CNN-based Pose Estimation of a Non-Cooperative Spacecraft with Symmetries from Lidar Point Clouds

LÉO RENAUT
German Aerospace Center (DLR), Wessling, Germany, and Julius-Maximilians-Universität Würzburg, Informatics XVII - Robotics, Germany

HEIKE FREI
German Aerospace Center (DLR), Wessling, Germany

ANDREAS NÜCHTER
Julius-Maximilians-Universität Würzburg, Informatics XVII - Robotics, Germany

***Abstract*— Light detection and ranging (lidar) sensors provide accurate 3D point clouds for non-cooperative spacecraft pose estimation. Several robust methods such as Iterative Closest Point (ICP) exist to perform a local refinement of the pose starting from an initial estimate. However, finding the initial pose of the spacecraft is a global optimization problem which is challenging to solve in real-time. This is especially true on space hardware with limited computing power. In addition, many spacecrafts have a shape with multiple symmetries, making an unambiguous initial pose estimation impossible. This work introduces a Convolutional Neural Network (CNN) based pose estimation method, accounting for potential symmetries of the target satellite. The point clouds are projected to a 2D depth image before being processed by the network. To generate a sufficient amount of training data, a lidar simulator integrating multiple effects such as reflections or laser beam divergence is developed. While being trained solely on synthetic point clouds, the pose estimation method shows to be precise, efficient and reliable when evaluated on real point clouds taken at a hardware-in-the-loop rendezvous test facility. A runtime evaluation on potential space computing hardware is also performed to demonstrate the applicability of the method to real-time onboard pose estimation.**

***Index Terms*—Non-cooperative spacecraft, Pose estimation, Lidar, CNN**

Manuscript received XXXXX 00, 0000; revised XXXXX 00, 0000; accepted XXXXX 00, 0000.

Léo Renaut and Heike Frei are within the German Aerospace Center (DLR), 82234 Wessling, Germany (e-mail: leo.renaut@dlr.de; heike.frei@dlr.de). Andreas Nüchter is within the Julius-Maximilians-Universität Würzburg, Informatics XVII - Robotics, 97074 Würzburg, Germany (e-mail: andreas.nuechter@uni-wuerzburg.de). *(Corresponding author: L. Renaut)*

I. INTRODUCTION

SPACE rendezvous technology is a component of many modern missions aiming to perform in-orbit maintenance of space infrastructure. Such missions involve in-orbit refueling and robotic operations [1], lifetime extension [2], inspection of a damaged or unknown object [3], or active debris removal [4]. In all these cases, an active satellite, the chaser, needs to autonomously navigate towards an inactive or damaged target satellite.

In close range, at a distance below hundred meters to the target, precise relative navigation becomes necessary to perform proximity operations. It implies the estimation of the relative position and attitude (pose) of the target satellite through the use of electro-optical sensors embarked on the chaser. If visual cameras are used for relative pose estimation, they are affected by the strongly varying illumination conditions in orbit, such as frequent day / night eclipses, Sun blindings or reflections on the target, and Earth albedo [5]. On the contrary, active sensors such as light detection and ranging (lidar) or Time-of-Flight (ToF) cameras are less affected by the external conditions, so that they can provide measurements of the target over a complete orbit. While ToF cameras using a modulated light source are limited to maximum working ranges around 10 m [6], [7], direct ToF sensors, also known as flash lidars, can achieve ranges in the order of kilometers [8], [9], [10]. Likewise, existing scanning lidars can operate in space up to a few kilometers distance [2].

A lidar produces a 3D point cloud which can be used for pose estimation [11], [12], [13], [6], [14]. It is assumed that a 3D model of the target satellite is known. This model is either available prior to the mission, or is the result of an inspection flight around the target in a previous phase. The alignment of the recorded point clouds with the model of the target satellite is usually split into two steps [15]: First, a pose initialization method provides an initial estimate of the relative pose. Second, this pose is refined using a local optimization method such as Iterative Closest Point (ICP) [16]. In most cases, the initialization is only performed once. For all subsequent point clouds, the previous estimate is used as an initial estimate [17]. Yet re-initialization is also possible in case of a failure, or if the target was outside the sensor's field of view for some time.

While pose refinement or tracking is a local optimization method which is solved using gradient-based methods, pose initialization is a global optimization problem, where the search space consists of all poses. Therefore, finding an initial estimate of the spacecraft's pose in real-time is challenging, especially when using space onboard computers. Due to radiation-hardening, power and thermal constraints, these devices typically have much less computing power than what is used on ground [18], [19]. Another difficulty comes from the fact that many spacecrafts consist of simple shapes presenting symmetries. With a symmetrical target, it might not be feasible

to decide between the different attitude candidates for the relative pose.

This work introduces an efficient Convolutional Neural Network (CNN)-based pose estimation architecture, additionally taking into account potential symmetries of the target. The main contributions are:

- A pipeline for non-cooperative pose estimation, by projecting lidar point clouds to depth images prior to performing CNN-aided pose estimation,
- A flexible formulation of the attitude constraint in case of a symmetrical spacecraft,
- The development of a high-fidelity lidar simulator for extensive training on synthetic data only,
- An evaluation on real lidar data taken at a hardware-in-the-loop facility,
- The demonstration of the onboard capability of the method by evaluation on possible space computing hardware.

The remainder of this paper is organized as follows: Section II reviews the state-of-the-art, while the pose estimation architecture is detailed in Section III. The lidar simulator as well as the datasets used for training and evaluation are presented in Section IV. Section V discusses the results of the method on lidar data taken at a hardware-in-the-loop facility, as well as the results of the evaluation on space computing hardware. Finally, Section VI concludes the paper.

II. RELATED WORK

One type of methods for pose estimation are feature based. When using visual cameras, feature based methods rely on the extraction and matching of features on the image, before usually retrieving 2D to 3D correspondences via a Perspective-n-Point (PnP) method [20], [21]. Feature descriptors also exist for 3D point clouds, and have been used to provide an initial pose estimate before performing ICP refinement [11]. Yet hand-crafted feature descriptors do not necessarily generalize well to different data such as sparser or noisier point clouds [22]. A stereo and a ToF camera can be combined in a hybrid approach [23]. The authors suggest to first extract feature points on the stereo camera images, and match them making additional use of the depth information gathered by the ToF sensor. Pose refinement is then performed by means of ICP on the point cloud.

Alternatively, polygonal matching techniques have been tested for pose acquisition. The principle is to select points from the point clouds forming polygons respecting certain properties, such as congruence or coplanarity. Polygons with similar properties between the two point clouds are matched, before applying a Random Sample Consensus (RANSAC)-based voting scheme. Such an algorithm was used to test the TriDAR system embarked onboard the Discovery Shuttle [12]. For efficient matching, the database of polygons from the 3D model of the spacecraft is built online, and correspondences between the model and target point cloud are retrieved via a hash map. A similar algorithm for extracting congruent tetrahedrons is proposed and tested on sparse synthetic point clouds [13], but it is unclear how the method generalizes to different target shapes. Instead of polygons, it is possible to match oriented point pairs using a geometric descriptor, a hash table and a voting scheme for efficient matching [24]. Such an algorithm was applied to pose initialization of a spacecraft using a ToF camera [7].

The alignment of two point clouds is a global optimization problem. Hence, methods for global optimization such as global branch and bound ICP (Go-ICP) [25] have been suggested for pose acquisition of a spacecraft using ToF cameras [6]. However, these methods are not applicable in real-time [25]. For pose refinement, ICP or one of its variants is most commonly used [15]. Recently, an alternative pose refinement method based on a variant of the Normal Distribution Transform (NDT) algorithm [26] has been introduced for efficient tracking of a spacecraft [17]. Hybrid methods or sensor setups can also be used for tracking. In [27], the amplitude and depth image of a ToF camera are processed separately and fused into a single estimate. The method is validated in hardware-in-the-loop experiments. [28] introduces a pose estimation method based on a fusion of two point clouds originating from different sources, a Kinect and a lidar. The tracking result can also be used for estimating the inertia parameters of the target in case they are unknown [29].

For pose acquisition without hand-crafted features or polygons, a template matching method has been proposed [14]. The principle is to discretize the attitude space, and to build offline a database of model point clouds corresponding to each discrete attitude. During online matching, the current point cloud is compared to the database with the ICP metric to retrieve the most likely attitude candidate. In spite of an acceleration proposed by discarding templates which have a different distribution of points than the current point cloud, the method is computationally expensive. In follow-up work [30], Principal Component Analysis (PCA) is performed to first determine the main axis of the spacecraft in case of an elongated target. Thereafter, the roll angle around the principal axis is found by matching the point cloud with a database of templates computed online. To achieve real-time capability, a variant of the template matching algorithm is introduced, in which the 3D point clouds are projected to 2D binary silhouette images [31]. Silhouette images are correlated with each other using a binary similarity metric, thus avoiding the expensive nearest neighbor search required in 3D. In spite of its efficiency, the method exhibits a success rate of 60 % to 90 % on synthetic point clouds.

In recent years, the attention has shifted to the use of neural networks and in particular CNNs for camera based pose estimation of spacecrafts [5]. Using visual cameras, a first step is to crop the part of the image containing the spacecraft, before applying a second CNN to extract

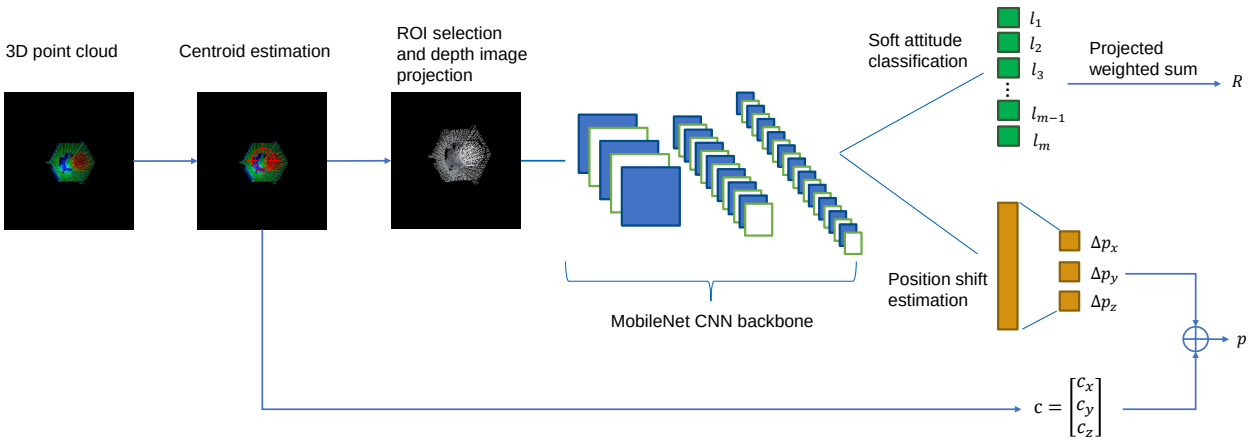


Fig. 1: Architecture of the pose initialization strategy.

feature points and match them to the model using a PnP scheme [32]. Alternatively, single stage approaches consisting in directly extracting the features on the complete image have been proposed [33]. To enable faster inference time compatible with real-time applications, neural network based pose estimation is implemented on a Field Programmable Gate Array (FPGA) [34]. Another possibility is to use lightweight CNN architectures such as MobileNetV2 [35], which show real-time capability on flight computing hardware for pose estimation tasks [36]. Instead of using CNNs for feature extraction, direct pose estimation methods have been proposed [37], [38]. They rely on multiple prediction heads to estimate the position as well as the attitude by discretizing the attitude space and performing a soft attitude classification, for which the attitude labels have continuous probability values. Recently, a multiple purpose neural network based on EfficientNet [39] combined with a second bi-directional network was introduced for pose estimation, background segmentation and feature detection [40].

A crucial point when training neural networks is the quantity and quality of the training data. For image based spacecraft pose estimation, several synthetic or hybrid datasets have been published [41], [38], [33], [42]. When training on synthetic data, a performance gap once the models are evaluated on real data is observed, an issue known as "domain gap". Hence, data augmentation techniques as well as domain randomization approaches consisting in randomizing the properties of the target spacecraft are used [43], [44]. Another possibility is to perform online refinement of the network [40]. Many spacecraft present symmetrical shapes, a possibility which is only rarely accounted for [44]. If the pose estimation method is not adapted for dealing with the symmetries, the attitude error and ambiguity in the pose estimation can be critical [42].

Specific neural network architectures have recently been developed for point cloud data processing [45], [46], but are not yet as advanced as their 2D counterparts. Due to the efficiency and advancement of CNN architectures for monocular vision, an alternative is to project 3D point clouds to depth images for processing via a 2D CNN. By projecting the point cloud in the three different directions of space, a three-channels image is created, and passed to a recurrent CNN for continuous pose estimation over a sequence of point clouds [47]. Similarly, a CNN is combined with a Recurrent Neural Network (RNN) for continuous feature extraction and pose estimation of point clouds projected to 2D images in the context of lunar landing of a spacecraft [48]. However, these methods require a Graphics Processing Unit (GPU) to run in real-time.

Similar to our work, in [49], the point cloud of a non-cooperative target acquired with a simulated ToF camera is projected on a normalized depth image, before being processed by a fully connected network, also called Multilayer Perceptron (MLP). The attitude estimate is then passed to a second MLP to retrieve the relative position. In contrast, we make use of a CNN for processing the depth image in a single step, and use an attitude classifier to handle symmetries of the target spacecraft.

III. POSE ESTIMATION ARCHITECTURE

In view of the advances and efficiency of modern CNN architectures, and their growing utilization for camera based pose estimation, this work proposes the use of an efficient CNN backbone for lidar based pose estimation. The different stages of the pose estimation pipeline are summarized in Fig. 1. In a first step, the point cloud is cropped and projected to a depth image, before being passed to a CNN backbone with two prediction heads, one for attitude estimation, and one for position estimation.

A. Depth image projection

1. *Trimmed centroid estimation*

The first step when a new 3D point cloud is received is to compute its centroid, which will serve as a coarse

first estimate of the position of the target's center of mass. Due to the high reflectivity of the materials which usually compose the spacecraft such as solar panels and golden Multilayer Insulation (MLI) sheets, the point clouds present some artifacts or ghost reflections. For eliminating these potential artifacts, a simple but robust method is proposed. It consists in computing the trimmed centroid, which is obtained by calculating the trimmed mean of the point cloud coordinates along each axis. In this work, a trimming factor of 25 % (interquartile mean) is chosen.

Let's consider a point cloud with N points $\boldsymbol{p}_i = (x_i, y_i, z_i)^T$ for $1 \leq i \leq N$. The first step is to sort the coordinates of the points independently for each dimension. There exists re-orderings $\sigma_x, \sigma_y, \sigma_z$ such that the coordinates are sorted in ascending order:

$$\begin{aligned} x_{\sigma_x(1)} &\leq \ldots \leq x_{\sigma_x(N)}\,, \\ y_{\sigma_y(1)} &\leq \ldots \leq y_{\sigma_x(N)}\,, \\ z_{\sigma_z(1)} &\leq \ldots \leq z_{\sigma_x(N)}\,. \end{aligned} \tag{1}$$

The trimmed centroid $\boldsymbol{c}$, is obtained as the mean along each coordinate after having discarded a certain fraction corresponding to lowest and highest values. This enables to filter out potential outliers. For a trimming factor of 25 %, the $N/4$ lowest and highest values are removed, so that the trimmed centroid is

$$\boldsymbol{c} = \frac{2}{N} \sum_{i=N/4+1}^{3N/4} \begin{pmatrix} x_{\sigma_x(i)} \\ y_{\sigma_y(i)} \\ z_{\sigma_z(i)} \end{pmatrix}\,. \tag{2}$$

2. Region of interest projection

A Region Of Interest (ROI) is selected around the centroid $\boldsymbol{c}$. It consists of a 3D bounding box aligned with the axes of the sensor. The size of this bounding box should be large enough to contain the whole spacecraft and leave room for errors on the centroid estimation. All points outside the ROI are discarded for further processing, so that this method enables to remove first outliers. In addition, setting the bounding box ensures that the region to be projected always has the same size, independently of the distance of the target to the sensor.

Given the size of the bounding box l, a point $\boldsymbol{p}_i$ is kept in the ROI is all three coordinates of $(\boldsymbol{p}_i - \boldsymbol{c})$ are within $[-l/2, l/2]$. The ROI is cube, and not, for example, a sphere, because the point cloud will be projected on a square image afterwards. Even though the target could have a main dimension, the ROI has the same size in each dimension because the orientation of the target is unknown at this stage.

The point cloud is projected to a depth image using a parallel projection along the sensor's optical axis, as schematically represented on Fig. 2. It is important to note that this projection is not made using a pinhole camera model. The optical axis of the lidar sensor is denoted by z, so that the image plane on which the point cloud is projected is (x, y).

For each point $\boldsymbol{p}_i$, the pixel to which it belongs is directly identified by its coordinates (x_i, y_i) with respect

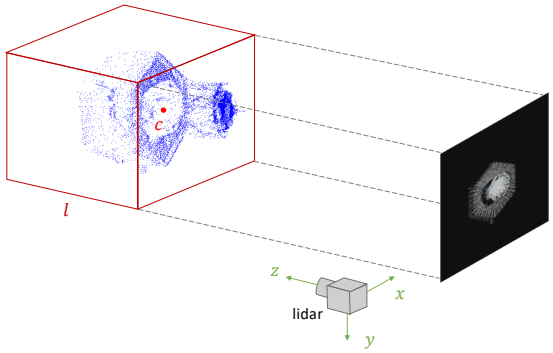


Fig. 2: Parallel projection of a point cloud on the image plane.

to the centroid coordinates (c_x, c_y) on the image plane. Each pixel holds a depth value, which is obtained by averaging the z coordinates of all points $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n_{x,y}}$ belonging to that pixel. Afterwards, the depth coordinate of the centroid c_z is subtracted to each depth value so that the absolute distances to the sensor are replaced by relative distances to the centroid. Finally, the depth values are normalized to be in the range $[0, 1]$. The depth value $v_{x,y}$ of pixel (x, y) is given by

$$v_{x,y} = \frac{1}{n_{x,y}} \sum_{i=1}^{n_{x,y}} \frac{(\boldsymbol{p}_i)_z - c_z + l/2}{l}\,. \tag{3}$$

This expression is only valid if $n_{x,y} > 0$. If no points belong to that pixel, the depth value is set to 0.

In accordance with the input resolution of many CNNs, the resolution of the depth image is chosen to be 224×224. Because each depth value is only computed relatively to the centroid's depth, the absolute position information is removed from the depth image. In addition, the parallel projection instead of a weak perspective projection ensures that the shape of the depth image does not depend on the absolute position of the target. This enables to further decouple the position estimation problem from the attitude estimation problem.

B. Position estimation

One output of the pose estimation is the relative position of the target's center of mass in the lidar coordinate frame, $\boldsymbol{p}$. Having estimated the centroid $\boldsymbol{c}$ of the point cloud in the previous step, the remaining position shift to estimate is

$$\Delta\boldsymbol{p} = \boldsymbol{p} - \boldsymbol{c}\,. \tag{4}$$

As illustrated on Fig. 1, one output of the CNN is the estimation of this position shift, based only on the depth image. While the image has been scaled in the projection step to remove the absolute position information, it still contains information about the position of the target with respect to the centroid, which is located in the middle of the image at a depth of 0.5.

C. Attitude classification

1. *Attitude sampling*

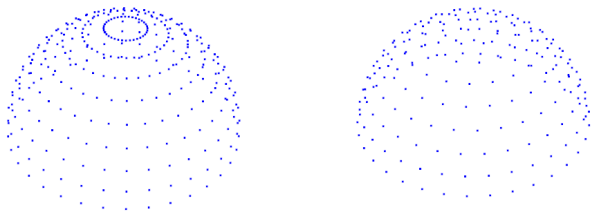
The relative attitude of the target spacecraft in the lidar coordinate frame is denoted by the rotation matrix $R \in SO(3)$. Instead of directly estimating this attitude, a soft attitude classification is performed by the CNN, similarly to the work of [37], [38]. Hence, the attitude space has to be sampled in distinct classes. For this sampling, the roll ($\phi \in [0, 2\pi]$), pitch ($\theta \in [-\pi/2, \pi/2]$) and yaw ($\psi \in [0, 2\pi]$) angles of the target are defined such that

$$R = R_z(\psi)R_y(\theta)R_x(\phi)\,. \tag{5}$$

If the three Euler angles are sampled uniformly, the resulting distribution of attitudes is not uniform in the attitude space, as illustrated on Fig. 3a. Some authors propose alternative strategies such as random sampling [37] or triangular mesh sampling [31]. We seek for a simple and deterministic sampling strategy, and apply spiral sampling as introduced in [50]. For a desired number of samples n_s on the unit sphere, the discrete yaw and pitch values are

$$\begin{cases} \psi_k &= \sqrt{n_s \pi}\, \arcsin \frac{2k-(n_s-1)}{n_s} \\ \theta_k &= \arccos \frac{2k-(n_s-1)}{n_s} - \frac{\pi}{2} \end{cases} , k = 0, ..., n_s - 1\,. \tag{6}$$

The roll ϕ is sampled uniformly in $[0, 2\pi]$ with a step $\Delta\alpha$. The result of such a sampling is presented on Fig. 3b.



(a) (b)

Fig. 3: Distribution of points on the half-sphere for **(a)** naive uniform sampling of pitch and yaw angles, or **(b)** the spiral sampling strategy.

In order to get a uniform sampling in the attitude space, given the desired angular resolution of the sampling $\Delta\alpha$, the number of points to take on the sphere is

$$n_s = \lfloor \frac{4\pi}{\Delta\alpha^2} \rfloor\,, \tag{7}$$

where $\lfloor\cdot\rfloor$ represents the integer part of a real number.

2. *Soft classification in the non-symmetric case*

In a first step, the case of a spacecraft without symmetries is analyzed to introduce notations for the labeling. The sampling of the attitude space with a step $\Delta\alpha$ produces M attitude classes $R_1, ..., R_M$. Rather than a one-hot representation, an attitude R is encoded by a soft labels vector

$$L(R) = \begin{pmatrix} l_1 \\ ... \\ l_M \end{pmatrix}\,, \tag{8}$$

where

$$l_i = K \exp\left(-\frac{d(R, R_i)^2}{2\sigma^2}\right)\,, \tag{9}$$

and K is set such that $\sum_{i=1}^{M} l_i = 1$. A high value l_i indicates that the attitude R is close to R_i. This definition is similar to Proença and Gao [38], but with a different standard deviation σ. We set σ proportional to the angular resolution of the sampling $\Delta\alpha$, such that $3\sigma = 2\Delta\alpha$.

Equations (8) and (9) enable to encode an attitude matrix R into a labels vector L. In a second step, a decoding function should compute a rotation matrix $\widetilde{R}$ from a labels vector L. The decoding function is denoted by Γ. A naive idea would be to assume that the attitude matrix can be estimated as the weighted average of the attitude samples $R_1, ..., R_M$ weighted by the labels $l_1, ..., l_M$. However, the result $\sum_{i=1}^{M} l_i R_i$ is of course not a valid rotation matrix. Therefore, a solution consists in enforcing this matrix to be a valid rotation matrix by projecting it on the attitude space $SO(3)$, similarly to [37].

Consider a 3×3 matrix M with Singular Value Decomposition (SVD) $M = U\Sigma V^T$. The orthogonal projection of this matrix on the rotation group $SO(3)$ is

$$\mathrm{Proj}(M) = UV^T\,. \tag{10}$$

Hence, the decoding function is

$$\widetilde{R} = \Gamma(l_1, ..., l_M) = \mathrm{Proj}(\sum_{i=1}^{M} l_i R_i)\,. \tag{11}$$

3. *Justification of the choice of the labeling function*

Ideally, L would be the exact inverse of Γ, meaning $\Gamma^{-1} = L$. This does not hold, but the error of the transformation chain is small. We will here present a short experimental justification of this choice of the labeling function and of the standard deviation σ. Two strategies are compared for defining L, the linear weighting strategy presented by Sharma and D'Amico [37], and the exponential weighting of (9).

We are interested in the magnitude of the error of the transformation chain when converting an attitude R to a labels vector, and then back to an attitude matrix. To measure the error, we introduce the usual angular distance function d between two rotations $R, Q \in SO(3)$. It is defined by

$$d(R, Q) = \arccos\left(\frac{\mathrm{tr}\,(R^T Q) - 1}{2}\right)\,. \tag{12}$$

The result of the transformation chain is $\widetilde{R} = \Gamma(L(R))$, while the original rotation matrix is R. The transformation error is hence defined by

$$\epsilon_{\text{trafo}}(R) = d(R, \widetilde{R})\,. \tag{13}$$

Using a Monte-Carlo approach, random attitudes R are generated, before the mean angular error of the transformation chain is computed. The results of this experimentation are presented in Fig. 4. The Monte-Carlo evaluation highlights that for all evaluated angular resolutions of the sampling $\Delta\alpha$, exponential weights are a better approximation of Γ^{-1} than linear weights. In addition, a standard deviation close to $\sigma = \frac{2}{3}\Delta\alpha$ leads to a consistently low error of the transformation ϵ_{trafo} for all sampling resolutions $\Delta\alpha$.

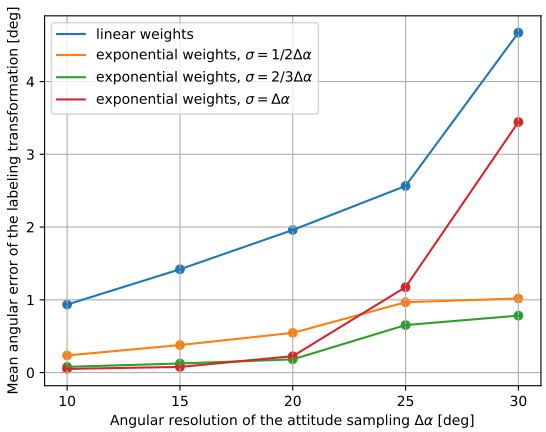


Fig. 4: Monte-Carlo evaluation of the mean angular error $\epsilon_{\text{trafo}}(R)$ for different labeling strategies and different angular resolutions $\Delta\alpha$ of the attitude sampling. Each data point was generated with 10,000 random attitudes.

For example, on Fig. 4, for a resolution $\Delta\alpha = 20$ deg, the encoding function using $\sigma = \frac{2}{3}\Delta\alpha$ has the lowest error. The mean value of the angular error ϵ_{trafo} is 0.18 deg, and the standard deviation of this error 0.10 deg.

4. *Symmetries of the considered spacecraft*

Multiple spacecrafts have a simple shape presenting possible symmetries, making an unambiguous estimation of the relative attitude impossible. Therefore, the objective of the attitude estimation will be to estimate the attitude modulo the symmetries of the spacecraft. In the case considered on Fig. 5, for each orientation of the target, three attitude candidates are considered equivalently. Indeed, the main body of the spacecraft has a hexagonal shape, but the three small bars on the edges of this hexagon are distinctive features which enable to reduce the number of symmetrical cases. This paper will discuss the handling of symmetries in this specific case, yet the proposed method is applicable to any finite number of symmetries, including a spacecraft without symmetries.

If $Q^{(1)}, Q^{(2)}, Q^{(3)}$ are the three rotations which equivalently represent the attitude Q of the symmetrical spacecraft, the distance function is modified to account for the symmetries

$$d_{sym}(R, Q) = \min(d(R, Q^{(1)}), d(R, Q^{(2)}), d(R, Q^{(3)}))\,. \tag{14}$$

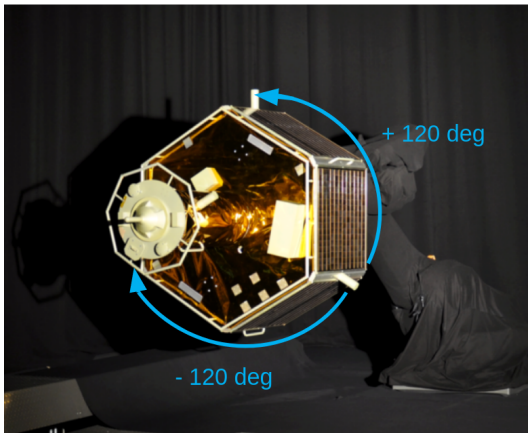


Fig. 5: Symmetries of the hexagonal spacecraft mockup at European Proximity Operations Simulator (EPOS). A rotation around the roll axis by $\pm$ 120 deg leaves the spacecraft nearly unchanged. *Image adapted from: German Aerospace Center (DLR).*

The objective is to minimize the distance d_{sym} between the true attitude and the estimation.

Because the symmetry is located on the roll axis (Fig. 5), restricting the roll angle ϕ of the estimation to $[0, \frac{2\pi}{3}]$ instead of $[0, 2\pi]$ enables to represent all attitudes without the symmetric redundancies. After removing all attitude classes amongst $R_1, ..., R_M$ which have a roll angle outside $[0, \frac{2\pi}{3}]$, the number of classes in the symmetric case is divided by three and becomes $m = M/3$.

5. *Soft classification accounting for symmetries*

Having filtered out redundant attitude classes, the sampling consists of $R_1, ..., R_m$. To ensure a correct labeling accounting for symmetric equivalents, the soft labeling function (9) becomes L_{sym} such that for $i = 1, ..., m$

$$(L_{sym}(R))_i = K_{sym} \exp\left(-\frac{d_{sym}(R, R_i)^2}{2\sigma^2}\right)\,, \tag{15}$$

where K_{sym} is the scaling constant.

The inverse function Γ is also modified to Γ_{sym} considering two observations: First, the weighted sum (11) fails to take a hard decision in the case where two or more attitudes are plausible but distant from each other. In the extreme case, if two very different classes have a weight of 0.5, the resulting estimation will be in between, which is certainly not the correct result. Thus the proposed strategy consists in first finding the index $i_{max} \in \{1, ..., m\}$ corresponding to the maximum weight, before only the attitude classes which are below an angular distance $\Delta\beta$ to the maximum weighted attitude class $R_{i_{max}}$ are kept. Considering the hexagonal shape of the spacecraft (Fig. 5), attitudes which only differ by a roll angle of ± 60 deg from the correct attitude might also have important weights because they are plausible candidates. To filter out these erroneous classifications, $\Delta\beta$ was set to 50 deg. One limitation of this method is that it takes a hard decision based only on the maximum

weight label, and not on the overall weight distribution around an attitude candidate.

In addition, two attitude classes R_i and R_j might be distant according to the classical distance function (12) but close according to the symmetric distance function (14). Hence, when adding a class in the weighted sum, care has to be taken to replace this attitude class R_j by the one of its symmetric equivalents $R_j^{(1)}, R_j^{(2)}, R_j^{(3)}$ which is closest to $R_{i_{max}}$. The complete procedure to obtain an attitude estimation from a soft labels vector is summarized in Algorithm 1.

Algorithm 1 Attitude estimation from weights in case of symmetries

Require: Attitude classes $R_1, ..., R_m$, threshold $\Delta\beta$.
  function $\Gamma_{sym}(l_1, ..., l_m)$
    Set i_{max} such that $\forall i, l_{i_{max}} \geq l_i$.
    $J \leftarrow \{i \in \{1, ..., m\} \mid d_{sym}(R_i, R_{i_{max}}) < \Delta\beta\}$.
    for all $j \in J$ **do**
      Compute symmetric attitudes $R_j^{(1)}, R_j^{(2)}, R_j^{(3)}$.
      Set $k_j \in \{1, 2, 3\}$ such that
        $d(R_{i_{max}}, R_j^{(k_j)}) = d_{sym}(R_{i_{max}}, R_j)$.
    return $\mathrm{Proj}(\sum_{j \in J} l_j R_j^{(k_j)})$.

D. CNN architecture

The CNN architecture is based on a MobileNetV3-Large [51] backbone. After a global average pooling and a batch normalization network, the output of this CNN architecture is a feature vector of size 960. This feature vector is then passed to a double prediction head, as shown on Fig. 1. The first prediction head for soft attitude labeling is a fully connected layer which reduces the feature vector to the number m of attitude classes. "Softmax" is finally applied to this layer to ensure a scaled output. The second prediction head consists of two fully connected layers of size 128 and 3 to predict the position shift. In order to ensure a result within the bounding box, so in $[-l/2, l/2]$, the $\tanh$ activation function is applied to scale the output, before the result is multiplied by $l/2$.

During training, a dropout layer with a rate of 0.2 is added between the CNN feature vector output and the prediction heads for regularization. The MobileNetV3-Large backbone is initialized with weights obtained by training on the ImageNet dataset [52], which shortens the learning duration and improves the end accuracy compared to setting random initial weights. Yet the networks have to be fully retrained, and no direct transfer learning is possible due to the difference in the nature of the problems and of the type input data, namely depth images instead of RGB images. The only exception concerns the batch normalization layers, which were maintained constant during the training and not modified, following the recommendation of [53]. Indeed, trying to re-train the batch normalization layers leads to very unstable training.

Because the objective is to minimize the distance between the estimated position and the true position, the mean squared error loss is chosen for the position estimation. For the attitude, the cross entropy loss commonly used in classification tasks is used. The resulting loss function of the network is a weighted sum of the cost of the position estimation and of the attitude classification, with a weighting factor λ. If $(\bar{l}_1, ..., \bar{l}_m)$ and $\Delta\bar{\boldsymbol{p}}$ are the true labels and position shift, and $(l_1, ..., l_m)$ and $\Delta\boldsymbol{p}$ are the network's predictions, the loss is

$$\text{loss} = -\sum_{i=1}^{m} \bar{l}_i \log l_i + \lambda \frac{||\Delta\bar{\boldsymbol{p}} - \Delta\boldsymbol{p}||^2}{3}. \qquad (16)$$

While the attitude of the input pose takes continuous values in the attitude space, the attitude predicted by the proposed architecture is discontinuous. Indeed, the strategy to avoid symmetric ambiguities in the predicted attitude leads to a prediction with a roll angle constrained to $[0, 2\pi/3]$. This means that the output is discontinuous in the attitude space. Yet the labeling function L_{sym} is continuous, so that the function that the CNN has to learn, as well as the loss, are continuous. This continuity is essential to facilitate the training process of the network and ensure good results. One strength of the proposed architecture resides in isolating the symmetry-induced discontinuities of the attitude prediction function in the last part of the computation, namely the function Γ_{sym} which transforms the continuous labels into a discontinuous attitude estimate.

E. Pose refinement

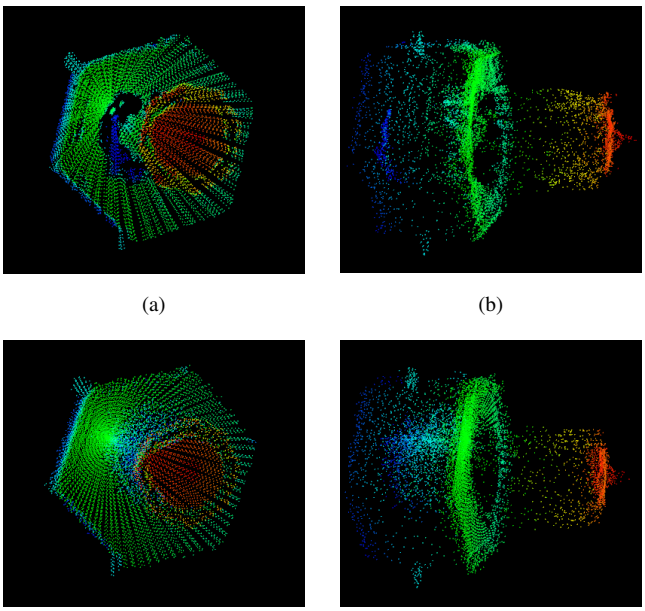
The pose estimation architecture provides an initial estimate, which is refined in a second step using a local optimization method. For this step, the 3D source point cloud is compared to a reference point cloud created from the 3D model of the target satellite. Instead of applying ICP, we propose the use of a robust and efficient registration method based on the NDT algorithm that we introduced in [17]. We showed that for this use case, this modified NDT algorithm exhibits better precision and robustness than ICP, alongside with an acceleration of the runtime by approximately a factor 5.

IV. DATASETS

A. High-fidelity lidar simulator

Neural networks are very powerful tools in computer vision tasks. They rely on important amount of data in order to be trained properly, and to be able to generalize to unseen data. In order to develop new methods quickly with sufficient data, and because real datasets are usually not available prior to the mission, the proposed approached is to train the dataset solely on simulated data. To bridge the "domain gap" between real and simulated data, the use of a physically accurate lidar simulator is essential.

The lidar sensor considered in this work is a scanning lidar. Several effects are relevant when simulating a scanning lidar for space rendezvous applications. Next to range and angular errors of the scanner, a first type of noise which affects the quality of real point clouds is the effect of the laser's beam divergence. Because the beam is not perfectly directional, erroneous points appear on the point clouds at the edge of some surfaces. This effect is observed on lidar scans taken by the Livox® Mid-40 sensor at the EPOS facility, as shown on Fig. 6b: All points between the tip of the satellite's tower (in red) and the planar surface coated with MLI (in green) are artifacts produced by the laser's beam divergence.



(a) (b) (c) (d)

Fig. 6: Real point cloud taken by the Livox® Mid-40 at EPOS: **(a)** Front view **(b)** Side view.
Point cloud generated by the lidar simulator: **(c)** Front view **(d)** Side view. Warm colors indicate a closer distance to the sensor.

The materials composing a satellite (solar panels, MLI) have a high reflectivity. Hence, while reflections are often neglected in lidar simulators, they are important for space applications. This is particularly visible on Fig. 6a, where the blue "hole" in the middle of the green region is a reflection of the tip of the satellite on the planar surface of the satellite covered with MLI. Finally, because the target spacecraft might be tumbling with an angular rate which is important compared to the low scanning frequency of the sensor, lidar point clouds show motion blur [17].

Several lidar simulators are openly available [54], [55], [56], yet they are often designed for airborne or terrestrial applications. None of them includes all the previously listed effects which are relevant for space applications. Woods and Christian [57] developed a lidar simulator specifically for spacecraft pose estimation, but it only includes basic noise modelling. Therefore, a custom lidar simulator for spacecraft pose estimation tasks has been developed for this work, and implemented in C++. It aims at being efficient while physically accurate to represent the main effects observed on real point clouds, as illustrated on Fig. 6.

The lidar simulator is developed in a generic way to be able to implement different scan patterns. Because a Livox® Mid-40 lidar with a non-repetitive rosette scan pattern is used in the hardware-in-the-loop experiments, the same scan pattern was reproduced in the simulator based on the modelization of [58]. In a first step, ray casting is implemented to compute the intersection of each ray with the target. The target is represented as a uniform triangular mesh which is obtained from standard 3D processing tools. It is stored in a k-d tree structure for fast querying of portions of the scene, and the intersection is retrieved by efficient computation of the ray-box and ray-triangle intersections [59], [60].

For modeling the reflected intensity of the laser on a surface, the Phong model [61] adapted to the problem of lidar scanning is used [62]. Each surface is described by two values, a specular reflectivity k_s, and a specular exponent η. The corresponding diffuse coefficient is $k_d = 1 - k_s$. We consider an incident ray emitted from a distance r with an incidence angle θ and an intensity I_I. The angle between the reflected ray and the observer is α. The intensity viewed by an observer is

$$I_r = \frac{I_I}{r^2} \left(k_d \cos(\theta) + (1 - k_d) \cos^{\eta}(\alpha) \right) . \qquad (17)$$

For a lidar, when considering no reflection effects, the receiver is located at the same position as the emitter, so that $\alpha = 2\theta$. However, additional reflections in other directions should be considered, as they are observable on real data. A rigorous modeling of the reflections would include a ray tracing strategy to follow the path of each possible reflection of a ray, before they finally reach the emitter again. To simplify the model and allow for fast computation on a Central Processing Unit (CPU), only a single reflected ray is computed per incoming ray, with a direction chosen at random within the potential reflection cone. This is a strong simplification, but still enables to observe reflection induced artifacts on the simulated point clouds (Fig. 6c).

In detail, for highly reflective surfaces ($\eta >> 1$ and $k_s \approx 1$), the intensity distribution around the reflected ray (close to $\alpha = 0$) approximates to $I_r \approx 1 - \eta \frac{\alpha^2}{2}$. Therefore, strongly reflected intensities will be within the reflection cone defined by $\alpha \in [0, \sqrt{\frac{2}{\eta}}]$. In addition to the direct backscattered ray, a reflected ray is computed with a random direction within this reflection cone. This ray is then propagated until it hits the next surface, before being propagated the same way back until the emitter, accounting for intensity attenuation at each surface reflection. The emitter receives two echos with different intensities, one from the direct backscattered ray, and one

from the reflected ray which is further away. The depth measure for this ray corresponds to the intensity weighted average of these two distances.

Next to the material properties, the ray properties are also modeled. To simulate beam divergence, a beam is represented by an aggregation of several rays which lightly diverge from the center. The angle β of the beam divergence is defined as the angle for which the intensity reaches only $\frac{1}{e^2}$ of the intensity along the main direction. It is $\beta = 0.28$ deg in the specifications of the Livox® Mid-40. Therefore, the intensity of a ray diverging by an angle γ from the main direction is

$$I(\gamma) = I_0 \exp(-\frac{2\gamma^2}{\beta^2}), \tag{18}$$

where I_0 is the maximum intensity.

This approach and the intensity modeling is similar to [56], but instead of uniform sampling of the rays around the main direction, random sampling is chosen to achieve a similar result with less rays (only five) for maintaining efficiency. The diverged rays are randomly selected such that the angular divergence γ with respect to the main direction follows a uniform distribution in the interval $[0, \beta]$. The return intensity and depth value of each diverged ray is first computed according to the logic presented previously, accounting for material properties and potential reflections. The intensity value is further attenuated according to (18), due to the ray's divergence angle. Afterwards, the resulting depth value of the whole beam is obtained as the intensity-weighted sum of the depth obtained for each of the five rays. With this modeling, simulated point clouds (Fig. 6d) present similar beam divergence artifacts than the real point clouds.

A range error with a standard deviation of 2 cm is further added according to the sensor's specification. Finally, motion blur is modeled to reproduce the deformation of the point cloud due to the sensor's relative motion. The motion blur model assumes a constant relative velocity and angular velocity of the target during the scan.

B. Synthetic dataset

The lidar simulator is used to generate a synthetic point cloud dataset of the German Orbital Servicing Mission (DEOS) target satellite. A 3D model of the target is used as input to the simulator. Because the exact reflectivity properties of the materials on the satellite are unknown, a domain randomization approach is adopted in this work: The properties of each material are chosen at random within a certain range, so that for each simulated point cloud, the reflectivity properties of the target are slightly different. The materials of the considered spacecraft can be roughly divided in three categories: Structural elements, solar panels and golden MLI. The value range of the material properties was partly guided by previous studies [63], but also set empirically. Table I summarizes the range of values chosen for each type of material.

The integration time of the sensor is variable as well, so that the point clouds have different densities. Because of the variable integration time and the non-repetitive scan-pattern starting at a random initial state, the scan pattern for each point cloud is slightly different.

TABLE I: Range of material properties used in the lidar simulator. k_d is the diffuse coefficient, chosen randomly with a logarithmic scale. η is the specular exponent.

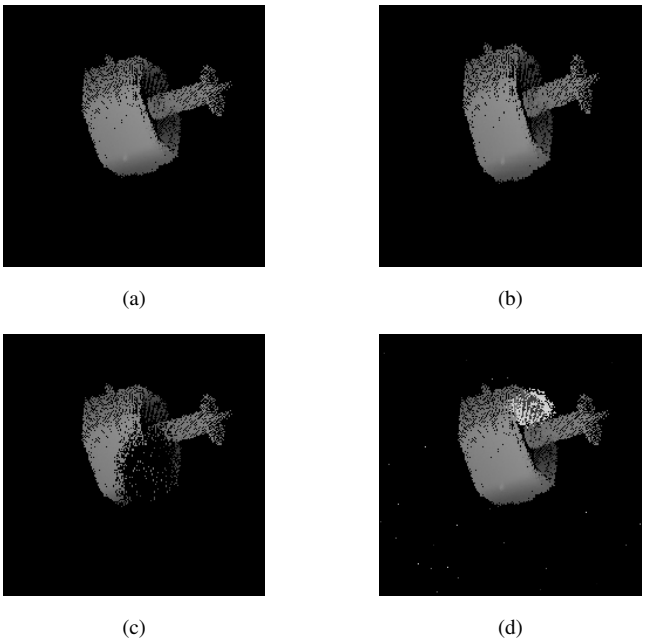
material	k_d	η
structure	[0.1, 1]	[0.5, 5]
solar panel	[1e-2, 1e-1]	[5, 20]
MLI	[1e-3, 1e-2]	[25, 75]

For this dataset, the distance of the target to the sensor is chosen randomly between 2.5 m and 20 m. At these distances, estimating the full pose of the target instead of just the relative position becomes necessary in a rendezvous mission. The position of the target in the field of view as well as the relative attitude are also randomized, as are the relative velocity and angular velocity, which are bounded to maximally 3 cm/s and 5 deg/s. The resulting synthetic dataset consists of half a million point clouds, and is split according to the 80 %-20 % rule in a training set *synthetic_train*, and a validation set *synthetic_val*. Increasing the size of the dataset consistently increases the performance of the CNN on unseen data, until a certain point where it has no effect anymore, when the size is close to half a million.

In addition to the domain randomization on the synthetic point cloud dataset, data augmentation is applied to the depth images during training to prevent overfitting. The first type of data augmentation consists in distorting the image (Fig. 7b), and enables to compensate for eventual errors between the 3D model of the target used for training and the real shape of the satellite. The other data augmentation layers were developed in this work specifically for depth images, and enable to further increase the robustness of the method. They consist of randomly overlaying black patches to mask some parts of the depth image (Fig. 7c), or adding points as well as erroneous surfaces on the depth image (Fig. 7d).

C. Hardware-in-the-loop dataset

Several datasets were collected at the hardware-in-the-loop test facility EPOS [64], located at the German Space Operations Center (GSOC) in Oberpfaffenhofen. The EPOS facility consists of two robotic arms, each representing one satellite. The first robotic arm, on the right on Fig. 8, carries a mockup of the DEOS satellite made of the real components, such as solar panels and MLI. The second robot, on the left on Fig. 8, carries the sensors of the chaser satellite, and additionally translates on a linear rail to reproduce relative distances to the target up to 25 m. The robotic facility is coupled to a dynamics simulator to reproduce space dynamics in real time, and the chaser can be commanded by a Guidance, Navigation and Control (GNC) system in closed loop.



(a) (b) (c) (d)

Fig. 7: Data augmentation: **(a)** Original depth image **(b)** Distorted image **(c)** Addition of a black patch **(d)** Addition of noise and a surface patch.

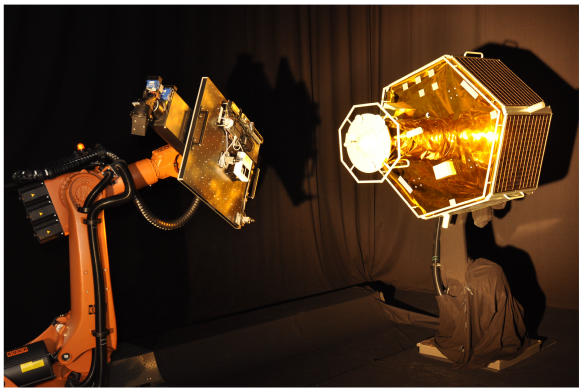


Fig. 8: Picture of the EPOS facility. A mockup of the DEOS target satellite is mounted on the right robot, while the robot on the left carries sensors of the chaser satellite. *Image source: DLR.*

The experiments were performed with a Livox® Mid-40 lidar from the automotive domain. Its overall characteristics are similar to what could be expected from a space qualified lidar, except for the very high scan rate of the sensor with 100,000 points/s. Yet, because it has a fix field-of-view close to 40 deg, most of the rays do not hit the target. All the points which belong to the background of the facility (floor, walls, robotic arm, ceiling) are removed before processing. The integration time of the lidar is set to be adaptive, so that after removing the points belonging to the background, each point cloud contains around 10,000 points, independently of the distance to the target. This behavior replicates the one of a space lidar with an adaptive field of view, which would capture approximately the same number of points on the target independently of the distance. Four point cloud datasets were collected for evaluating the pose estimation method, they are presented in Table. II. For comparing with the performance on synthetic data, a second small synthetic test dataset was generated with the lidar simulator, and is named *synthetic_test*.

TABLE II: Characteristics of the test datasets. Four of them are taken at EPOS and one is synthetic.

dataset name	target distance	number of point clouds	attitude space coverage
epos_5m	5 m	3976	10.8 %
epos_10m	10 m	1781	12.0 %
epos_15m	15 m	1253	9.9 %
epos_20m	20 m	1638	11.8 %
synthetic_test	2.5 m - 20 m	10,000	100 %

While it is possible to simulate all attitudes with the lidar simulator, this cannot be reproduced at EPOS. The facility is bound to physical constraints due to the large size of the DEOS satellite. The mockup can be rotated around its axis, towards both sides and "up" until approximately 45 deg, but not more. Therefore, the datasets recorded at EPOS do not cover the whole range of possible relative attitudes, as presented in the last column of Table. II. The attitude space coverage is computed as the fraction of attitude samples which are represented in the point cloud dataset, with respect to the total number of attitude samples.

V. RESULTS

A. Success criteria

To determine the proportion of estimates which are correct, thresholds in form of a maximum error in the position and attitude estimation are defined. The maximum acceptable error in the estimation of the position of the target's center of mass is set to 15 cm. If R is the estimated attitude and $\bar{R}$ the ground truth, the attitude error is measured as $d_{sym}(R, \bar{R})$. For defining success, the upper threshold on the attitude error is set to 5 deg.

The criteria of 15 cm and 5 deg enable to retain only pose estimates which are sufficiently precise to allow for tracking the target over consecutive point clouds. These requirements are in line with typical requirements on the precision of a GNC system for rendezvous operations: For docking with another spacecraft, Fehse [65] defines requirements on the navigation accuracy of 1 % of the relative distance and 1 deg, but states that these tolerances can be multiplied by a factor 5 in case of a robotic capture (berthing). Similarly, for the design study of the "e.Deorbit" mission [66], the initial tolerance on the spacecraft attitude estimation is of 5 deg, before being reduced to 2 deg and 5 cm for the robotic capture phase.

B. Parameter variations of the network

The CNN is trained to minimize its result on the cost function using the Adam optimizer [67] with an initial learning rate of 10^{-4}. Training is performed on the synthetic dataset *synthetic_train* with a number of epochs (number of times the complete dataset is seen by the network) set to 35. The weighting parameter λ of the loss function can be tuned. Small values of λ focus the optimization on the precision of the attitude estimation, while high values of λ lead to a more precise position estimation. When testing values of λ in the range $[1, 20]$, a good compromise is found to be $\lambda = 10$, which will be used for the rest of the evaluations.

Another parameter to be tuned is the angular resolution of the attitude sampling $\Delta\alpha$, which determines the number of attitude classes m. To select a good angular resolution, the network is trained for multiple values of $\Delta\alpha$. For this first evaluation, the training is performed without using data augmentation. The results when evaluating the network on the *synthetic_test* dataset are presented on Fig. 9.

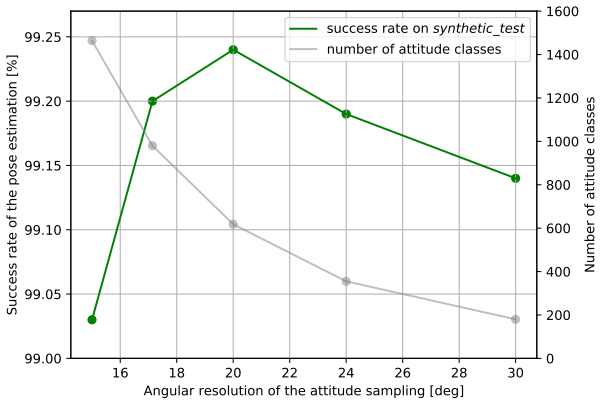


Fig. 9: Success rate of the network trained without data augmentation for different angular resolutions $\Delta\alpha$, when evaluated on the *synthetic_test* dataset. The success rate is evaluated after the pose refinement step. Additionally, the gray line shows the number of attitude classes m for each angular resolution.

The results of the angular resolution variation show that the precision of the pose estimation method initially increases when the resolution of the sampling decreases from 30 deg to 20 deg. Past this point, the network starts to overfit when the number of attitude classes further increases and the resolution is reduced to 15 deg. The number of attitude classes for each attitude sampling, represented in gray on Fig. 9, is proportional to $1/(\Delta\alpha)^3$. Based on this evaluation, the angular resolution is set to $\Delta\alpha = 20$ deg in the following. The corresponding number of attitude classes is $m = 618$.

The smoothed NDT algorithm used for the refinement step [17] uses following parameters: Both the cell size of the NDT grid, and the maximum point-to-cell distance, are set to 7.5 cm. In a pre-processing step, the point cloud is down-sampled with a voxel grid filter of resolution 2 cm. The iterative algorithm terminates when the increment is below 1 mm and 0.05 deg, or when the number of iterations reaches 30.

C. Results on test datasets

Having fixed the network, it is trained on the full synthetic training set with data augmentation. The method is then evaluated on the real point cloud datasets taken at EPOS, as well as on the synthetic test set for comparison. The detailed results of the pose estimation before and after the refinement step when evaluated on the test datasets are presented in Table. III.

On all test datasets, the success rate of the pose initialization before the refinement step is above 91 %. The NDT pose refinement helps to correct initial estimates and improve the overall precision of the method. After refinement, the success rate increases to above 96 % for all datasets. The refinement step also adds an important gain in the angular precision of the pose estimation: After refinement, the average angular error on each of the four EPOS datasets is below 1 deg in case of success. The average position errors on the EPOS datasets are in the range 5-8 cm.

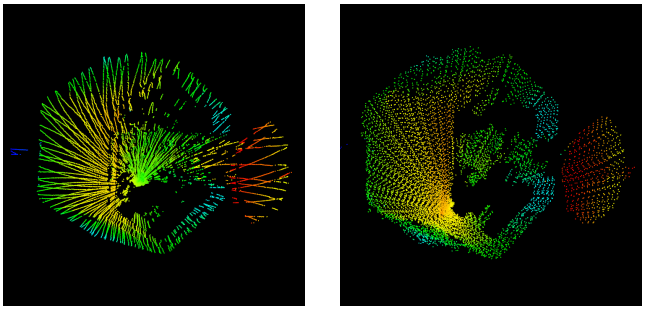
While the refinement step increases the success rate, the mean error of the position estimation slightly increases after refinement. At the same time, the standard of this error decreases. The refined position estimation is less precise in average but a more consistent estimator than the initial guess. This might indicate some systematic biases in the position estimation. The precision of the position estimation is limited by errors originating from the test environment (calibration and sensor errors), as well as by discrepancies between the 3D model of the target used for NDT matching and the real mockup installed at the facility.

An interesting point is that the performance of the pose estimation on the EPOS datasets decreases with the distance of the chaser to the target: While the success rate is of 99.9 % at 20 m, it is only of 96.2 % at 5 m distance, even though the network was trained for point clouds simulated at distances sampled uniformly in the range 2.5-20 m. This is due to the effect of the scanning pattern of the Livox® Mid-40, which scans more points in the middle of the field of view, and less on the sides. Therefore, in close range, areas of the target which are not in the center of the field of view will be scanned more sparsely, as illustrated on Fig. 10. This leads to fewer details being visible on the point cloud for pose estimation. This effect at short distances is a consequence of setting the lidar integration time to be adaptive, so that all point clouds contain approximately the same number of points, independently of the target's distance to the sensor.

Overall, on the four joint EPOS datasets, the success rate of the pose estimation is of 97.9 %, with an aver-

TABLE III: Results of the pose estimation on the test datasets before and after the refinement step. Values for the position and angular errors are shown as mean ± standard deviation. These are computed based on the successful estimates only. The dataset *epos_overall* represents the four aggregated EPOS datasets.

dataset name	intial success [%] ($\leq$ 5 deg and 15 cm)	mean initial pos. error [cm]	mean initial ang. error [deg]	refined success [%] ($\leq$ 5 deg and 15 cm)	mean refined pos. error [cm]	mean refined ang. error [deg]
epos_5m	91.55	6.63 ± 1.77	2.26 ± 0.97	96.15	6.91 ± 0.54	0.67 ± 0.24
epos_10m	97.59	5.19 ± 2.12	2.03 ± 0.87	98.93	5.83 ± 0.44	0.73 ± 0.23
epos_15m	98.32	5.93 ± 1.30	1.88 ± 0.80	99.60	5.41 ± 0.56	0.86 ± 0.27
epos_20m	97.86	5.53 ± 1.42	2.51 ± 0.98	99.94	7.82 ± 0.80	0.99 ± 0.40
epos_overall	94.97	6.00 ± 1.83	2.20 ± 0.95	97.94	6.64 ± 1.00	0.77 ± 0.31
synthetic_test	98.45	3.28 ± 1.71	2.18 ± 0.99	99.24	2.49 ± 1.47	1.89 ± 1.08



(a) (b)

Fig. 10: Point clouds taken at the EPOS facility at a distance of **(a)** 5 m and **(b)** 20 m. Both point clouds contain approximately the same number of points, but the distribution of points on the target is more uniform at 20 m than at 5 m distance.

age position and angular error of 6.6 cm and 0.77 deg, respectively. This shows that the method is precise and reliable when evaluated on real point clouds. Also, the performance on the *synthetic_test* dataset is comparable to the performance on the real datasets, highlighting that the network was able to generalize to real data. The most noticeable difference is a lower position error, as the synthetic data is not affected by calibration or 3D model errors. The final angular error is also slightly higher, due to the fact that the synthetic point clouds were simulated with high levels of motion blur (angular rates between 0 deg and 5 deg).

D. Runtime evaluation

The lidar pose estimation is developed to be run in real-time on the CPU of an onboard computer. If the exact type of onboard computer might vary from mission to mission, a representative platform can be the ARM-based Xilinx® Zynq 7000. Such System-on-Chip (SoC) devices are gaining attention for hybrid and reconfigurable onboard computing due to their low cost and direct availability on the market [18]. The Zynq 7000 is also integrated in the ScOSA onboard computer developed by DLR [19], hence it might be available in orbit. Only one CPU core of the Zynq platform is used for the runtime evaluation. For comparison, the runtime was also evaluated on one core of a standard desktop computer platform, in this case an x64-based Intel® Xeon W-2135 CPU. The total runtimes, including the pose refinement step, are presented in Table. IV. The pre-processing comprises the trimmed centroid estimation and image projection. The pose estimation method is implemented on C++, using the TensorFlow Lite library for inference of the neural network.

TABLE IV: Runtime of the pose estimation method on different platforms. Values are in milliseconds and shown as mean ± the standard deviation.

platform (CPU)	pre-proc.	CNN inference	NDT refinement	total
Intel® Xeon	1 ± 0	33 ± 3	17 ± 4	52 ± 6
Xilinx® Zynq	24 ± 1	737 ± 0	347 ± 101	1108 ± 101

The first striking observation is that the runtime of both the CNN inference and the NDT refinement step is 20 to 25 times higher on the Zynq onboard computer than on the standard desktop computer. This stresses again that efficiency is a highly important criteria in the implementation of real-time onboard software for space applications. In total, the average runtime of the pose estimation method on the onboard computer is slightly more than 1 s, which is still acceptable for a real-time pose estimation task. This processing time can be compared to the expected scan frequency of a space lidar sensor, which would be below 1 Hz. In addition, the initial estimation by the CNN does not necessarily need to be run for each point cloud, but only when a re-initialization of the pose is desired. In the other cases, the previous pose estimation can be used as an initial estimate to perform pose tracking using the NDT algorithm [17].

E. Trade-off between runtime and accuracy

The choice of MobileNetV3-Large as a backbone is the result of a trade-off between precision and runtime. For comparison, two other state-of-the-art CNN backbones were assessed. The first one, MobileNetV3-Small [51], is a small version of the MobileNetV3 architecture. The second backbone, EfficientNetV2-B0 [68], is larger

than the baseline, but is the smallest version of the EfficientNetV2 family of networks. These two backbones were inserted in the pose estimation architecture in the same way than the baseline MobileNetV3-Large. One difference is that the MobileNetV3-Small architecture produces a feature vector of size 576, and EfficientNetV2-B0 of size 1280, while the feature vector of the baseline has a size of 960. In addition, the training settings for EfficientNetV2 were slightly adapted compared to the MobileNetV3 models. For EfficientNetV2, better results were obtained by setting a dropout rate of 0.4 and re-training the batch normalization layers.

The accuracy of the pose estimation pipeline for each of the three backbones is compared by evaluating the total success rate on the four aggregated EPOS datasets. This success rate is compared to the runtime of running an inference of the CNN on the flight representative computing hardware, as presented on Table V.

TABLE V: Inference time of different CNNs on the Xilinx® Zynq, and success rate after refinement on the joint EPOS test datasets when using these CNNs as a backbone. Inference times are shown as mean ± standard deviation.

CNN backbone	inference time on Zynq [ms]	refinement success [%] ($\leq$ 5 deg, $\leq$ 15 cm)
MobileNetV3-S	248 ± 0	92.39
MobileNetV3-L	737 ± 0	97.94
EfficientNetV2	2232 ± 1	99.28

Table V shows that the more deep the CNN backbone is, the better the accuracy of the method is. With an overall success rate of above 99 % on the EPOS test datasets, the version using the EfficientNetV2 backbone outperforms the two others. However, this higher accuracy comes at the cost of a higher runtime of more than 2 s on the Zynq platform. Such a runtime can be critical for real-time operations. On the contrary, switching to the small version of MobileNetV3 leads to a significantly faster inference time of only 248 ms in average, but at the cost of an important loss in the overall accuracy, with a score of 92.4 %. In between those two models, the large version of MobileNetV3 shows to be a good trade-off between runtime and accuracy, with a runtime that is still acceptable for real-time operations.

F. Ablation study

An ablation study is performed to analyze the effect of both the data augmentation technique, and the use of the trimmed centroid. Table VI shows the overall success rate of the baseline architecture using MobileNetV3-Large when trained with, and without data augmentation (second line). The baseline model uses the trimmed centroid estimation (2) during pre-processing. To evaluate the effect of this pre-processing, we also train and test a model using a regular centroid estimation, i.e a direct average of all points' positions, to center the point cloud and select the ROI. This model is presented on the third line of the table.

TABLE VI: Ablation study: The baseline model uses MobileNetV3-Large. It is compared with the same model trained without data augmentation, and a model trained and tested without using the trimmed centroid during pre-processing. It uses a regular centroid estimator instead. The results show the success rate after the NDT refinement step.

method	refinement success [%] ($\leq$ 5 deg, $\leq$ 15 cm)
baseline	97.94
no data augm.	91.57
no trimmed centroid	76.64

From Table VI, it can be observed that the use of data augmentation leads to an accuracy gain of the method of over 6 %. The lidar simulator encompasses multiple effects, and the synthetic dataset used for training is very large with half a million point clouds. Still, if no data augmentation technique is used, the domain gap is high when testing on real lidar point clouds. The proposed data augmentation effectively enables the network to generalize to real data.

Concerning the trimmed centroid, it can be seen that its usage has a significant effect on the overall accuracy. When a classical centroid is used instead, the overall success rate drops to around 76.6 %. While both the trimmed centroid and the regular centroid are only very coarse estimates of the center of mass, the trimmed version is less affected by potential outliers or reflections in the point cloud. By discarding these outliers, the trimmed centroid is a more consistent estimator across point clouds, and enables to select the ROI more reliably than the regular centroid. Centering the point cloud around the trimmed centroid before projection on the depth image is a form of normalization of the network's inputs. It is helpful at train and test time.

VI. CONCLUSION

This work introduced a CNN based pose estimation pipeline of a non-cooperative spacecraft using lidar point clouds. The strength of the architecture is the use of both the 3D information contained in the lidar point clouds for extracting the centroid and later performing pose refinement, and the projection to a 2D depth image for processing by a CNN. After the initial centroid estimation, the neural network is tasked with predicting two outputs, a soft attitude classification, and the difference between the centroid and the target's center of mass. In the case where the spacecraft presents a symmetrical shape, the attitude distance function is adapted to remove symmetric ambiguities, and reduce the number of attitude classes.

For extensive training of the neural network, a realistic lidar simulator encompassing multiple effects such as

beam divergence, motion blur and reflections was developed. The synthetic training dataset is randomized with different material properties of the target. In addition, several data augmentation layers for depth images are proposed to make the network more robust. The pose estimation method is tested on real lidar datasets taken at a hardware-in-the-loop facility. The results of this evaluation demonstrate that the method is precise and reliable, and that the network trained on synthetic data can successfully generalize to real lidar point clouds.

A runtime measurement of the method is also performed on flight representative computing hardware, showing that the method is adapted for real time onboard implementation with a total average runtime slightly above 1 s. The runtime evaluation also highlights how varying the execution time can be from one platform to another. If multiple neural networks are planned to be run onboard during a mission, dedicated AI computing hardware is surely to be considered.

The proposed pose estimation method consists in applying a 2D CNN to a depth image obtained from a lidar point cloud. Future research might explore processing the 3D point clouds without this projection step. For this purpose, the use of neural networks designed for directly processing 3D point clouds [45], [46] could be studied. Separately, a quantitative evaluation of the lidar and target material properties could be performed to improve the realism of the lidar simulator.

REFERENCES

[1] R. B. Friend, "Orbital Express program summary and mission overview," in *Sensors and Systems for space applications II*, vol. 6958. SPIE, 2008, pp. 11–21, https://doi.org/10.1117/12.783792.

[2] M. Pyrak and J. Anderson, "Performance of Northrop Grumman's Mission Extension Vehicle (MEV) RPO Imagers at GEO," in *Autonomous systems: Sensors, processing and security for ground, air, sea and space vehicles and infrastructure 2022*, vol. 12115. SPIE, 2022, pp. 64–82, https://doi.org/10.1117/12.2631524.

[3] U. S. S. Force, "Geosynchronous Space Situational Awareness Program," [Available online]: https://www.spaceforce.mil/About-Us/Fact-Sheets/Article/2197772/geosynchronous-space-situational-awareness-program/, 2020.

[4] R. Biesbroek, S. Aziz, A. Wolahan, S.-f. Cipolla, M. Richard-Noca, and L. Piguet, "The Clearspace-1 Mission: ESA and Clearspace Team up to Remove Debris," in *Proc. 8th Eur. Conf. Sp. Debris*, 2021, pp. 1–3.

[5] L. P. Cassinis, R. Fonod, and E. Gill, "Review of the Robustness and Applicability of Monocular Pose Estimation Systems for Relative Navigation with an Uncooperative Spacecraft," *Progress in Aerospace Sciences*, vol. 110, p. 100548, 2019, https://doi.org/10.1016/j.paerosci.2019.05.008.

[6] L. Liu, G. Zhao, and Y. Bo, "Point Cloud Based Relative Pose Estimation of a Satellite in Close Range," *Sensors*, vol. 16, no. 6, p. 824, 2016, https://doi.org/10.3390/s16060824.

[7] K. Klionovska and H. Benninghoff, "Initial Pose Estimation using PMD Sensor during the Rendezvous Phase in On-Orbit Servicing Missions," in *27th AIAA/AAS Space Flight Mechanics Meeting*, 2017.

[8] F. Amzajerdian, V. E. Roback, A. Bulyshev, P. F. Brewster, and G. D. Hines, "Imaging Flash Lidar for Autonomous Safe Landing and Spacecraft Proximity Operation," in *AIAA SPACE 2016*, 2016, p. 5591, https://doi.org/10.2514/6.2016-5591.

[9] A. B. Dietrich and J. W. McMahon, "Filter Initialization with Three-Dimensional Lidar Images in Proximity to Small Bodies," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 2, pp. 310–318, 2020, https://doi.org/10.2514/1.G004468.

[10] J. M. Leonard, M. C. Moreau, P. G. Antreasian, K. M. Getzandanner, E. Church, C. Miller, M. G. Daly, O. S. Barnouin, and D. S. Lauretta, "Cross-Calibration of GNC and OLA LIDAR Systems Onboard OSIRIS-REx," in *Proceedings of the 44th Annual American Astronautical Society Guidance, Navigation, and Control Conference, 2022*. Springer, 2022, pp. 1585–1607, https://doi.org/10.1007/978-3-031-51928-4_88.

[11] J. O. Woods and J. A. Christian, "Lidar-based relative navigation with respect to non-cooperative objects," *Acta Astronautica*, vol. 126, pp. 298–311, 2016, https://doi.org/10.1016/j.actaastro.2016.05.007.

[12] S. Ruel, T. Luu, and A. Berube, "Space shuttle testing of the TriDAR 3D rendezvous and docking sensor," *Journal of Field robotics*, vol. 29, no. 4, pp. 535–553, 2012, https://doi.org/10.1002/rob.20420.

[13] F. Yin, W. Chou, Y. Wu, G. Yang, and S. Xu, "Sparse Unorganized Point Cloud Based Relative Pose Estimation for Uncooperative Space Target," *Sensors*, vol. 18, no. 4, p. 1009, 2018, https://doi.org/10.3390/s18041009.

[14] R. Opromolla, G. Fasano, G. Rufino, and M. Grassi, "A Model-Based 3D Template Matching Technique for Pose Acquisition of an Uncooperative Space Object," *Sensors*, vol. 15, no. 3, pp. 6360–6382, 2015, https://doi.org/10.3390/s150306360.

[15] ——, "A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations," *Progress in Aerospace Sciences*, vol. 93, pp. 53–72, 2017, https://doi.org/10.1016/j.paerosci.2017.07.001.

[16] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. SPIE, 1992, pp. 586–606, https://doi.org/10.1109/34.121791.

[17] L. Renaut, H. Frei, and A. Nüchter, "Lidar Pose Tracking of a Tumbling Spacecraft Using the Smoothed Normal Distribution Transform," *Remote Sensing*, vol. 15, no. 9, p. 2286, 2023, https://doi.org/10.3390/rs15092286.

[18] A. D. George and C. M. Wilson, "Onboard Processing With Hybrid and Reconfigurable Computing on Small Satellites," *Proceedings of the IEEE*, vol. 106, no. 3, pp. 458–470, 2018, https://doi.org/10.1109/JPROC.2018.2802438.

[19] D. Lüdtke, T. Firchau, C. G. Cortes, A. Lund, A. M. Nepal, M. M. Elbarrawy, Z. H. Hammadeh, J.-G. Meß, P. Kenny, F. Brömer *et al.*, "ScOSA on the Way to Orbit: Reconfigurable High-Performance Computing for Spacecraft," in *2023 IEEE Space Computing Conference (SCC)*. IEEE, 2023, pp. 34–44, https://doi.org/10.1109/SCC57168.2023.00015.

[20] S. D'Amico, M. Benn, and J. L. Jørgensen, "Pose estimation of an uncooperative spacecraft from actual space imagery," *International Journal of Space Science and Engineering 5*, vol. 2, no. 2, pp. 171–189, 2014, https://doi.org/10.1504/IJSPACESE.2014.060600.

[21] V. Capuano, S. R. Alimo, A. Q. Ho, and S.-J. Chung, "Robust Features Extraction for On-board Monocular-based Spacecraft Pose Acquisition," in *AIAA Scitech 2019 Forum*, 2019, p. 2005, https://doi.org/10.2514/6.2019-2005.

[22] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. M. Kwok, "A Comprehensive Performance Evaluation of 3D Local Feature Descriptors," *International Journal of Computer Vision*, vol. 116, pp. 66–89, 2016, https://doi.org/10.1007/s11263-015-0824-y.

[23] L. Hu, D. Sun, H. Duan, A. Shu, S. Zhou, and H. Pei, "Non-Cooperative Spacecraft Pose Measurement with Binocular Camera and TOF Camera Collaboration," *Applied Sci-*

ences, vol. 13, no. 3, p. 1420, 2023, https://doi.org/10.3390/app13031420.

[24] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model Globally, Match Locally: Efficient and Robust 3D Object Recognition," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 998–1005, https://doi.org/10.1109/CVPR.2010.5540108.

[25] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-ICP: A Globally Optimal Solution to 3D ICP Point-Set Registration," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 11, pp. 2241–2254, 2015, https://doi.org/10.48550/arXiv.1605.03344.

[26] P. Biber and W. Straßer, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 3. IEEE, 2003, pp. 2743–2748, https://doi.org/10.1109/IROS.2003.1249285.

[27] K. Klionovska and M. Burri, "Hardware-in-the-Loop Simulations with Umbra Conditions for Spacecraft Rendezvous with PMD Visual Sensors," *Sensors*, vol. 21, no. 4, p. 1455, 2021, https://doi.org/10.3390/s21041455.

[28] J. Li, Y. Zhuang, Q. Peng, and L. Zhao, "Pose Estimation of Non-Cooperative Space Targets Based on Cross-Source Point Cloud Fusion," *Remote Sensing*, vol. 13, no. 21, p. 4239, 2021, https://doi.org/10.3390/rs13214239.

[29] A. Nocerino, R. Opromolla, G. Fasano, M. Grassi, P. F. Balaguer, S. John, H. Cho, and R. Bevilacqua, "Experimental validation of inertia parameters and attitude estimation of uncooperative space targets using solid state LIDAR," *Acta Astronautica*, vol. 210, pp. 428–436, 2023, https://doi.org/10.1016/j.actaastro.2023.02.010.

[30] R. Opromolla, G. Fasano, G. Rufino, and M. Grassi, "Pose Estimation for Spacecraft Relative Navigation Using Model-Based Algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 1, pp. 431–447, 2017, https://doi.org/10.1109/TAES.2017.2650785.

[31] W. Guo, W. Hu, C. Liu, and T. Lu, "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, https://doi.org/10.2514/1.G005042.

[32] B. Chen, J. Cao, A. Parra, and T.-J. Chin, "Satellite Pose Estimation with Deep Landmark Regression and Nonlinear Pose Refinement," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0, https://doi.org/10.48550/arXiv.1908.11542.

[33] Y. Hu, S. Speierer, W. Jakob, P. Fua, and M. Salzmann, "Wide-Depth-Range 6D Object Pose Estimation in Space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 870–15 879, https://doi.org/10.48550/arXiv.2104.00337.

[34] K. Cosmas and A. Kenichi, "Utilization of FPGA for Onboard Inference of Landmark Localization in CNN-Based Spacecraft Pose Estimation," *Aerospace*, vol. 7, no. 11, p. 159, 2020, https://doi.org/10.3390/aerospace7110159.

[35] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520, https://doi.org/10.48550/arXiv.1801.04381.

[36] K. Black, S. Shankar, D. Fonseka, J. Deutsch, A. Dhir, and M. R. Akella, "Real-Time, Flight-Ready, Non-Cooperative Spacecraft Pose Estimation Using Monocular Imagery," in *31st AIAA/AAS Space Flight Mechanics Meeting*, 2021, https://doi.org/10.48550/arXiv.2101.09553.

[37] S. Sharma and S. D'Amico, "Neural Network-Based Pose Estimation for Noncooperative Spacecraft Rendezvous," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 6, pp. 4638–4658, 2020, https://doi.org/10.1109/TAES.2020.2999148.

[38] P. F. Proença and Y. Gao, "Deep Learning for Spacecraft Pose Estimation from Photorealistic Rendering," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6007–6013, https://doi.org/10.1109/ICRA40945.2020.9197244.

[39] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114, https://doi.org/10.48550/arXiv.1905.11946.

[40] T. H. Park and S. D'Amico, "Robust Multi-Task Learning and Online Refinement for Spacecraft Pose Estimation across Domain Gap," *Advances in Space Research*, 2023, https://doi.org/10.48550/arXiv.2203.04275.

[41] S. Sharma and S. D'Amico, "Pose Estimation for Non-Cooperative Rendezvous Using Neural Networks," in *29th AIAA/AAS Space Flight Mechanics Meeting*, 2019, https://doi.org/10.48550/arXiv.1906.09868.

[42] M. A. Mohamed Ali, A. Rathinam, V. Gaudilliere, M. Ortiz Del Castillo, and D. Aouada, "CubeSat-CDT: A Cross-Domain Dataset for 6-DoF Trajectory Estimation of a Symmetric Spacecraft," in *Proceedings of the 17th European Conference on Computer Vision Workshops (ECCVW 2022)*, 2022, https://doi.org/10.1007/978-3-031-25056-9_8.

[43] L. P. Cassinis, A. Menicucci, E. Gill, I. Ahrns, and M. Sanchez-Gestido, "On-Ground Validation of a CNN-based Monocular Pose Estimation System for Uncooperative Spacecraft: Bridging Domain Shift in Rendezvous Scenarios," *Acta Astronautica*, vol. 196, pp. 123–138, 2022, https://doi.org/10.1016/j.actaastro.2022.04.002.

[44] K. M. Kajak, C. Maddock, H. Frei, and K. Schwenk, "Domain randomisation and CNN-based keypoint-regressing pose initialisation for relative navigation with uncooperative finite-symmetric spacecraft targets using monocular camera images," *Advances in Space Research*, 2023, https://doi.org/10.1016/j.asr.2023.02.024.

[45] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660, https://doi.org/10.48550/arXiv.1612.00593.

[46] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic Graph CNN for Learning on Point Clouds," *ACM Transactions on Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019, https://doi.org/10.48550/arXiv.1801.07829.

[47] O. Kechagias-Stamatis, N. Aouf, V. Dubanchet, and M. A. Richardson, "DeepLO: Multi-projection deep LIDAR odometry for space orbital robotics rendezvous relative navigation," *Acta Astronautica*, vol. 177, pp. 270–285, 2020, https://doi.org/10.1016/j.actaastro.2020.07.034.

[48] Z. Chekakta, A. Zenati, N. Aouf, and O. Dubois-Matra, "Robust deep learning LiDAR-based pose estimation for autonomous space landers," *Acta Astronautica*, vol. 201, pp. 59–74, 2022, https://doi.org/10.1016/j.actaastro.2022.08.049.

[49] E. A. Pensado, L. M. G. de Santos, H. G. Jorge, and M. Sanjurjo-Rivo, "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, https://doi.org/10.1109/TAES.2023.3262505.

[50] S. T. Wong and M. S. Roos, "A strategy for sampling on a sphere applied to 3D selective RF pulse design," *Magnetic Resonance in Medicine*, vol. 32, no. 6, pp. 778–784, 1994, https://doi.org/10.1002/mrm.1910320614.

[51] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for MobileNetV3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324, https://doi.org/10.48550/arXiv.1905.02244.

[52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 248–255, https://doi.org/10.1109/CVPR.2009.5206848.

[53] F. Chollet, "Transfer learning & fine-tuning," [Available online]: https://keras.io/guides/transfer_learning/, 2020.

[54] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree, "BlenSor: Blender Sensor Simulation Toolbox," in *Advances in Visual Computing: 7th International Symposium, ISVC 2011, Las Vegas, NV, USA, September 26-28, 2011. Proceedings, Part II 7*. Springer, 2011, pp. 199–208, https://doi.org/10.1007/978-3-642-24031-7_20.

[55] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An Open Urban Driving Simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16, https://doi.org/10.48550/arXiv.1711.03938.

[56] L. Winiwarter, A. M. E. Pena, H. Weiser, K. Anders, J. M. Sánchez, M. Searle, and B. Höfle, "Virtual laser scanning with HELIOS++: A novel take on ray tracing-based simulation of topographic 3D laser scanning," *Remote Sensing of Environment*, vol. 269, p. 112772, 2022, https://doi.org/10.1016/j.rse.2021.112772.

[57] J. O. Woods and J. A. Christian, "Glidar: An OpenGL-based, Real-Time, and Open Source 3D Sensor Simulator for Testing Computer Vision Algorithms," *Journal of Imaging*, vol. 2, no. 1, p. 5, 2016, https://doi.org/10.3390/jimaging2010005.

[58] R. G. Brazeal, B. E. Wilkinson, and H. H. Hochmair, "A Rigorous Observation Model for the Risley Prism-Based Livox Mid-40 Lidar Sensor," *Sensors*, vol. 21, no. 14, p. 4722, 2021, https://doi.org/10.3390/s21144722.

[59] A. Williams, S. Barrus, R. K. Morley, and P. Shirley, "An Efficient and Robust Ray–Box Intersection Algorithm," in *ACM SIGGRAPH 2005 Courses*, 2005, pp. 9–es, https://doi.org/10.1145/1198555.1198748.

[60] T. Möller and B. Trumbore, "Fast, Minimum Storage Ray/Triangle Intersection," in *ACM SIGGRAPH 2005 Courses*, 2005, pp. 7–es, https://doi.org/10.1145/1198555.1198746.

[61] B. T. Phong, "Illumination for Computer Generated Pictures," in *Seminal graphics: pioneering efforts that shaped the field*, 1998, pp. 95–101, https://doi.org/10.1145/360825.360839.

[62] B. Jutzi and H. Gross, "Normalization Of Lidar Intensity Data Based On Range And Surface Incidence Angle," *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci*, vol. 38, pp. 213–218, 2009, https://doi.org/10.24406/publica-fhg-362664.

[63] Y. Nakajima, T. Sasaki, N. Okada, and T. Yamamoto, "Development of LiDAR Measurement Simulator Considering Target Surface Reflection," in *Proceedings of the 8th European Conference on Space Debris, Darmstadt, Germany*, 2021, pp. 20–23.

[64] H. Benninghoff, F. Rems, E.-A. Risse, and C. Mietner, "European Proximity Operations Simulator 2.0 (EPOS) - A Robotic-Based Rendezvous and Docking Simulator ," *Journal of large-scale research facilities JLSRF*, 2017, https://doi.org/10.17815/jlsrf-3-155.

[65] W. Fehse, *Automated Rendezvous and Docking of Spacecraft*. Cambridge university press, 2003, vol. 16, https://doi.org/10.1017/CBO9780511543388.

[66] J. Telaar, I. Ahrns, S. Estable, W. Rackl, M. De Stefano, R. Lampariello, N. Santos, P. Serra, M. Canetri, F. Ankersen *et al.*, "GNC architecture for the e.Deorbit mission," in *7th European Conference for Aeronautics and Space Sciences (EUCASS)*. ESA Publications Division, ESTEC Noordwijk, The Netherlands, 2017, pp. 1–15, https://doi.org/10.13009/EUCASS2017-317.

[67] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015, https://doi.org/10.48550/arXiv.1412.6980.

[68] M. Tan and Q. Le, "EfficientNetV2: Smaller Models and Faster Training," in *International conference on machine learning*. PMLR, 2021, pp. 10 096–10 106, https://doi.org/10.48550/arXiv.2104.00298.

Léo Renaut received his M.S. degree from École Polytechnique, Palaiseau, France, in 2019. In 2020, he obtained a second M.S. from ISAE-Supaero, Toulouse, France, as part of a double degree program.

Since 2020, he has been working as a scientific employee the German Aerospace Center (DLR) in Oberpfaffenhofen, Germany. Within the "On-Orbit Servicing and Autonomy" group of the "Spaceflight Technology" department, he works on the development of GNC algorithms for on-orbit servicing and active debris removal missions. Since 2022, he is a Ph.D. candidate at the Julius-Maximilians-Universität Würzburg, Germany, on the topic of lidar navigation for autonomous space rendezvous.

Mr. Renaut's research interests include optimal trajectory design, vision and lidar-based navigation, and spaceflight dynamics.

Heike Frei graduated in mathematics in 2010 and received her Ph.D. (Dr. rer. nat.) in 2015 from the University of Regensburg, Germany.

Since 2010, she is with the German Aerospace Center (DLR) in Oberpfaffenhofen, Germany. She worked as scientific employee (2010-2015) and led the "On-Orbit Servicing and Autonomy" group (2015-2022). Since, 2023 she is the head of the department of "Spaceflight Technology" within the German Space Operations Center (GSOC) and lecturer at the University of Würzburg, Germany.

Dr. Frei's research interests comprise relative orbit and attitude determination, on-orbit servicing and active debris removal, rendezvous and inspection, optical navigation, image processing, numerical mathematics and hardware-in-the-loop simulation.

Andreas Nüchter received the diploma degree in computer science and the Ph.D. degree (Dr. rer. nat) in semantic 3D maps for autonomous mobile robots, both from the University of Bonn, Bonn, Germany, in 2002 and 2006, respectively.

He was with Fraunhofer Institute for Autonomous Intelligent Systems (AIS, Sankt Augustin), with the University of Bonn, and with the Washington State University. He was a Research Associate with University of Osnabrück. Before the summer of 2013, he was an Assistant Professor with the Automation Group, Constructor University, formerly Jacobs University Bremen, Bremen, Germany. Before the fall of 2022, he was an Associate Professor (tenured) for telematics with the University of Würzburg, Würzburg, Germany, where he is currently a Professor and Chair of robotics. He works on robotics and automation, cognitive systems, and artificial intelligence. Andreas developed fast 3-D scan matching algorithms that enable robots to perceive and map their environment in 3-D representing the pose with 6 degrees of freedom. The capabilities of these robotic SLAM approaches were demonstrated at RoboCup Rescue competitions, ELROB, and several other events.

Prof. Nüchter's main research interests include reliable robot control, 3-D environment mapping, 3-D vision, and laser scanning technologies for various applications, e.g., for planetary exploration, safety security and rescue robotics, or underwater inspection. He is a Member of the GI.