinder extraction capability of our algorithm employing real scan data to evaluate the performances in real-world scenarios. We adopted a well-known model dataset containing everyday items, that is the Yale-CMU-Berkeley Object and Model Set [49]; specifically we used 3d scan data presented in [50]. Such dataset includes scans of the objects from the Model Set, each one consisting in coloured 3d vertices with no orientation. We selected a subset of cylindrical items and composed them in several scenes from which some cylindrical primitives are to be extracted.

In the first experiment we analysed the repeatability and accuracy of our method when applied to real-world point clouds. We compared the proposed algorithm against the other techniques that we already described in the previous section. The line fitting method [12] was excluded because it is designed to work with a single cylindrical shape to be fitted, therefore its application in such scenario has no sense. Since MSAC and Tran et al. approaches require vertex normals, we computed them for each point by locally fitting a plane on its 80 neighbours[4] To extract multiple cylinders with MSAC approach, we iteratively removed all points falling inside the region delimited by $\pm$ 5% of the estimated radius. Note that, while MSAC approach needs to run each time the input is modified, our algorithm performs the ellipses fitting just once at the beginning, then the extraction is performed by dynamically run the game-theoretical selection with a modified payoff matrix (as explained in 3.5).

We choose one single object (*"tomato soup"*), with radius equal to 33 mm and height equal to 101 mm. The point cloud contains 373,205 vertices; it has no outlier points but the surface is quite thick, probably due to scanner rangemap registration errors. We generated a simple scene containing two instances of the object, separated with a known random translation. In Fig. 11 an example of such scene is displayed, together with the cylinders extracted by our algorithm (in red). Since the exact axis of the scanned object is not known, we evaluated the accuracy of the algorithms mea-

[4] We empirically determined that this value gives good results without being too slow in processing.

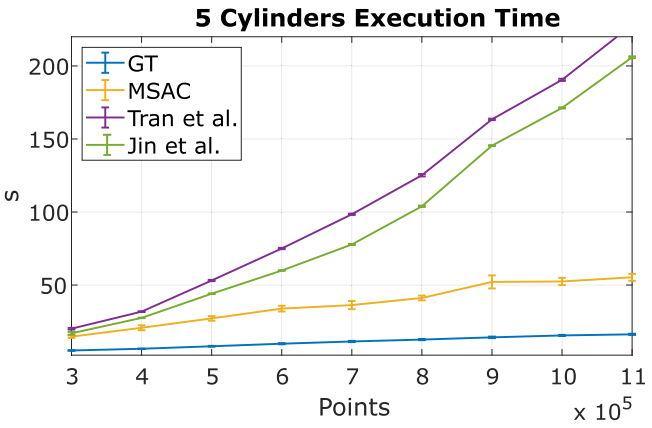

Fig. 12. Execution time for our method, MSAC and Jin et al. methods when extracting five cylinders (real-scanned data) from a point cloud with an increasing number of points.

suring the relative angle between the two estimated axes from the scene. Ideally, since the object was transformed with a pure translation, such angle should be almost zero. Plots in Fig. 11 display the relative angle values between the couple of extracted primitives for each technique. In the leftmost plot we simulated various point cloud densities (from 30% to 100%) starting from the original scan; we plot the relative angle value against the percentage of randomly sampled points (in x-axis). In this case we used 40 slicing planes and we performed 4000 MSAC iterations, with minimum inlier distance equal to 2mm. The experiment was repeated 100 times and the same uniformly sampled data was used as input for both methods. In the rightmost plot of Fig. 11 we simulated a partial scene acquisition (as in a range-map), varying the minimum angle between a fixed point of view and vertices normals. For this tests we removed all the points for which the relative angle between the normal and a fixed direction vector is below an increasing threshold (displayed on x-axis, note that min angle = 0 is the whole point cloud) and computed the relative angle between the two axes. In this setup we used 50 planes and increased MSAC iterations up to 5000. The view vector was $(0, 1, 0)^T$ and each test was performed 20 times with a random subsampling equal to 0.8. For both experiments we observed that our algorithm gave a more stable angle value ranging between 2 and 5 degrees. Moreover, the proposed approach is not influenced by variations in data density. In general the other approaches exhibited a more unstable behaviour and a greater standard error.

In the following experiment we compared the extraction speed of several approaches in a real-case scenario. We generated a scene including five cylinders (instances of *"tomato soup"*) and additional objects to be ignored by the extraction algorithm. Then, we randomly subsampled each scene to a fixed number of points (from 300K to 1000K). Fig. 12 displays the execution times for our approach, *MSAC, Tran* et al. and *Jin* et al., where x-axis shows the total number of data points. Each test have been repeated 100 times to obtain different samplings. For a fair comparison, we used 20 slicing planes and chose specific plane angles to always intersect all the cylinders in the scene. This is the worst scenario for our method since the size of the similarity matrix depends on the number of ellipses extracted in each slicing plane and not the number of points. Thus, the execution time of our algorithm strongly depends on the number of cylinders to be extracted, since it basically increases the number of RANSAC executions to estimate multiple ellipses. Consequently, as long as the average inlier/outlier ratio is kept constant in the scene, the amount of processed points has a very weak influence on the performances of our technique (indeed *GT* time curve is almost constant). On the other hand, *MSAC* and other approaches relying on normals exhibit longer execution times as the number of points increases. The approaches proposed by Tran et al. and Jin et al. depend on the number of points, so their execution time grows linearly with the point cloud size.

Finally, we performed some qualitative tests in various conditions to show the output of our algorithm in a general cluttered scene. Fig. 13 displays a selection of generated point clouds with the extracted cylindrical shapes. We composed the leftmost scene with objects of different radius and size: *"tomato soup"* (radius 33 mm), *"Master Chef"* (radius 51 mm) and *"c cup"* (radius 32.5 mm). We used 50 random planes to extract the three cylinders and displayed the computed radii. Other configurations include objects with random rotations and a scene with several cylindrical objects to be extracted (Fig. 13, centre and right respectively). In particular, for the latter we employed 60 planes and run 9 times the game-theoretical inlier selection.

We also generated some scenes adding non-cylindrical, generic objects to be excluded by the extraction algorithm. Fig. 14 shows two examples of such scenes. We employed the cylindrical objects *"tomato soup"*, *"Master Chef"* and *"Chips Can"* and other everyday objects of various shape and size. The displayed point clouds count in total 6,132,681 (left) and 5,283,187 (right) 3D points. We sliced both scenes with 60 planes (of all possible random inclinations and locations) to extract the cylindrical primitives. The displayed cylinders (in red and blue) are direct outputs of the proposed algorithm, with no further refinement. The primary objective of this experiment was to test the ability of our method in detecting cylindrical objects in the presence of clutter. Indeed, such scenario is realistic in a number of robotic grasping applications, where the need of identifying a target object is fundamental. Experimental results obtained with these real-world data show the effectiveness of our cylinder extraction method in different cluster conditions, effectively excluding other elements in the input point cloud.

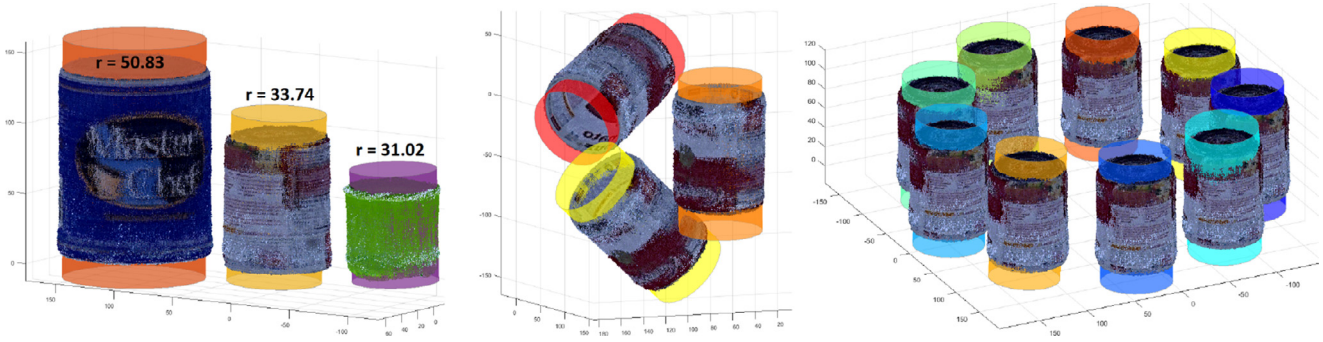

Fig. 13. Qualitative examples obtained in different kinds of scenes, generated using real scans data.

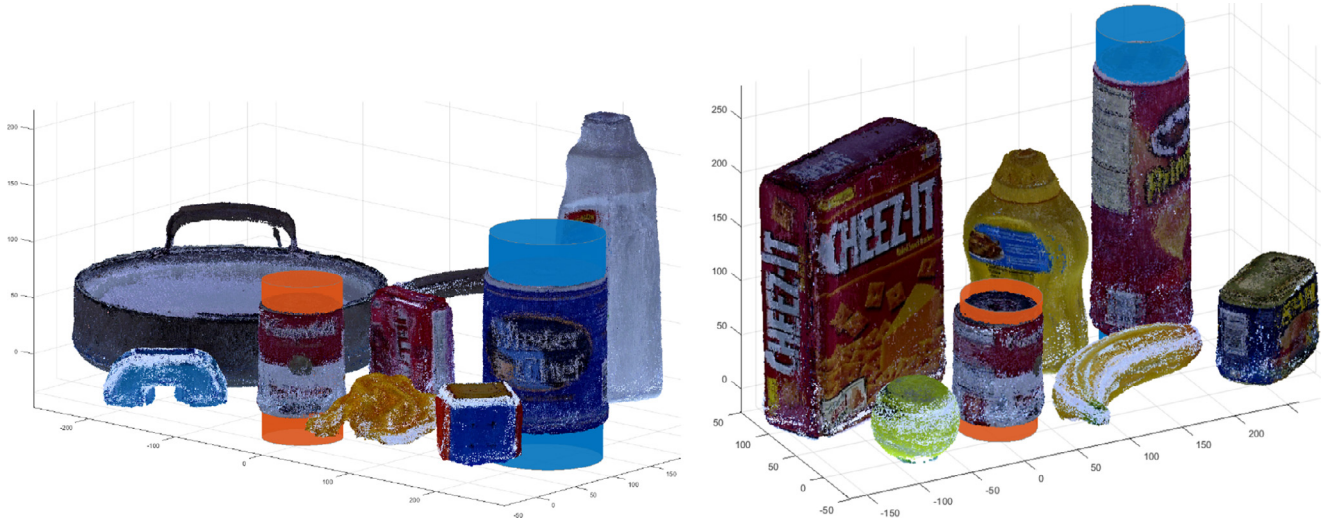

Fig. 14. Cylindrical primitives extracted "in the wild" from two difference scenes.

5. Conclusions

In this paper we proposed a novel technique to extract multiple cylindrical primitives from generic, non-oriented point clouds. The method combines two crucial elements: the definition of an effective similarity function between couples of candidate cylinders based on dual quaternion algebra, and a candidate extraction based on geometrical properties of the scene. The game theoretical inlier selection exploits a specially designed payoff-function among cylinder candidates and allows the method to be general enough to be used without prior knowledge of the scene. Our approach is designed to exhibit a high robustness with respect to outliers, that makes it perfect to be applied in practical scenarios where a noisy point cloud is acquired without normal vectors. However, this limits its usage on massive point clouds where several cylindrical patches are simultaneously present (like in pipe-routing reverse engineering) or when the spatial resolution is low (for example, in some Lidar-acquired data).

Nevertheless, qualitative experiments show that the proposed extraction method offers high flexibility in very heterogeneous and complex scenes typically faced in robotics and industrial applications.

Appendix A. Dual quaternions and rigid motions

In this appendix we give a minimal introduction to dual quaternions to understand the proposed distance function, suggesting the reader to [51,52] for more information on dual numbers and dual quaternion algebra for geometrical applications.

Dual quaternions are mathematical objects defined by a composition of quaternions and dual numbers. The former are common for researchers and practitioners working with rotations, since it is well known that rotations around the origin can be efficiently represented by unitary quaternions. The latter is less known, but simple in practice. A dual number can be written as $\hat{a} = a_0 + \epsilon a_\epsilon$ where ϵ is called *dual unit*. The algebra is similar to complex numbers, except that $\epsilon^2 = 0$. Indeed, the dual conjugate is analog to complex conjugate $\bar{\hat{a}} = a_0 - \epsilon a_\epsilon$ and the multiplication is given by the formula $(a_0 + \epsilon a_\epsilon)(b_0 + \epsilon b_\epsilon) = a_0 b_0 + \epsilon(a_0 b_\epsilon + a_\epsilon b_0)$. The square root of a dual number with non zero scalar part a_0 is given by the formula

$$\sqrt{a_0 + \epsilon a_\epsilon} = \sqrt{a_0} + \epsilon \frac{a_\epsilon}{2\sqrt{a_0}}. \tag{A.1}$$

Interestingly, the Taylor series of a function with dual argument is limited to the first order because higher powers of ϵ are zero. Therefore, sine and cosine of a dual number are easily defined as:

$$\sin(a_0 + \epsilon a_\epsilon) = \sin(a_0) + \epsilon a_\epsilon \cos(a_0)$$
$$\cos(a_0 + \epsilon a_\epsilon) = \cos(a_0) + \epsilon a_\epsilon \sin(a_0).$$

Since the dual unit commutes with quaternion units (ie $\mathbf{i}\epsilon = \epsilon\mathbf{i}$), a dual quaternion $\hat{\mathbf{q}}$ can be written as an ordinary quaternion of dual numbers $\hat{\mathbf{q}} = \hat{w} + i\hat{x} + j\hat{y} + k\hat{z}$ (where $\mathbf{i} = (i, j, k)$) or as a dual number in which the scalar and dual part are ordinary quaternions $\hat{\mathbf{q}} = \mathbf{q_0} + \epsilon\mathbf{q}_\epsilon$. The norm of a dual quaternion is defined as $\|\hat{\mathbf{q}}\| = \sqrt{\hat{\mathbf{q}}^* \hat{\mathbf{q}}}$, where $\hat{\mathbf{q}}^*$ denotes the classical quaternion conjugation $\hat{\mathbf{q}}^* = \mathbf{q}^* + \epsilon\mathbf{q}_\epsilon^*$. The set of dual quaternions for which $\|\hat{\mathbf{q}}\| = 1$ are called *unitary*. It can be demonstrated that every rigid transformation can be represented by a unitary dual quaternion, and conversely, every unitary dual quaternion represents a rigid transformation. Finally, for the so-called *antipodal* property of dual quaternions, $\hat{\mathbf{q}}$ and $-\hat{\mathbf{q}}$ represent the same transformation but with two different trajectories.

Unitary Dual Quaternion Manifold

The screw motion between l_1 and l_2 can be related to the scalar and vector part of a unit dual quaternion $\hat{\mathbf{q}}$ according to the following formula:

$$\hat{\mathbf{q}} = \cos\frac{\hat{\theta}}{2} + \hat{\mathbf{s}}\sin\frac{\hat{\theta}}{2}$$
$$\hat{\theta} = 2\beta + \epsilon 2\|T\|$$
$$\hat{\mathbf{s}} = \mathbf{i}\vec{s} + \epsilon\mathbf{i}(m_1 \times \vec{s}) \tag{A.2}$$

where $\hat{\mathbf{s}}$ is a dual quaternion (with zero scalar part) describing the rotation axis and $\hat{\theta}$ is the dual angle expressing the angle of rotation and the amount of translation.

The set of unit dual quaternions can be seen as 6-dimensional manifold embedded an 8-dimensional Euclidean space [53]. The corresponding log map can be defined in closed form as:

$$log(\hat{\mathbf{q}}) = \hat{\mathbf{s}}\frac{\hat{\theta}}{2}. \tag{A.3}$$

Such log map allow us to define a distance function between two lines l_1 and l_2 induced by the length of the geodesic path in SE(3) from the origin (ie. the identity motion) to $\hat{\mathbf{q}}$. Let $\hat{\mathbf{q}}$ be the screw motion described in 3.1, we define such function as:

$$d(l_1, l_2) = \min(\|\log(\hat{\mathbf{q}})\|, \|\log(-\hat{\mathbf{q}})\|). \tag{A.4}$$

In fact, it represents the length of the shortest geodesic path of the two trajectories described by $\hat{\mathbf{q}}$ due to the antipodal property.

Operatively, the distance between two axes l_1 and l_2 in our application is computed as follows:

1. Determining the points m_1 and m_2, and the distance vector $T = m_2 - m_1$
2. Computing the axis vector $\vec{s}$
3. Computing the dual quaternion $\hat{\mathbf{q}}$ as in (A.2)
4. Computing the length of the geodesic path as in (A.4)

References

[1] J.-A. Beraldin, M. Picard, S.F. El-Hakim, G. Godin, V. Valzano, A. Bandiera, Combining 3d technologies for cultural heritage interpretation and entertainment, in: Videometrics VIII, 5665, International Society for Optics and Photonics, 2005, p. 56650C.

[2] T.S. Newman, A.K. Jain, A system for 3d cad-based inspection using range images, Pattern Recognit. 28 (10) (1995) 1555–1574.

[3] H.B. Adallah, J.-J. Orteu, B. Dolives, I. Jovančević, 3d point cloud analysis for automatic inspection of aeronautical mechanical assemblies, in: Fourteenth International Conference on Quality Control by Artificial Vision, 11172, International Society for Optics and Photonics, 2019, p. 111720U.

[4] J. Shen, W. Yang, Q. Liao, Part template: 3d representation for multiview human pose estimation, Pattern Recognit. 46 (7) (2013) 1920–1932.

[5] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, S. Savarese, Segcloud: semantic segmentation of 3d point clouds, in: 2017 International Conference on 3D Vision (3DV), IEEE, 2017, pp. 537–547.

[6] A. Goshtasby, Recovering scene structures from scattered surface points, Pattern Recognit. 26 (10) (1993) 1543–1547.

[7] Z. Zhang, J. Li, Y. Guo, X. Li, Y. Lin, G. Xiao, C. Wang, Robust procedural model fitting with a new geometric similarity estimator, Pattern Recognit. 85 (2019) 120–131.

[8] Y.M. Amir, B. Thörnberg, High precision laser scanning of metallic surfaces, Int. J. Opt. 2017 (2017).

[9] C.F. Olson, Locating geometric primitives by pruning the parameter space, Pattern Recognit. 34 (6) (2001) 1247–1256.

[10] A. Nurunnabi, Y. Sadahiro, D.F. Laefer, Robust statistical approaches for circle fitting in laser scanning three-dimensional point cloud data, Pattern Recognit. 81 (2018) 417–431.

[11] A. Kaiser, J.A. Ybanez Zepeda, T. Boubekeur, A survey of simple geometric primitives detection methods for captured 3d data, in: Computer Graphics Forum, 38, Wiley Online Library, 2019, pp. 167–196.

[12] M.R. Rahayem, J.A.P. Kjellander, Quadric segmentation and fitting of data captured by a laser profile scanner mounted on an industrial robot, Int. J. Adv. Manuf.Technol. 52 (1) (2011) 155–169.

[13] D. Hakala, R. Hillyard, P. Malraison, B. Nourse, Natural quadrics in mechanical engineering, CAD/CAM VIII, Autofact West, Anaheim, California, 1980.

[14] G. Lukács, A. Marshall, R. Martin, Geometric Least-Squares Fitting of Spheres, Cylinders, Cones and Tori, 2, 1997, pp. 1–20. RECCAD deliverable documents

[15] G. Wang, Z. Houkes, G. Ji, B. Zheng, X. Li, An estimation-based approach for range image segmentation: on the reliability of primitive extraction, Pattern Recognit. 36 (1) (2003) 157–169.

[16] F. Bergamasco, A. Albarelli, A. Torsello, Image-Space Marker Detection and Recognition using Projective Invariants, 2011, pp. 381–388, doi:10.1109/3DIMPVT.2011.55.

[17] F. Bergamasco, L. Cosmo, A. Albarelli, A. Torsello, Camera calibration from coplanar circles, in: 22nd International Conference on Pattern Recognition, IEEE, 2014, pp. 2137–2142, doi:10.1109/ICPR.2014.372.

[18] P. Polewski, W. Yao, M. Heurich, P. Krzystek, U. Stilla, A voting-based statistical cylinder detection framework applied to fallen tree mapping in terrestrial laser scanning point clouds, ISPRS J. Photogramm. Remote Sens. 129 (2017) 118–130.

[19] R. Qiu, Q.-Y. Zhou, U. Neumann, Pipe-run extraction and reconstruction from point clouds, in: European Conference on Computer Vision, Springer, 2014, pp. 17–30.

[20] R. Maalek, D.D. Lichti, R. Walker, A. Bhavnani, J.Y. Ruwanpura, Extraction of pipes and flanges from point clouds for automated verification of pre-fabricated modules in oil and gas refinery projects, Autom. Constr. 103 (2019) 150–167.

[21] M. Pistellato, F. Bergamasco, A. Albarelli, A. Torsello, Robust cylinder estimation in point clouds from pairwise axes similarities, in: Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods, 2019, pp. 640–647.

[22] G. Lukács, R. Martin, D. Marshall, Faithful least-squares fitting of spheres, cylinders, cones and tori for reliable segmentation, in: European Conference on Computer Vision, Springer, 1998, pp. 671–686.

[23] B. Paláncz, J. Awange, A. Somogyi, N. Rehány, T. Lovas, B. Molnár, Y. Fukuda, A robust cylindrical fitting to point cloud data, Aust. J. Earth Sci. 63 (5) (2016) 665–673.

[24] A. Nurunnabi, Y. Sadahiro, R. Lindenbergh, D. Belton, Robust cylinder fitting in laser scanning point cloud data, Measurement 138 (2019) 632–651.

[25] L. Busé, A. Galligo, J. Zhang, Extraction of cylinders and cones from minimal point sets, Graph. Models 86 (2016) 1–12.

[26] M. Rahayem, N. Werghi, J. Kjellander, Best ellipse and cylinder parameters estimation from laser profile scan sections, Opt. Lasers Eng. 50 (9) (2012) 1242–1259, doi:10.1016/j.optlaseng.2012.03.014.

[27] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395.

[28] P.H. Torr, A. Zisserman, Mlesac: a new robust estimator with application to estimating image geometry, Comput. Vision Image Understanding 78 (1) (2000) 138–156.

[29] R.C. Bolles, M.A. Fischler, A ransac-based approach to model fitting and its application to finding cylinders in range data., in: IJCAI, 1981, 1981, pp. 637–643.

[30] T. Chaperon, F. Goulette, Extracting cylinders in full 3d data using a random sampling method and the gaussian image, in: Vision Modeling and Visualization Conference, 2001.

[31] R. Schnabel, R. Wahl, R. Klein, Efficient RANSAC for point-cloud shape detection, in: Computer Graphics Forum, 26, Wiley Online Library, 2007, pp. 214–226.

[32] R.O. Duda, P.E. Hart, Use of the hough transformation to detect lines and curves in pictures, Commun. ACM 15 (1) (1972) 11–15.

[33] T. Rabbani, F. Van Den Heuvel, Efficient hough transform for automatic detection of cylinders in point clouds, in: ISPRS Wg III/3, III/4, 3, 2005, pp. 60–65.

[34] A.K. Patil, P. Holi, S.K. Lee, Y.H. Chai, An adaptive approach for the reconstruction and modeling of as-built 3d pipelines from point clouds, Autom. Constr. 75 (2017) 65–78.

[35] A.M. Araújo, M.M. Oliveira, Connectivity-based cylinder detection in unorganized point clouds, Pattern Recognit. 100 (2020) 107161.

[36] R. Figueiredo, A. Dehban, P. Moreno, A. Bernardino, J. Santos-Victor, H. Araújo, A robust and efficient framework for fast cylinder detection, Rob. Auton. Syst. 117 (2019) 17–28.

[37] L.C. Goron, Z.-C. Marton, G. Lazea, M. Beetz, Robustly segmenting cylindrical and box-like objects in cluttered scenes using depth cameras, in: ROBOTIK 2012; 7th German Conference on Robotics, VDE, 2012, pp. 1–6.

[38] T.-T. Tran, V.-T. Cao, D. Laurendeau, Extraction of cylinders and estimation of their parameters from point clouds, Comput. Graph. 46 (2015) 345–357.

[39] Y.-H. Jin, W.-H. Lee, Fast cylinder shape matching using random sample consensus in large scale point cloud, Appl. Sci. 9 (5) (2019) 974.

[40] K. Kawashima, S. Kanai, H. Date, As-built modeling of piping system from terrestrial laser-scanned point clouds using normal-based region growing, J. Comput. Des. Eng. 1 (1) (2014) 13–26.

[41] Y.-J. Liu, J.-B. Zhang, J.-C. Hou, J.-C. Ren, W.-Q. Tang, Cylinder detection in large-scale point cloud of pipeline plant, IEEE Trans. Vis. Comput. Graph. 19 (10) (2013) 1700–1707.

[42] F. Bergamasco, A. Albarelli, L. Cosmo, A. Torsello, E. Rodolà, D. Cremers, Adopting an unconstrained ray model in light-field cameras for 3d shape reconstruction, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2015, 2015, pp. 3003–3012, doi:10.1109/CVPR.2015.7298919.

[43] M. Chasles, Note sur les proprietes generales du systeme de deux corps semblables entr'eux et places d'une maniere quelconque dans l'espace; et sur le deplacement fini ou infiniment petit d'un corps solide libre, Bull. Sci. Math. Astronom.Phys. Chim. 14 (1830) 321–326.

[44] R. Halır, J. Flusser, Numerically stable direct least squares fitting of ellipses, in: Proc. 6th International Conference in Central Europe on Computer Graphics and Visualization. WSCG, 98, Citeseer, 1998, pp. 125–132.

[45] J. Weibull, Evolutionary Game Theory, MIT Press, Cambridge, Mass. [U.A.], 1995.

[46] P.D. Taylor, L.B. Jonker, Evolutionary stable strategies and game dynamics, Math. Biosci. 40 (1–2) (1978) 145–156.

[47] L. Kavan, S. Collins, C. O'Sullivan, J. Zara, Dual Quaternions for Rigid Transformation Blending, Tech. Rep. TCD-CS-2006-46, Trinity College Dublin, 2006.

[48] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, Surface reconstruction from unorganized points, in: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques, 1992, pp. 71–78.

[49] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, A.M. Dollar, The ycb object and model set: towards common benchmarks for manipulation research, in: Advanced Robotics (ICAR), 2015 International Conference on, IEEE, 2015, pp. 510–517.

[50] B. Calli, A. Singh, J. Bruce, A. Walsman, K. Konolige, S. Srinivasa, P. Abbeel, A.M. Dollar, Yale-CMU-Berkeley dataset for robotic manipulation research, Int. J. Rob. Res. 36 (3) (2017) 261–268.

[51] K. Daniilidis, Hand-eye calibration using dual quaternions, Int. J. Rob. Res. 18 (3) (1999) 286–298, doi:10.1177/02783649922066213.

[52] L. Kavan, S. Collins, J. Žára, C. O'Sullivan, Geometric skinning with approximate dual quaternion blending, ACM Trans. Graph. 27 (4) (2008) 105:1–105:23, doi:10.1145/1409625.1409627.

[53] J.M. McCarthy, Introduction to Theoretical Kinematics, MIT Press, Cambridge, MA, USA, 1990.

Filippo Bergamasco received the PhD degree in Computer Science from University Ca'Foscari Venice, Italy, in 2015. He is currently an Assistant Professor at Ca'Foscari University of Venice. His research interests are in the area of computer vision and pattern recognition, ranging from 3D reconstruction, camera calibration to structure from motion, structured-light scanning and augmented reality.

Mara Pistellato received her PhD degree in Computer Science from the University Ca'Foscari of Venice in 2020. Her research interests are in the areas of Computer Vision and Pattern Recognition. In particular, she is interested in 3D reconstruction techniques with focus on real-world applications. During her PhD she studied novel techniques to improve the quality of structured-light 3D reconstruction methods and she was also involved in technological transfer projects.

Andrea Albarelli works as a Computer Science professor with the University Ca'Foscari Venezia since 2019. His research interests are in the area of *Artificial Intelligence*, with a special focus on the design of disruptive data-driven methodologies to be applied on real-world scenarios. To this end, he works in close collaboration with companies willing to undertake a radical digital transformation process. He led several technological transfer projects, resulting in more than 70 research papers published in top international journals and presented in key Engineering conferences. He received several scientific and industrial recognitions, including the Nvidia best paper award, for his research on 3D data processing, and innovation grants from companies like Electrolux and TIM, for the technical contributions from several academic spin-offs he founded.

Andrea Torsello received his PhD in computer science at the University of York, UK. From 2007 he is with Ca'Foscari University of Venice, Italy, where he is Full Professor. His research interests are in the areas of Computer Vision and Pattern Recognition, in particular the interplay between Stochastic and Structural approaches as well as Game-Theoretic and Physical models, with applications in 3D reconstruction and recognition. Recently he has focused on the application of Structural Pattern Recognition techniques to Network Science. Professor Torsello has published over 150 technical papers in refereed journals and conference proceedings and is repeatedly in the program committees of various international conferences and workshops reference for the area. He is currently a member of the Editorial Boards of the international journals Pattern Recognition and Pattern Recognition Letters.