Deep Learning-Based Pose Regression for Satellites: Handling Orientation Ambiguities in LiDAR Data

Margherita Piccinin * and Ulrich Hillenbrand

Institute of Robotics and Mechatronics, German Aerospace Center (DLR), 82234 Weßling, Germany
Email: margherita.piccinin@dlr.de (M.P.); ulrich.hillenbrand@dlr.de (U.H.)
*Corresponding author

***Abstract*—In orbital spaceflight today, there is high demand for servicing of satellites, assembling of space structures, as well as clearing of orbits from harmful debris. Orbital robotics is a critical technology for accomplishing these tasks. On-board autonomy of servicing spacecraft requires imaging or 3D sensors, LiDAR in the case considered here, and intelligent processing of their data to estimate the relative pose between servicer and target satellite. In this study we investigate a parametrization for pose regression based on Deep Learning (DL) that can be superior to the standard parameters. In particular, we show that higher prediction accuracy can be achieved by adapting the parametrization to symmetries or more generally pose ambiguities of the target object. This result is established in extensive experiments on both synthetic and real LiDAR data for several DL-based methods. Moreover, our own lightweight network is both more accurate and faster than classical methods, even on a standard Central Processing Unit (CPU), and more accurate than also the other recent DL-based methods we compare to. Our synthetically trained regressor also achieves excellent sim2real transfer.**

***Keywords*—pose estimation, deep learning, LiDAR data, satellite, orbital robotics**

I. Introduction

The development of on-orbit robotic technologies, such as for servicing satellites, removing debris, or assembling space structures, becomes increasingly important for the maintenance of orbital assets and for the sustainability of the orbital environment. Several missions successfully demonstrated on-orbit robotic capture of a cooperative target using a space manipulator system, such as JAXA's ETS-VII [1, 2], DARPA's Orbital Express [3], and China's Aolong-1. Orbital ATK (Northrop Grumman) has developed the Mission Extension Vehicle (MEV-1 and MEV-2), which successfully completed docking operations with a client spacecraft [4]. There is a generally growing interest in the orbital robotics field, and indeed several mission studies are being carried out [5].

Spacecraft relative pose estimation is a crucial but still challenging capability for such missions. Early approaches require the target's cooperation and knowledge of the target's characteristics. Today, however, autonomous navigation skills that work reliably also for uncooperative targets are becoming a critical component for enabling the objectives of upcoming orbital mission. These missions typically require a low latency for relative navigation and robotic operation, and thus can not rely on ground commands in these phases. Altogether they have higher demands on advanced on-board data processing.

While cameras are still the most commonly used instruments for relative navigation, LiDAR sensors are an attractive alternative. In fact, they can reliably deliver range measurements and directly capture 3D target geometry, which is much more difficult to obtain with vision sensors. Moreover, LiDAR sensors are neither affected by the harsh orbital illumination conditions, nor by the presence of distant celestial bodies in their field of view, nor by the temperature of the target (as is the case for thermal cameras). On the downside, LiDARs are active sensors and hence typically have higher energy consumption than cameras.

Autonomous navigation techniques based on LiDARs have been used in past missions for cooperative targets and recently also for uncooperative targets. In 2009 the STS-128 mission demonstrated in orbit a TriDAR (triangulation + LiDAR)-based rendezvous and docking with the ISS. Experiments successfully included the automatic acquisition of the ISS, a targetless tracking and the demonstration of lighting immunity [6]. Another prominent case is the operation of the Automated Transfer Vehicle (ATV, 2008–2014), which employed as navigation sensor a videometer, a system that operates similar to a Flash LiDAR. At far distances, the sensor was employed to measure the range, range-rate, and LoS. Whereas at close distances, the relative position and relative attitude were estimated utilizing retro-reflectors on the ISS [7]. Of particular interest for the development of improved navigation algorithms was the ATV-5 experiment in 2014, which allowed the collection of synchronized data from visible, thermal cameras, and a scanning LiDAR. However, such data are not publicly

Manuscript received June 7, 2024; revised June 15, 2024; accepted July 24, 2024; published March 26, 2025.

available. In 2021, the commercial MEV-2 mission performed rendezvous, proximity operations, and docking with an uncooperative target in geostationary orbit, carrying a sensor suite consisting of visual cameras, thermal cameras, and a LiDAR. At far range, the LiDAR was used to detect and track the target, while at close range it was employed to support 6D relative pose estimation. Mission outcome statements point out the value of having LiDAR on-board, which provides robust relative navigation capabilities and is immune to any effect of lighting and time of day [4, 8]. For future rendezvous and docking missions of the ORION module with the lunar GATEWAY, a LiDAR has been selected as the primary sensor to perform pose estimation [9].

One challenge for any method for on-board 6D pose estimation from image or LiDAR data is the real-time requirement in relative navigation and robotic operation. This has to be met under the constraints of limited on-board computing resources, not comparable to state-of-the-art on-ground devices. A method is thus needed that gives accurate predictions at low computational costs.

Another challenge, particularly for LiDAR-based methods, is pose ambiguity in the typically sparse 3D point cloud, resulting from common geometric symmetries of satellites. These can be either exact or approximate, then broken only by minor structural elements that are not well captured in the LiDAR data. A method is thus needed that takes ambiguities into account in the predictions.

Finally, a major problem in the development and validation of LiDAR-based pose estimation algorithms is the absence of public benchmark datasets, especially with real data and accurate ground truth. Hence validating a method, also demonstrating good sim2real transfer, requires an adequate laboratory facility for taking real measurements and a realistic simulation of the data from the used LiDAR.

In this study we address all these challenges. Specifically, the main contributions are:

- We propose a lightweight DL-based method for regressing the 6D pose of a known target satellite from LiDAR data.
- We address the problem of pose ambiguities in the case of satellite targets with exact or approximate discrete rotational symmetries. We investigate and compare an adapted set of rotation parametrizations. They all ensure uniqueness and some additionally avoide discontinuities in the mapping from data (or its representation) to pose parameters, both being conditions for successfully training a neural network regressor.
- We present an extensive study of the performance of DL-based pose regression with discrete (approximate) rotational symmetries for a range of different pose parametrizations on both synthetic data from a realistic LiDAR simulation and real LiDAR data from our OOS-SIM lab facility [10]. The results confirm that an adapted parametrization with uniqueness and global continuity consistently outperforms standard parametrizations with an adaptation just for uniqueness.
- We make a comparison of our DL-based method to other recent DL-based methods as well as to classical methods for pose estimation. Our own lightweight network turns out to be both much more accurate and much faster than classical methods, even on a standard CPU, and more accurate than the other DL-based methods with similar runtimes.
- We demonstrate excellent sim2real transfer of our synthetically trained regressor, comparing different data augmentations to minimize the sim2real gap.

II. Related Work

In the space community, most of the research works for LiDAR-based pose estimation still rely on classical methods. The focus of such works is the initial pose acquisition, which is usually followed by some refinement steps via a local optimization method, typically the Iterative Closest Point (ICP).

Some of these methods are based on the principle of geometric hashing. In Ref. [11], the Polygonal Aspect Hashing (PAH) method is introduced, which performs polygonal matching via a lookup hash table. Similarly, the Congruent Tetrahedron Align (CTA) method [12] constructs the tetrahedron occupying the largest volume in the point cloud convex hull and finds the congruent one from a hash table.

Other methods don't rely on a feature description. In Ref. [13] an offline database is first built containing point clouds of the target with different rotations; then, in the online phase, the most likely template matching candidate is retrieved using a correlation function. Similarly, Guo *et al.* [14] constructs a database of binary silhouette images obtained by projecting point clouds, and then finds the best correspondence using a similarity metric on binary images.

Recently, some learning methods have been studied. In Ref. [15], a MultiLayer Perceptron (MLP) is used to process depth images and predict the cosines and sines of Euler angles, while a second MLP predicts the translation parameters based on the previously predicted rotation. However, a non-symmetric target is considered and only simulated data are employed to prove the method. Following a different idea, Zhang *et al.* [16] proposes an iterative procedure that first uses a Point Transformer block as embedding module to extract features from the source and target point clouds, and then uses a registration module relying on an attention mechanism to process the extracted features and derive correspondences and hence compute the relative transformation between the point clouds. These two steps are reiterated, transforming the source point cloud with the computed pose. The method is proven using only simulated uniform point clouds and for non-symmetric targets. In Ref. [17], the same authors propose an iterative Gaussian Mixture Model (GMM) method, where a neural network is used to learn point-to-component correspondences. In Ref. [18], the output of a

stereo-LiDAR is voxelized and processed by convolutional and affine layers to regress the relative pose of an axisymmetric space debris.

In contrast to these methods, we use PointNet layers to process directly the LiDAR point clouds, in a fashion similar to [19] but with deeper encoding, and we regress the 6D pose in a non-iterative way. Moreover, we also specifically address the problem of orientation ambiguities induced by exactly or approximately symmetric targets through proposing an adapted set of parametrizations for improved accuracy.

III. Problem Statement and Outline

Consider an object, in our case a satellite, that has an exact or approximate discrete symmetry of orientation, in the sense that the sensor used for data acquisition, in our case a LiDAR, does not allow to discriminate between a number of different orientations. The effect is an orientation ambiguity in the sensory data, even if some structural details of the object may actually break an exact symmetry but are not captured. In other words, using any standard rotation parametrization (e.g. Euler angles, quaternions, etc.), indistinguishable point clouds would be mapped to a set of non-unique orientation parameter values.

However, regression of the object orientation necessarily requires unique parameter values for each data sample. In this situation there are two ways to make parameter values unique:

- folding standard parameters from the full parameter space into a sector, effectively selecting a restricted range of parameter values that are by symmetry equivalent to all other parameter values;
- expanding standard parameters from a sector to the full parameter space, effectively selecting a restricted range of orientations that are by symmetry equivalent to all other orientations.

We call the former adaptation of standard parameters the *folded parameters*, the latter we call the *expanded parameters*.

Specifically, if an object has an (approximate) *n*-fold symmetry around an axis, a folded parameter for orientation around that axis only covers a fraction of 1/*n* of a full rotation, as orientation can be uniquely predicted only in this range. An expanded parameter is "upscaled"[1] from that restricted section to the full parameter range, while still representing only the discriminated range of orientations. See Fig. 1 for an illustration.

The folded parameters have an unavoidable discontinuity in their representation of orientation across the borders of the selected section, even when starting out from a parametrization that covers *all* orientations without discontinuities (such as quaternions), as is illustrated in Fig. 1. Discontinuities, however, are hard to follow accurately in learning-based regression, especially with a neural network which can naturally implement only continuous mappings.

The expanded parameters, on the other hand, can be made continuous everywhere by choosing a smooth expansion of globally continuous standard parameters, also illustrated in Fig. 1.

We verify in an extensive set of experiments with synthetic and real LiDAR data that DL-based pose regression indeed improves when using expanded parametrizations instead of various folded parametrizations. An allover best parameterization is also identified within our test set.

We also propose a lightweight neural network architecture that is not only more accurate but also faster than tested classical methods on a standard CPU. This aspect is a critical factor in space missions where computing resources are severely limited in most cases. Moreover, the accuracy achieved by our network is superior also to the ones from recent DL-based methods that we compare to, while having similar run times.

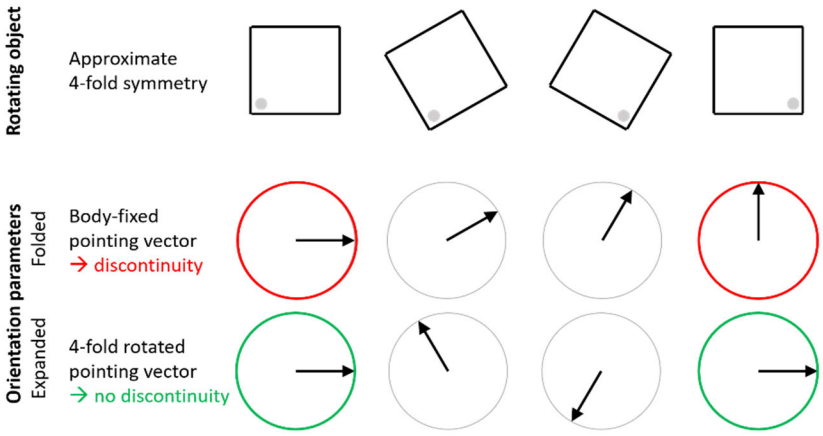


Fig. 1. Illustration in 2D of the effects of orientation ambiguity. Rotating square: approximate 4-fold symmetry (top row). For folded parametrizations a discontinuity problem occurs: body-fixed pointing vector with a restricted range (middle row). For an expanded version of the parameters the problem is eliminated: pointing vector with 4-fold rotation (bottom row).

[1] The parameter transformations needed for the expansion differ and are mostly nonlinear; it is usually not a rescaling in the literal sense.

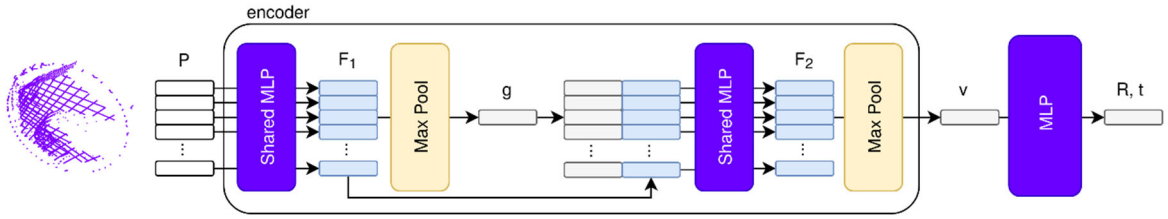


Fig. 2. Illustration of our DL architecture, P2PReg. For a given point cloud $\mathcal{P}$ we use an encoder based on two PointNet [20] blocks, in a fashion similar to [21]. The finally obtained encoding vector $\boldsymbol{v}$ is regressed via a MLP to the parameters for rotation and translation.

Note that continuous object symmetries are a different case where a degree of freedom entirely disappears, which is here not considered.

IV. Method

Given a 3D point cloud from a LiDAR, our goal is to estimate $[\boldsymbol{R}|\boldsymbol{t}] \in \mathrm{SE}(3)$, where $\boldsymbol{R}$ is the rotation matrix and $\boldsymbol{t}$ the translation vector, representing the 6D object pose of a known rigid object, a satellite which is quasi-symmetric in the cases considered here.

To this end we train a neural network for the regression of a set of pose parameters. The pose parameters to be regressed include folded ones as well as expanded versions for the pose ambiguities of the target satellite.

In this section we describe our network architecture and the pose parameters we compared for the regression task.

A. Network Architecture

As a pre-processing step, we have down-sampled the input point cloud with a voxel size of 0.03 m and subtracted its centroid from each point, followed by scaling to just fit into a unit sphere.

Our method, P2PReg (for Point cloud to Pose Regression), is illustrated in Fig. 2. The architecture is composed of a lightweight encoder that stacks two PointNet layers [20], followed by a MLP for regression of pose parameters. Unlike in [19], we do not separate the translation and rotation regressions, as we observed to achieve a better performance with a joint MLP.

The pre-processed point cloud $\mathcal{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m\}$ is an unordered set of m points, and the encoder is invariant to all point permutations. For encoding, each point $\boldsymbol{p}_i \in \mathbb{R}^3$ is individually processed via a shared two-layered MLP that outputs a feature $\boldsymbol{f}_i \in \mathbb{R}^k$ per point. This operation carried out for each point results in the matrix $\boldsymbol{F} \in \mathbb{R}^{m \times k}$ (k = 1024) of the point features. Then the maxpool operator extracts a global feature $\boldsymbol{g} \in \mathbb{R}^k$ with components $g_j = max_{i=1,\ldots,m}(F_{ij})$ and dimension 1024. The obtained global feature is then stacked on each point feature in $\boldsymbol{F}$, which then is point-wise parsed via another shared two-layered MLP, obtaining other point features. The maxpool operator is then applied again, finally yielding the encoding vector $\boldsymbol{v}$ of $\mathcal{P}$, which then is regressed to the pose parameters via a four-layer MLP. The encoding $\boldsymbol{v}$ has dimension 1024. We use ReLu activation for all hidden layers.

The encoding and the MLP on top are trained end-to-end for the regression. We train with a loss function that is the sum of a translation and a rotation loss term: $\mathcal{L} = \mathcal{L}_{\text{trans}} + \mathcal{L}_{\text{rot}}$. These loss terms depend on the chosen parameters.

B. Translation Parameters

As commonly done, we don't estimate directly the translation vector $\boldsymbol{t}$, but we decompose the translation as follows:

$$\boldsymbol{t} = \boldsymbol{c} + \boldsymbol{\delta} \tag{1}$$

where $\boldsymbol{c}$ is the centroid of the input scan. The centroid $\boldsymbol{c}$ can be computed for a given point cloud, and our network regresses to the translation residual $\boldsymbol{\delta}$. The loss that we choose for the translation parameters is the Euclidean distance: $\mathcal{L}_{\text{trans}} = \|\boldsymbol{\delta} - \hat{\boldsymbol{\delta}}\|$, where $\hat{\boldsymbol{\delta}}$ is the ground truth.

C. Rotation Parameters

We consider different parametrizations of the rotation, derived from standard parametrizations, which are adapted to the discrete (approximate) symmetry of the object: some folded and some expanded ones.

1) Folded parameters

Due to symmetry, or more generally ambiguity of orientation, the rotation parameter space can be split into several sectors that each represent the full range of appearances the target object can have in the LiDAR data; see Fig. 1. In order to cope with these orientation ambiguities, we need to transform the ground-truth and predicted orientation parameters such that they fall into only one of these sectors. The regressor is hence trained for the chosen sector only, but using point cloud data collected from the entire pose space.

Considering an n-fold (approximate) rotational symmetry, the folding is realized by mapping all orientation parameter values around the symmetry axis to fall into a range that corresponds to rotation angles α between 0 and $2\pi/n$, hence $\alpha \mapsto \alpha - 2\pi/n \lfloor \alpha n/2\pi \rfloor$. This folding transformation is not explicitly indicated in the parametrizations defined below, which hence appear like the corresponding standard parametrizations.

Folded Euler angles: We consider the Euler angles $\boldsymbol{e} = [\alpha, \beta, \gamma]$, which are respectively the roll, yaw, and pitch angles, where α is the rotation angle around the symmetry axis. The rotation matrix is computed as:

$$\boldsymbol{R} = \boldsymbol{R}_y(\gamma)\, \boldsymbol{R}_z(\beta)\, \boldsymbol{R}_x(\alpha)\,. \tag{2}$$

Such a representation has a discontinuity when an angle has value 0 or 2π. However, as we reduce the parameter space to one symmetric sector only, this discontinuity for

α is eliminated, while another one is introduced at the borders of the sector; see Fig. 1. Another issue of this representation is the Gimbal lock, which occurs when two rotation axes align, yielding the same rotation for a range of parameters. We choose as rotation loss $\mathcal{L}_{\text{rot}} = \text{MSE}(\boldsymbol{e}, \hat{\boldsymbol{e}})$, where $\text{MSE}(\cdot)$, denotes the mean-squared error and $\hat{\boldsymbol{e}}$ is the ground truth.

Folded quaternions: Unit quaternions ($\boldsymbol{q} \in \mathbb{R}^4$, $\|\boldsymbol{q}\| = 1$) can also be employed to represent 3D orientations. Rotations vary smoothly and without discontinuity over the $\mathbb{R}^4$ unit sphere, and there is no singularity like the Gimbal lock issue. However, two separate quaternions represent the same orientation. We choose as rotation loss $\mathcal{L}_{\text{rot}} = \text{MSE}(\boldsymbol{q}, \hat{\boldsymbol{q}})$, where $\hat{\boldsymbol{q}}$ is the ground truth.

Folded axis-angle: In the axis-angle representation, a vector $\boldsymbol{w} \in \mathbb{R}^3$ represents a rotation of $\theta = \|\boldsymbol{w}\|$ radians around the unit vector $N(\boldsymbol{w})$, where $N(\cdot)$ is the normalization function. Given an axis-angle representation, the corresponding rotation matrix $\boldsymbol{R}$ is obtained via the Rodrigues formula as:

$$\boldsymbol{R} = \exp(\boldsymbol{w}_\times) = \boldsymbol{I} + \frac{\sin(\theta)}{\theta}\boldsymbol{w}_\times + \frac{1-\cos(\theta)}{\theta^2}\boldsymbol{w}_\times{}^2, \quad (3)$$

where $\boldsymbol{w}_\times$ is the skew-symmetric matrix form of $\boldsymbol{w}$. As loss we choose the angle of the difference rotation: $\mathcal{L}_{\text{rot}} = \arccos(\text{Tr}(\hat{\boldsymbol{R}}\boldsymbol{R}^{-1} - 1)/2)$, where $\hat{\boldsymbol{R}}$ is the ground truth.

Folded pointing vectors: The pointing vectors parametrization, studied in [22], is a 6D continuous representation without singularities on SO(3). It consists of a pair of 3D vectors $\boldsymbol{r} = [\boldsymbol{r}_1, \boldsymbol{r}_2]$, which correspond to the first two columns of the rotation matrix. Via the inverse mapping the rotation matrix can be retrieved column-wise, $\boldsymbol{R} = [\boldsymbol{R}_1|\boldsymbol{R}_2|\boldsymbol{R}_3]$:

$$\begin{aligned} \boldsymbol{R}_1 &= N(\boldsymbol{r}_1) \\ \boldsymbol{R}_2 &= N(\boldsymbol{r}_2 - (\boldsymbol{R}_1 \cdot \boldsymbol{r}_2)\,\boldsymbol{R}_1), \\ \boldsymbol{R}_3 &= \boldsymbol{R}_1 \times \boldsymbol{R}_2 \end{aligned} \quad (4)$$

where $N(\cdot)$ is the normalization function. We choose as loss the Euclidean difference $\mathcal{L}_{\text{rot}} = \|\boldsymbol{r} - \hat{\boldsymbol{r}}\|$, where $\hat{\boldsymbol{r}}$ is the ground truth. This loss proved to be better than the angle of the difference rotation.

2) Expanded parameters

All of the above rotation parametrizations cause a discontinuity to the regression problem at the borders of the chosen section of the parameter space, see Fig. 1. We investigate an expansion of two of the parametrizations above in order to eliminate these symmetry-induced discontinuities by covering the full parameter space, thus removing the section borders. Again, we consider the case of an object with *n*-fold rotational symmetry, or more generally rotation ambiguity.

Expanded Euler angles: Choosing the roll axis as the symmetry axis, we define the expanded Euler angles $\boldsymbol{e}_{\text{ex}} = [e_1, e_2, e_3, e_4] \in \mathbb{R}^4$ as:

$$\begin{aligned} e_1 &= \sin(n\alpha) \\ e_2 &= \cos(n\alpha) \\ e_3 &= \beta \\ e_4 &= \gamma \end{aligned} \quad (5)$$

With this mapping, we achieve the same representation in the parameters space for equal ambiguous input point clouds. Moreover, by going from the roll angle α to its sine and cosine we eliminate the discontinuity at $\alpha = 0$ or 2π; cf. Fig. 1. The rotation matrix is:

$$\boldsymbol{R} = \boldsymbol{R}_y(e_4)\,\boldsymbol{R}_z(e_3)\,\boldsymbol{R}_x\left(\frac{\arctan(e_1/e_2)}{n}\right). \quad (6)$$

As loss we choose $\mathcal{L}_{\text{rot}} = \text{MSE}(\boldsymbol{e}_{\text{ex}}, \hat{\boldsymbol{e}}_{\text{ex}})$, where $\hat{\boldsymbol{e}}_{\text{ex}}$ is the ground truth.

Expanded pointing vectors: The expanded pointing vectors $\boldsymbol{r}_{\text{ex}}$ are defined as the first two columns of the expanded rotation matrix $\boldsymbol{R}_{\text{ex}} = \boldsymbol{R}_y(\gamma)\,\boldsymbol{R}_z(\beta)\,\boldsymbol{R}_x(n\alpha)$. The inverse mapping can be achieved in two steps: first retrieving $\boldsymbol{R}_{\text{ex}}$ using equation (4), then computing the expanded Euler angles (5), then unfolding the rotation passing through the mapping in equation (6). We choose the same loss as for the folded pointing vectors, the Euclidean difference $\mathcal{L}_{\text{rot}} = \|\boldsymbol{r}_{\text{ex}} - \hat{\boldsymbol{r}}_{\text{ex}}\|$, where $\hat{\boldsymbol{r}}_{\text{ex}}$ is the ground truth.

The same idea was proposed in [23], however, not using LiDAR data, and no experiments were carried with discrete rotational symmetry of objects.

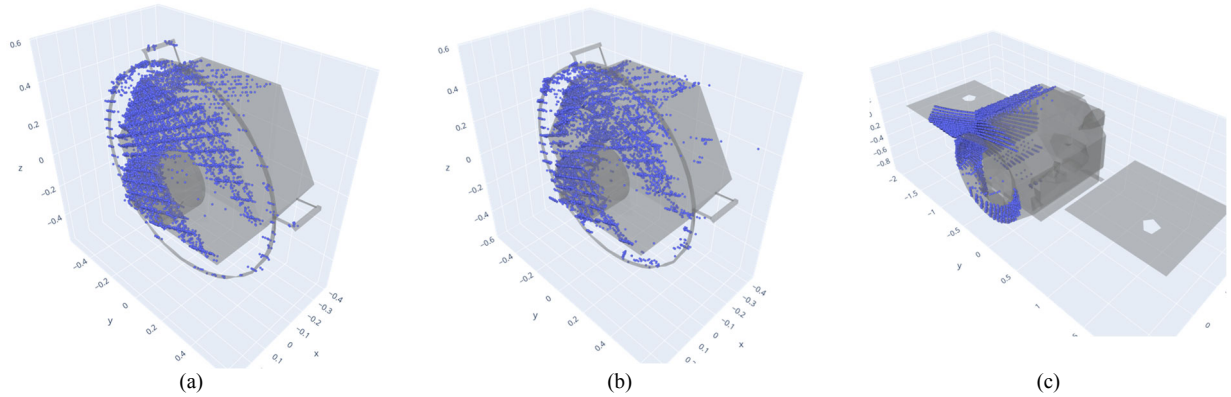
V. Experiments

In this section, we describe our experimental setup used for training and testing the different methods. We then investigate the effects of different parametrizations of pose on the regression performance. Next our P2PReg network with the best performing parametrization is compared to two recent alternative DL-based methods, both with their original and with our best parametrization. In this comparison we also include some classical methods of pose estimation. Finally, we show the dependence of sim2real performance on different ways of augmenting the training data.

A. Experimental Setup

1) Scenarios

For the purpose of training and testing our methods we have generated synthetic datasets for two different satellites, examples shown in Figs. 3(a) and (c): the OOS-SIM client satellite (`sat1`) and the Clementine satellite (`sat2`). Moreover, we have acquired real data for `sat1`, an example shown in Fig. 3(b).



(a) (b) (c)

Fig. 3. Illustration of the scenarios under study. We deal with partial, non-uniform point clouds affected by artifacts. (a) `sat1`: synthetic data; (b) `sat1`: real data; (c) `sat2`: synthetic data.

The OOS-SIM [10] is our lab facility for realistic simulation of on-orbit servicing scenarios. The facility is equipped with two satellites mock-ups, a servicer and a client (`sat1`). The servicer carries a lightweight robot arm with gripper to perform servicing tasks such as capturing of the client satellite. Two large industrial robots hold the satellite mock-ups, simulate their weightlessness and relative motion. In order to perform proximity operations with the client, the servicer is equipped with a stereo camera attached to the gripper of the robot and a LiDAR system consisting of two scanning LiDARs mounted on a boom attached to the servicer. The two LiDARs are VLP-16 by Velodyne™, rotated by 90° with respect to each other. This LiDAR system is simulated for synthetic data generation as well as used for real data acquisition; see section V.A.2) for details.

Clementine (`sat2`) was a NASA LEO satellite devoted to scientific observations and technology demonstrations, whose CAD model we took from [24].

These two satellites have been chosen because of the availability of CAD models, the availability of real data from our lab (`sat1`), and the quasi-symmetric geometry of the satellites. In particular, `sat1` has a 6-fold symmetry, broken by small details, like three handles and a grasping point; while `sat2` has a 2-fold symmetry dominated by the large solar panels, broken by a small appendage on the bottom. Of course, our method can generally work also for other (approximately) symmetric object shapes not considered in this study.

2) Dataset

The synthetic dataset for `sat1` contains 500k sample, with random uniform positions in the range 0.5 m to 2 m. The dataset for `sat2` contains 400k sample (i.e. fewer due to memory limitations during training), with random uniform positions in the range 2 m to 4 m. In both cases, rotations have been drawn randomly and uniformly by sampling quaternions. We split each dataset in training set (60%), validation set (20%), and testing set (20%).

The sensor that we simulate is the OOS-SIM dual-LiDAR system (VLP-16). Our data simulator includes anisotropic noise in the lasers' firing direction and beam divergence. The latter is obtained by aggregating for each laser several rays, slightly diverging from the nominal line of sight direction, and taking the average of their depth values. No reflections and motion blur effects are included in the simulator.

Moreover, in order to prove the methods against the sim2real gap, we have collected a dataset at the OOS-SIM facility in our lab for the `sat1` scenario, which consists of 190 sampled relative poses taken along 6 different trajectories. We refined the ground truth of the data by applying ICP using the satellite CAD model. As visible in Fig. 3(b), the experimental data contain several artifacts, such as noise in the firing direction, beam divergence, and reflections on the MLI highly reflective surface which covers the satellite's front panel. Comparing to Fig. 3(a), it can be qualitatively seen that a lot of these artifacts are reproduced in the simulated data.

3) Training

The model is trained on the synthetic data for 300 epochs with a batch size of 32, using the Adam optimizer with a learning rate of 1e-4 and a weight decay of 1e-6.

4) Evaluation metrics

For evaluation of our experimental results we use different error metrics. Since the methods under study cannot resolve the pose ambiguities, the relevant pose errors are the minimum ones achieved across all (approximately) symmetric variants of a predicted pose. Let $\boldsymbol{T} = [\boldsymbol{R}|\boldsymbol{t}]$ be the predicted pose, $\mathcal{S}_{\boldsymbol{T}}$ the set of poses which are equivalent by symmetry, and $\widehat{\boldsymbol{T}} = [\widehat{\boldsymbol{R}}|\widehat{\boldsymbol{t}}]$ be the ground-truth pose.

The minimum Average Distance of model points for Distinguishable (minADD) and for Indistinguishable (minADI) points, given a complete uniform point cloud $\mathcal{M}$ of the satellite model, are defined as:

$$\begin{aligned} \text{minADD} &= \min_{\boldsymbol{T}' \in \mathcal{S}_{\boldsymbol{T}}} \operatorname*{avg}_{\boldsymbol{p} \in \mathcal{M}} \left\| \widehat{\boldsymbol{T}}\boldsymbol{p} - \boldsymbol{T}'\boldsymbol{p} \right\|, \\ \text{minADI} &= \min_{\boldsymbol{T}' \in \mathcal{S}_{\boldsymbol{T}}} \operatorname*{avg}_{\boldsymbol{p}_1 \in \mathcal{M}} \min_{\boldsymbol{p}_2 \in \mathcal{M}} \left\| \widehat{\boldsymbol{T}}\boldsymbol{p}_1 - \boldsymbol{T}'\boldsymbol{p}_2 \right\|. \end{aligned} \quad (7)$$

The translational and rotational errors are defined as:

$$e_{\text{trans}} = \|\boldsymbol{t} - \hat{\boldsymbol{t}}\|, \; e_{\text{rot}} = \arccos\left(\text{Tr}\left(\widehat{\boldsymbol{R}}\boldsymbol{R}^{-1} - 1\right)/2\right) \quad (8)$$

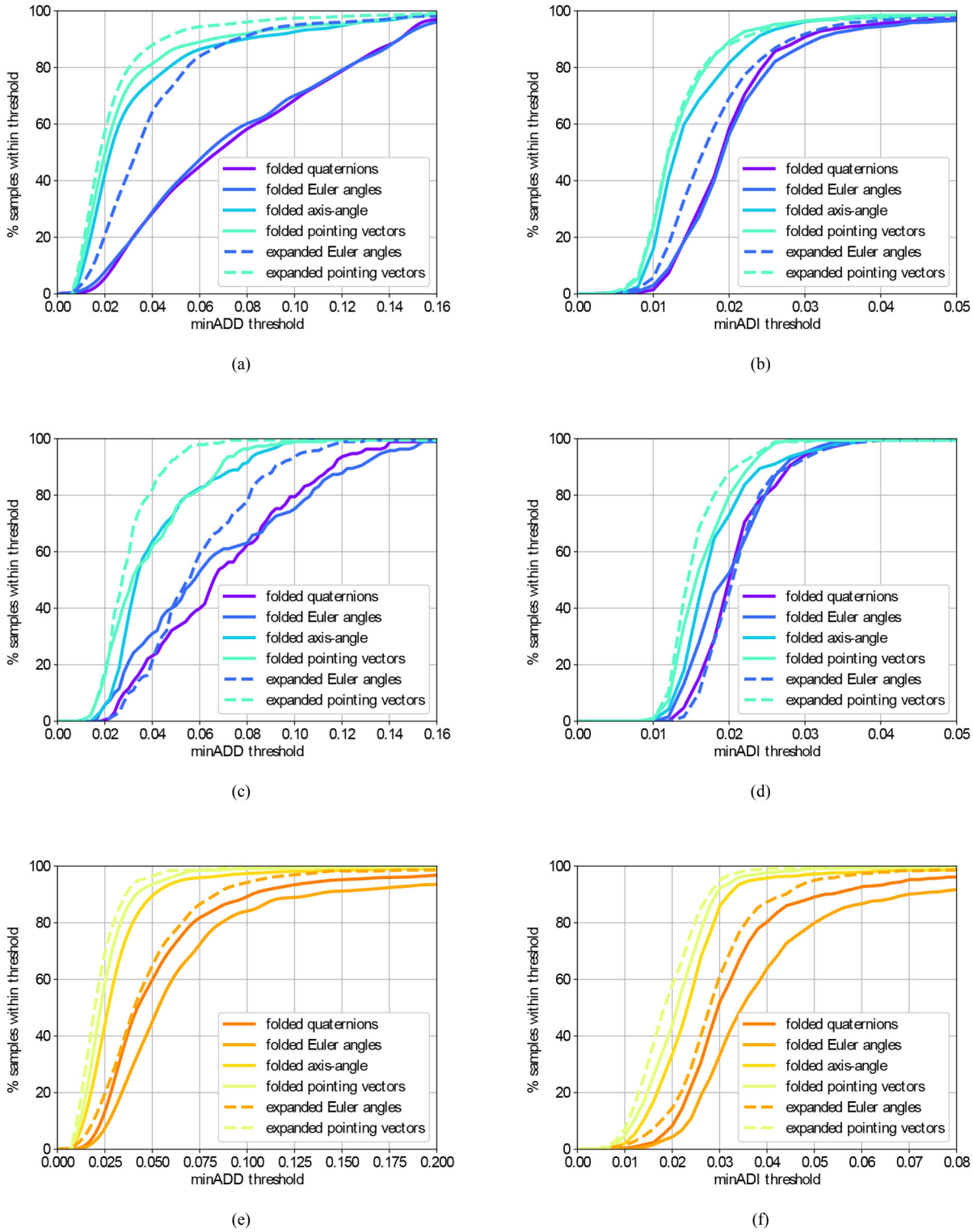


Fig. 4. Results of the comparison of the different pose parametrizations. Methods with expanded parameters clearly outperform the methods with corresponding folded parameters. The method with expanded pointing vectors as parameters has the allover best performance on both synthetic and real data, and on both targets. a) minADD for `sat1` on synthetic data; (b) minADI for `sat1` on synthetic data; (c) minADD for `sat1` on real data; (d) minADI for `sat1` on real data; (e) minADD for `sat2` on synthetic data; (f) minADI for `sat2` on synthetic data.

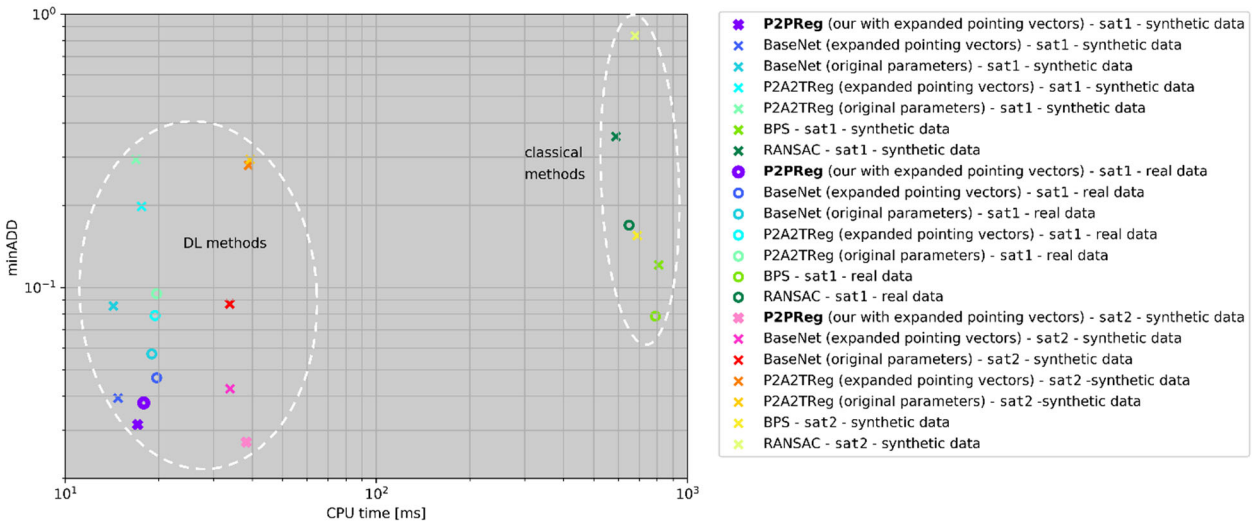


Fig. 5. Comparison between DL-based and classical methods in terms of accuracy and speed. Clearly P2PReg (our method) presents the best accuracy on both synthetic and real data, and for both targets.

Note that minADD and minADI are related to the ADD and ADI metrics that are respectively meant to penalize or tolerate pose variations due to (approximate) symmetries [25].

B. *Comparison of Pose Parametrizations*

The comparison of the different rotation parametrizations defined in Section IV.C is shown in Fig. 4. A local optimization for pose refinement (like ICP) can further improve the accuracies, but is not included here. In general, the global pose estimate evaluated here can serve as the initialization of an optional refinement method, and better initialization gives higher chance and speed of convergence.

The plots in Fig. 4 show the percentage of testing samples within the minADD/minADI thresholds, where such thresholds are a given as a fraction of the object diameter. The larger the area under the curve, the better the performance. As clearly visible in Fig. 4, the parametrizations with expanded pointing vectors and expanded Euler angles yield a superior performance compared to their folded versions. The beneficial effect of expanding parameters is verified for both of the target satellites (hence for different symmetries), and on both synthetic and real data (the latter only for `sat1`). This confirms the rationale we give in section III for choosing the expanded parameters for regression.

It can be noticed that the minADI is generally lower than the minADD. This is due to the fact that the minADI metric is less strict, since it considers the distance between nearest-neighbor points, not strictly corresponding points as minADD does.

Among the parametrizations studied, the best results are achieved with expanded pointing vectors, being superior to the other ones in all the test cases: it achieves a minADD below 0.05d, i.e., less than 5% of the object diameter, for 92% of the test samples in the `sat1` case with synthetic data and for 93% with real data, and for 96% in the `sat2` case with synthetic data.

C. *Comparison to Classical Methods and Other DL-Based Methods*

The space community still considers classical methods as the baseline for on-board algorithms. Only in recent years, learning methods are being considered, especially for image processing, and mostly at research level. We compare P2PReg (our DL-based method) using expanded pointing vectors as parameters with two different classical methods, in terms of both accuracy and speed.

The first classical method is the RANSAC algorithm as implemented in Open3D [26], which finds corresponding points between the source and the target point clouds by querying the nearest neighbors in the FPFH space [27]. In the case of `sat1`, the performance is further improved by checking the inlier count for known frequent cases of large misalignment and retaining the best scoring transformation. Such improvement is not implemented for `sat2`, as the number of occurring large-error cases is very high and would require a highly engineered solution.

The second classical method consists of encoding the input point cloud using the Basis Point Set (BPS) [28], which is an efficient non-learning method for the global representation of a point cloud. The training set (that was used for the DL-based methods) is still exploited for constructing a codebook of such global features. During inference, the given point cloud is transformed to the BPS encoding, and the nearest neighbor in the codebook is found, looking up the corresponding pose.

We also compare to two other DL-based methods that were proposed for satellite pose estimation. The first is BaseNet from [19]: two networks for separate regression of orientation and translation, using axis-angle parameters for orientation. The second method, P2A2TReg (for Point cloud to Attitude to Translation Regression, our terminology), is inspired by [15]: a first network predicts the orientation, a second predicts the translation from it (based on an implicit dependence of the offset of the data centroid from the model centroid on the viewpoint). While the original version uses depth images as input, we found

that turning our sparse LiDAR data into that format gives unsatisfactory results. We therefore have used our network for the orientation prediction, and the translation network from [15] for the translation prediction. Their original orientation parameters are the sine and cosine of the Euler angles. We implemented two versions for each of these DL-based methods, one with folded versions of their original orientation parameters, the other with the expanded pointing vectors, the best from our study.

The computational time for DL-based and classical methods is evaluated on a standard CPU, an Intel(R) Xeon(R) CPU E5-1630 v4 @ 3.70GHz.

In Fig. 5, the average minADD and computational time on the testing set are shown for each method. It can be seen that our P2PReg outperforms in accuracy the other DL-based methods, with just small differences in computational time. Moreover, the expanded pointing vectors as parameters improve the other DL-based method. It can be noticed that P2A2TReg has a poor performance compared to the other DL-based methods; it turns out that the translation offset does not depend uniquely on the target orientation, as the centroid of the LiDAR point clouds is different for different translations.

Our P2PReg outperforms the classical methods in both accuracy and computational time by a large margin. The RANSAC method is the one with poorest accuracy, especially in the `sat2` case. Regarding the BPS method, we noticed that the costliest step in terms of computational time is the nearest-neighbor search in the codebook.

D. Augmentations for sim2real

The sim2real domain gap is a key challenge for pose estimation especially in space applications, due to the lack of real data from orbit (or also planets) with accurate ground truth and to the stringent qualification requirements for space algorithms. From Figs. 4 and 5 it is evident that P2PReg transfers very successfully from synthetic training to real test data from the lab: the accuracy on synthetic and real test data is quite similar.

In Table I we present for P2PReg with expanded pointing vectors as parameters the effect of training on synthetic data with increasingly aggressive augmentations on real-data performance.

TABLE I. TEST RESULTS ON REAL DATA FOR DIFFERENT AUGMENTATIONS OF SYNTHETIC TRAINING DATA (BEST IN BOLD)

Augmentation	minADD	minADI	e_{trans} [cm]	e_{rot} [deg]
`no aug`	0.0570	0.0276	5.44	3.95
`aug1`	0.0599	0.0268	5.00	5.12
`aug2`	**0.0379**	**0.0191**	**3.34**	**3.08**
`aug3`	0.0394	0.0200	3.55	3.13

More specifically, `no aug` is training without data augmentations, `aug1` includes noise in the laser firing direction with varying intensity, `aug2` adds also outlier points, `aug3` adds deformations (e.g., shearing, tapering, and torsion on random axes). `aug2` achieves the best performance for all the metrics, considerably increasing the accuracy with respect to `no aug`. We can also observe that the severity of the applied augmentations should be limited, especially as we want to stay with a lightweight network due to the constrained computational resources on space hardware. All results in sections V.B and V.C were obtained using augmentation `aug2`.

Note that the effect of beam divergence is included in all the versions of the training data, i.e., not only as part of the augmentations. We have found that including beam divergence into the data simulation (see Section V.A.2) generally has a strong positive effect on real data performance.

VI. CONCLUSION

In this work, we presented a DL-based method to estimate the 6D pose of an exactly or approximately symmetric target satellite from LiDAR data. We handled the problem of the resulting orientation ambiguities by proposing expanded parametrizations of rotation and showed that they outperform their folded versions of standard parametrizations. Moreover, we have identified an allover best parametrization that yields the highest accuracy on all of our test cases.

Our lightweight deep network for directly processing LiDAR point clouds is both more accurate and faster than classical methods, even on a standard CPU, and more accurate also than other recent DL-based methods we compare to.

We also demonstrated that our synthetically trained method successfully overcomes the sim2real gap, using realistic data collected in our OOS-SIM facility.

Clearly, the superior performance of the expanded rotation parametrization in cases of orientation ambiguities can be expected to hold more generally for pose regression problems from other application domains, and when using other sensory data such as depth or RGB camera images.

The current study is limited to the evaluation of the methods on standard CPUs. Future work should include evaluation on space relevant hardware as well as validation and verification strategies to qualify DL-based methods for space applications.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Margherita Piccinin implemented the methods, conducted the experiments with evaluation, and contributed to the underlying concepts and the writing of the article. Ulrich Hillenbrand gave guidance in the research, contributed to the underlying concepts and to writing of the article. All authors have approved the final version.

FUNDING

This research was funded by the German Aerospace Center (DLR) as part of the projects ION and RICADOS 2.0 about on-orbit servicing.

REFERENCES

[1] K. Yoshida, "Engineering test satellite VII flight experiments for space robot dynamics and control: Theories on laboratory test beds ten years ago, now in orbit," *The International Journal of Robotics Research*, vol. 22, no. 5, pp. 321–335, 2003.

[2] M. Oda, "Space robot experiments on NASDA's ETS-VII satellite-preliminary overview of the experiment results," in *Proc. 1999 IEEE International Conference on Robotics and Automation*, 1999, vol. 2, pp. 1390–1395.

[3] D. A. Whelan, E. A. Adler, and S. B. Wilson III *et al.*, "DARPA Orbital Express program: Effecting a revolution in space-based systems," *in Proc. SPIE, Small Payloads in Space*, 2000, vol. 4136, pp. 48–56.

[4] M. Pyrak and J. Anderson, "Performance of Northrop Grumman's Mission Extension Vehicle (MEV) RPO imagers at GEO," in *Proc. SPIE, Autonomous Systems: Sensors, Processing and Security for Ground, Air, Sea and Space Vehicles and Infrastructure*, 2022, vol. 12115, pp. 64–82.

[5] E. Papadopoulos, F. Aghili, and O. Ma *et al.*, "Robotic manipulation and capture in space: A survey," *Frontiers in Robotics and AI*, vol. 8, 2021.

[6] S. Ruel and T. Luu, "STS-128 on-orbit demonstration of the TriDAR targetless rendezvous and docking sensor," in *Proc. 2010 IEEE Aerospace Conference*, Big Sky, 2010, pp. 1–7.

[7] J. A. Christian and S. Cryan, "A survey of LIDAR technology and its use in spacecraft relative navigation," in *Proc. AIAA Guidance, Navigation, and Control (GNC) Conference*, Boston, 2013.

[8] C. Schmitt, M. Windmüller, and M. Schwarz *et al.*, "RVS® 3000-3D LIDAR–3D imaging and pose estimation during first geo satellite servicing," in *Proc. of the 44th Annual American Astronautical Society Guidance, Navigation, and Control Conference*, 2022, pp. 1711–1717.

[9] C. Schmitt, M. Windmüller, and M. Schwarz *et al.*, "RVS® 3000-3D LIDAR–next stop: Gateway," *in Proc. the 44th Annual American Astronautical Society Guidance, Navigation, and Control Conference*, 2022, pp. 489–496.

[10] J. Artigas, M. De Stefano, and W. Rackl *et al.*, "The OOS-SIM: An on-ground simulation facility for on-orbit servicing robotic operations," in *Proc. 2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2015, pp. 2854–2860.

[11] S. Ruel, T. Luu, and M. Anctil *et al.*, "Target localization from 3D data for on-orbit autonomous rendezvous & docking," in *Proc. 2008 IEEE Aerospace Conference*, 2008, pp. 1–11.

[12] F. Yin, W. Chou, and Y. Wu *et al.*, "Sparse unorganized point cloud based relative pose estimation for uncooperative space target," *Sensors*, vol. 18, no. 4, p. 1009, 2018.

[13] R. Opromolla, G. Fasano, and G. Rufino *et al.*, "A model-based 3D template matching technique for pose acquisition of an uncooperative space object," *Sensors*, vol. 15, no. 3, pp. 6360–6382, 2015.

[14] W. Guo, W. Hu, and C. Liu *et al.*, "Pose initialization of uncooperative spacecraft by template matching with sparse point cloud," *Journal of Guidance, Control, and Dynamics*, vol. 44, no. 9, pp. 1707–1720, 2021.

[15] E. A. Pensado, L. M. G. de Santos, and H. G. Jorge *et al.*, "Deep learning-based target pose estimation using lidar measurements in active debris removal operations," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 5, pp. 5658–5670, 2023.

[16] S. Zhang, W. Hu, and W. Guo, "6-DoF pose estimation of uncooperative space object using deep learning with point cloud," in *Proc. 2022 IEEE Aerospace Conference (AERO)*, 2022, pp. 1–7.

[17] S. Zhang, W. Hu, and W. Guo *et al.*, "Neural-network-based pose estimation during noncooperative spacecraft rendezvous using point cloud," *Journal of Aerospace Information Systems*, vol. 20, no. 8, pp. 462–472, 2023.

[18] S. Hashimoto, Y. Nakajima, and N. Ishihama, "6-DoF pose estimation from stereo LiDAR of actual machine using deep learning," in *Proc. 2023 IEEE Aerospace Conference*, 2023, pp. 1–11.

[19] G. Gao, M. Lauri, and Y. Wang *et al.*, "6D object pose regression via supervised learning on point clouds," in *Proc. 2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 3643–3649.

[20] C. R. Qi, H. Su, and K. Mo *et al.*, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 652–660.

[21] W. Yuan, T. Khot, and D. Held *et al.*, "PCN: Point completion network," in *Proc. 2018 International Conference on 3D Vision (3DV)*, 2018, pp. 728–737.

[22] Y. Zhou, C. Barnes, J. Lu, and J. Yang *et al.*, "On the continuity of rotation representations in neural networks," in *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5745–5753.

[23] A. Saxena, J. Driemeyer, and A. Y. Ng. "Learning 3-D object orientation from images," in *Proc. 2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 794-800.

[24] B. Kumanchik. (2018). NASA 3D resources. [Online]. Available: https://nasa3d.arc.nasa.gov/detail/clementine

[25] T. Hodaň, J. Matas, and Š. Obdržálek, "On evaluation of 6D object pose estimation," in *Proc. Computer Vision–ECCV 2016 Workshops*, Amsterdam, 2016, pp. 606–619.

[26] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *CoRR*, 2018. http://arxiv.org/abs/1801.09847

[27] R. B. Rusu, N. Blodow, and M. Beetz, "Fast Point Feature Histograms (FPFH) for 3D registration," in *Proc. 2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3212–3217.

[28] S. Prokudin, C. Lassner, and J. Romero, "Efficient learning on point clouds with basis point sets," in *Proc. the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4332–4341.