DOI: 10.13009/EUCASS2025-011
11TH EUROPEAN CONFERENCE FOR AERONAUTICS AND SPACE SCIENCES (EUCASS)

Enabling Robust, Real-Time Verification of Vision-Based Navigation through View Synthesis

Marius Neuhalfen[1,2,3†*] Jonathan Grzymisch[1] Manuel Sánchez-Gestido[1]

[1]*European Space Agency, ESTEC, The Netherlands*
[2]*RWTH Aachen University, Aachen, Germany* [3]*École Centrale de Lille, Lille, France*

[†]*Corresponding author*

June 15, 2025

Abstract

This work introduces VISY-REVE: a novel pipeline to validate image processing algorithms for Vision-Based Navigation. Traditional validation methods such as synthetic rendering or robotic testbed acquisition suffer from difficult setup and slow runtime. Instead, we propose augmenting image datasets in real-time with synthesized views at novel poses. This approach creates continuous trajectories from sparse, pre-existing datasets in open or closed-loop. In addition, we introduce a new distance metric between camera poses, the Boresight Deviation Distance, which is better suited for view synthesis than existing metrics. Using it, a method for increasing the density of image datasets is developed. Project page[1]

1. Introduction

Autonomous Guidance, Navigation and Control (GNC) technologies are becoming increasingly relevant for modern space applications. Such systems must be able to independently make decisions, plan maneuvers, and safely execute them; all without the intervention of operators. Two prominent mission profiles that need these capabilities are in-orbit close-proximity operations with human-made targets[28] (e.g., satellite servicing or debris removal) and entry, descent, and landing on natural bodies[10] (e.g., the Moon or Mars). These autonomous capabilities often rely on *vision-based navigation* (VBN),[9] where images from passive or active optical sensors provide relative state information between a target and the host spacecraft. Due to the high reliability requirements of space missions, the entire GNC stack, including the VBN, must be carefully validated on the ground before launch. This validation includes testing the image processing algorithms that are part of the VBN, ideally with hardware-in-the-loop and in real-time.[12]

Unfortunately, representative on-orbit imagery necessary for validation (examples include LIRIS[19] and PRISMA[3,9]) is scarce, often proprietary, and rarely accompanied by accurate ground truth. *In-situ* testing before a mission is impossible, and transferring results from previous datasets is unreliable. Therefore, VBN validation relies on surrogate imagery that approximates the in-space environment as closely as possible, ideally minimizing the domain gap.[6]

2. Related Work

Surrogate validation datasets for VBN

Currently, two surrogate image acquisition methods to generate validation datasets dominate.

The first is *synthetic rendering*. Here, software packages such as PANGU[18] or SurRender[16] aim to recreate the real environment of space and to model radiometric properties from first principles. A particular advantage of this method is that it can simulate a variety of optical effects, which is useful for Monte Carlo campaigns and robustness testing of image processing algorithms. However, creating accurate reflection models, high-fidelity 3D geometries and material textures requires a significant upfront effort. Additionally, real-time rendering of synthetic imagery remains computationally intensive[16] and therefore difficult to integrate into closed-loop testing, particularly on flight hardware.

The second uses *robotic testbeds (R.T.)* (also called *laboratories*). These are special facilities where physical spacecraft mock-ups are manipulated with robotic actuators and photographed under calibrated lighting. The advantage of this approach is that the images will include the effects of a real optical sensor, including lens distortions, noise and other artifacts. However, the space of possible poses (positions and attitudes) is inherently limited, mock-ups must be

*Contact: *neuhalfen [dot] marius [at] gmail [dot] com*. This work was carried out while the corresponding author was a trainee at ESA.
[1]Project Page Link: https://marius-ne.github.io/visyreve.github.io/

DOI: 10.13009/EUCASS2025-011

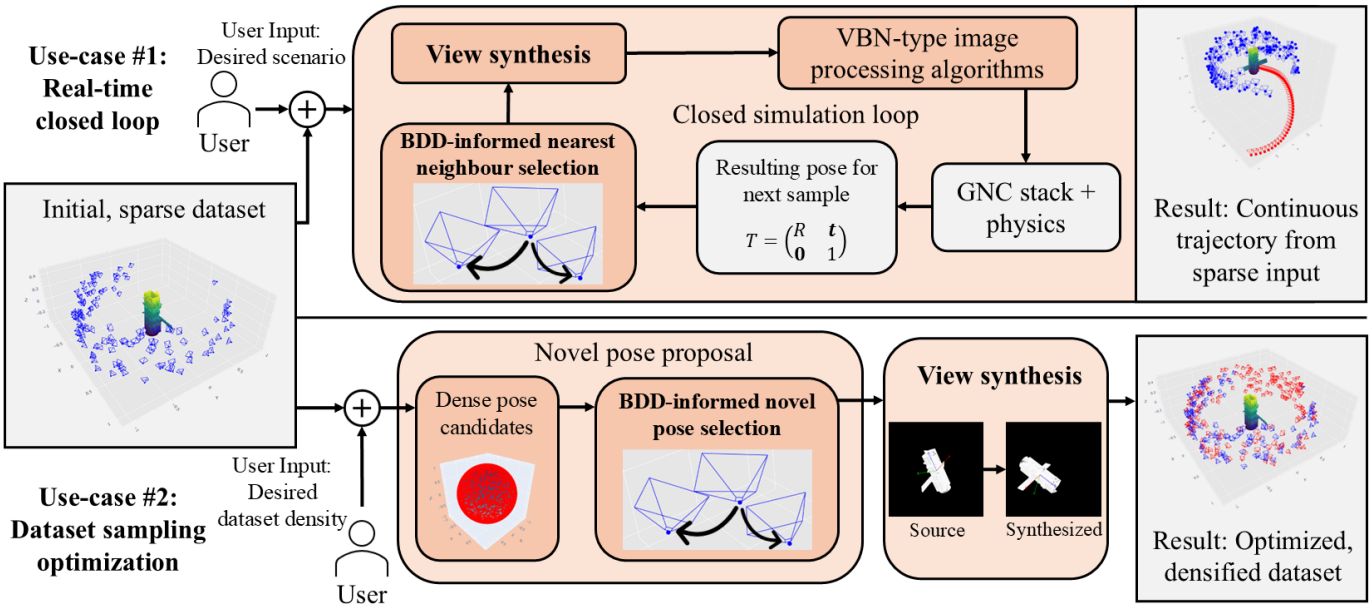


Figure 1: **Key contributions of this work**. We introduce a versatile view synthesis pipeline that fits a variety of use-cases. Two example ones are illustrated here: 1. Running closed-loop, real-time tests of VBN image processing algorithms from an existing, *sparse* dataset. 2. Optimizing pose sampling of a dataset to increase image density, note that this can also be used in advance of dataset creation to efficiently plan image acquisition campaigns.

scaled down to fit inside the facility, which leads to deviations in lighting, materials must be recreated or approximated, and each campaign requires careful re-calibration and integration with facility actuators and processors.[5,7]

Combining synthetic and R.T. methods is advantageous as it has been shown that training models with mixed imagery from both domains increases the domain adaptation of the models from the surrogate to in-situ images.[28,31]

Datasets can further be divided into *trajectory-based* datasets, which follow physically feasible camera paths, and *random-pose* datasets, which randomly sample camera poses. Trajectory-based datasets are preferable for open-loop tests, but generally cover the whole image space less well than random-pose datasets without additional effort.

Our method of *view synthesis* is complementary to these surrogate image acquisition methods; it can synthesize images in real-time from an existing dataset, be it R.T. or synthetic. It can create new images similar in quality to the input images without having to perform heavy rendering or complex interfacing with a facility. This makes it possible to enhance previously acquired datasets, yielding more imagery, and enabling them to be expanded in real-time reacting to navigation results from a GNC stack, thus enabling closed-loop functional testing of image processing.

View Synthesis

The term View Synthesis (VS) refers to the process of generating novel imagery from an existing sparse set of images acquired by cameras located at different viewpoints (poses).[4] This is achieved by deriving underlying information about the scene from the input images and using this information to transform them to obtain novel views of the scene from different camera poses. We refer to the combination of an image and its associated camera pose as a *view*.

VS has been used in the space domain by[27] for self-supervised test-time adaptation of convolutional neural networks (CNN) for keypoint-based pose estimation. The authors showed that VS is beneficial to improve the CNN's ability to bridge the domain gap, that is, the difference between the aforementioned surrogate image types and *in-situ* imagery. However, the synthesized views needed to be temporally close to their source, which is not the case in our work.

In the context of validation, VS has been used in other domains such as autonomous driving, for example by VISTA,[1,22] to validate control of self-driving cars using sparsely imaged trajectories. The authors showed in a similar manner as before that training models with off-line collected sparse datasets together with in-between views synthesized by VS in real-time improves the transfer capabilities of these models to real-world driving scenarios.

Pose Distance Metrics

VS techniques often involve computing distances between camera poses. This is done, for example, when choosing an optimal source image from multiple near-neighbor candidates. The more distant (loosely defined) the novel camera pose is from the one of the chosen source view, the more difficult is the VS and the less accurate will be the result. The difficulty arises from the fact that, in general, VS is not good at synthesizing novel information.[20] The novel information in this case stems from novel parts of the scene becoming visible when the pose of the camera changes.

DOI: 10.13009/EUCASS2025-011

Thus, it becomes necessary to quantify the distance between camera poses, both to choose source views and to create performance models that indicate the limits of VS. For this purpose, multiple pairwise distance metrics exist; widely used is the Euclidean norm of the displacement between camera positions.[1,33,36]

Alternatively, one can use the magnitude of the relative rotation between two views.[17] This better encapsulates the fact that changes in the visible parts of a scene can result even from translationally static cameras. However, it ignores the fact that new parts of a scene can also become visible by pure translation of the camera.

To resolve these shortcomings, we propose a new metric to measure the distance between camera poses that is able to better predict the difficulty of view synthesis than the two previously mentioned metrics. We call it the *Boresight Deviation Distance*. Its definition will be given later on.

2.1 Our contributions

This work introduces a real-time VS pipeline that generates novel views from *a priori* available sparse image datasets with associated ground-truth pose labels. We call this pipeline **VISY-REVE** which stands for **VI**ew **SY**nthesis for **RE**al-time **V**alidation for space **E**xploration. The novel views can be used to run closed-loop testing campaigns with imagery that responds in real-time to changes in the pose of a simulated spacecraft. Additionally, it can be used off-line to enhance the image sampling of a dataset. These applications are illustrated in 1. To the best of our knowledge, our work is the first to introduce this technique to the problem of real-time validation in the space domain.

Our approach closes the gap between photorealistic but slow rendering for software-in-the-loop and real-time but difficult-to-set up R.T. acquisition for hardware-in-the-loop validation. VISY-REVE's two key contributions are:

1. **Real-time view-synthesis pipeline.** The pipeline accepts an input dataset (e.g. synthetic or robotic-testbed, with optional depth or 3-D model) and synthesizes geometrically accurate novel views at real-time speed.
2. **Boresight Deviation Distance.** A novel dimensionless distance metric between camera poses that is able to predict VS quality, agnostically compare different datasets and identify poorly sampled pose regions.

3. Conventions

Our work relies on the notions of projective geometry and related concepts from computer vision. A detailed description of these topics can be found in.[13] Here we will only give a short overview of the notation used. The projection of a 3D point given in target coordinates into an image is given as:

$$\begin{bmatrix} \hat{u} \\ \hat{v} \\ \hat{z} \end{bmatrix} = K\,[I \mid \mathbf{0}]\,T_{target2cam} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1}$$

Where both points are given in homogeneous coordinates. The 2D image coordinates can therefore be obtained by dividing both $\hat{u}$ and $\hat{v}$ by $\hat{z}$. The projection matrix is given with the extrinsics $T_{target2cam}$ and intrinsics K. Extrinsics are given as a passive target to camera rigid body transformation. The boresight (camera) axis is defined to be looking down the positive z direction. The intrinsics are based on a pinhole camera model with zero skew. Formally:

$$K = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix},\; T_{target2cam} = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2}$$

We refer to mappings between images as image transformations. These transformations operate on each pixel in the input image and map it to a given pixel in the output image. Some transformations do this smoothly across the image; in this case the transformation can be called affine or projective and can be expressed by 3×3 matrices in the homogeneous image coordinates. Other transformations are non-linear warps that act on each pixel independently.

4. Datasets

To use VS, preexisting datasets are needed. We rely exclusively on publicly available ones for the reproducibility of the achieved results. Note that this work focuses exclusively on the human-made target scenario although we acknowledge that the methods presented can also be applied to the natural target scenario.

The following datasets are chosen:

DOI: 10.13009/EUCASS2025-011

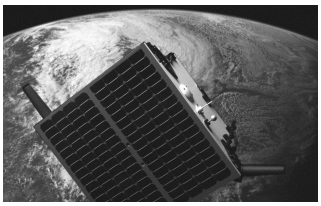


Figure 2: SPEED+

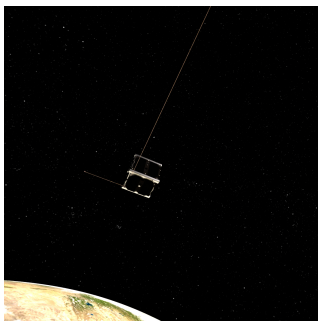


Figure 3: SWISSCUBE

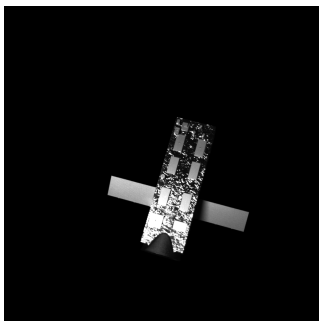


Figure 4: ESA-AIRBUS

Figure 5: Examples from the chosen datasets.

1. SPEED+ dataset (synthetic only),[26] Link
2. SWISSCUBE dataset,[14] Link
3. ESA-AIRBUS dataset (MAN-DATA-L1),[15] Link

See 5 for example images from these datasets. Below is a short table that summarizes their key characteristics.

Table 1: Comparison between the chosen datasets

Dataset	Image-type	Poses	Color	Num. of Images	Target Satellite
SPEED+	R.T. & Synthetic	Random	Grayscale	9500 (R.T.), 60.000 (S)	Tango
SWISSCUBE	Synthetic	Trajectories	RGB	43.209	Swisscube
ESA-AIRBUS	Robotic testbed	Random	Grayscale	16.000	Envisat

The ESA-AIRBUS and SWISSCUBE datasets include the 3D models of the target mock-ups used.

For the SPEED+ dataset, the 3D model was not publicly available at the time of writing. To still be able to apply model-based methods, we reconstruct a coarse 3D `.obj` model from the dataset's images and their associated ground-truth labels. We release it alongside this work[2]. This model is equally useful for other datasets that rely on the same mock-up, namely SPEED[30] and SHIRT.[23]

5. Boresight Deviation Distance

In this chapter, we introduce a novel pose distance metric with the goal of characterizing the performance of VS. First, we motivate the need for its definition intuitively (5.1) and formally (5.2). We state its definition (5.3), derive its useful properties (5.4) and use it to define a metric of sampling density of datasets (5.5).

5.1 Motivation

To enable VS, it is necessary to have a distance metric between camera poses, as explained before, see 2. This metric is used to create a performance model that encodes the "furthest" distance that VS is capable of faithfully recreating. Therefore, such a distance metric should be roughly proportional to the degradation of accuracy of a synthesized image resulting from a distance increase to its source image. If this is the case, the metric can then be used to choose nearest-neighbor source views in the first place, as these views will lead to the most accurate VS.

The distance metric usually chosen for this task is the Euclidean norm of the distance between the camera positions in $\mathbb{R}^3$, i.e. $\|\mathbf{t_i} - \mathbf{t_j}\|$ or $\|\mathbf{C}_i - \mathbf{C}_j\|$ where $\mathbf{C} = -R^{-1}\,\mathbf{t}$. This metric will be referred to as **C-L2**. In the following, we show that it is not an ideal choice for this task. Its principal flaw is the fact that it is *isotropic* w.r.t. the axes of the camera reference frame, i.e. that it treats a translation in for example the x and z axes in the same way.

To solve this, we introduce the **Boresight Deviation Distance** (BDD), a novel distance metric between poses, more specifically, their attitudes. Intuitively, the BDD measures how far the rotation axis of the relative rotation between two poses is from the camera's viewing direction (boresight axis). The basis for its definition is the simple observation that

[2] https://github.com/marius-ne/Tango3DModel

DOI: 10.13009/EUCASS2025-011

the quality of VS depends on the relative pose in an *anisotropic* manner. The relative pose is the displacement of the camera from the pose of a source view to the one of a target view. Some relative poses are easier to synthesize using VS than others. This is shown visually in 6. Although this is intuitive, we want to prove it formally.

Specifically, we investigate which image transformations depend on having precise depth information to the individual corresponding 3D points and which do not. If knowledge of pixel-wise depth information is necessary, then VS becomes more difficult, as this depth information is lost by projection, as shown in 3.

Therefore, we investigate which components of a relative pose imply an image transformation that needs precise depth information and which do not.

5.2 Proof of the anisotropy of view synthesis

Notation: In the following A (upper-case) will refer to a matrix and $\mathbf{a}$ (lower-case, bold) to a vector. Specifically, $\mathbf{x}$ will refer to a 2D image point, $\mathbf{x_{hom}}$ to its homogeneous counterpart, $\mathbf{p_C}$ to the corresponding 3D point and $\hat{(\cdot)}$ to the transformed version of any point. All 3D points and axes are given in the reference frame of the camera.

First, we will derive the expression for the image transformation associated with a pure relative rotation around the z axis of the camera (boresight axis). We expect that this transformation is possible to perfectly recreate using VS as it is simply a planar rotation of the image. We will choose an arbitrary image point $\mathbf{x} = [u, v]$ with its homogeneous counterpart defined as $\mathbf{x}_{\text{hom}} = \lambda \begin{bmatrix} u & v & 1 \end{bmatrix}^T$. This is the ray extending from the camera center $\mathbf{C}$ through the image point $\mathbf{x}$. Any point along this ray will project to the image point. This fact is represented by the scale factor λ.

The homogeneous image point is first back-projected into camera space using the inverse of the intrinsic matrix:

$$\mathbf{p}_c = \lambda\, K^{-1}\, \mathbf{x}_{\text{hom}} = \lambda \begin{bmatrix} \dfrac{u - p_x}{f_x} \\ \dfrac{v - p_y}{f_y} \\ 1 \end{bmatrix} \tag{3}$$

Then, a relative rotation of α around the boresight axis z is applied:

$$\hat{\mathbf{p}}_\mathbf{C} = R\, \mathbf{p}_\mathbf{C} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \lambda \begin{bmatrix} \dfrac{u - p_x}{f_x} \\ \dfrac{v - p_y}{f_y} \\ 1 \end{bmatrix} \tag{4}$$

$$\hat{\mathbf{x}}_\mathbf{hom} = K\, \hat{\mathbf{p}}_\mathbf{C} = K\, R\, \lambda \begin{bmatrix} \dfrac{u - p_x}{f_x} \\ \dfrac{v - p_y}{f_y} \\ 1 \end{bmatrix} \tag{5}$$

The rotated 3D point in the camera space is then projected into the target image:

$$\hat{\mathbf{x}}_\mathbf{hom} = \lambda \begin{bmatrix} f_x(\cos\alpha\, \bar{u} - \sin\alpha\, \bar{v}) + p_x \\ f_y(\sin\alpha\, \bar{u} + \cos\alpha\, \bar{v}) + p_y \\ 1 \end{bmatrix} = \lambda\, A\, \mathbf{x}_\mathbf{hom} \tag{6}$$

$$\bar{u} = \frac{u - p_x}{f_x}, \quad \bar{v} = \frac{v - p_y}{f_y} \tag{7}$$

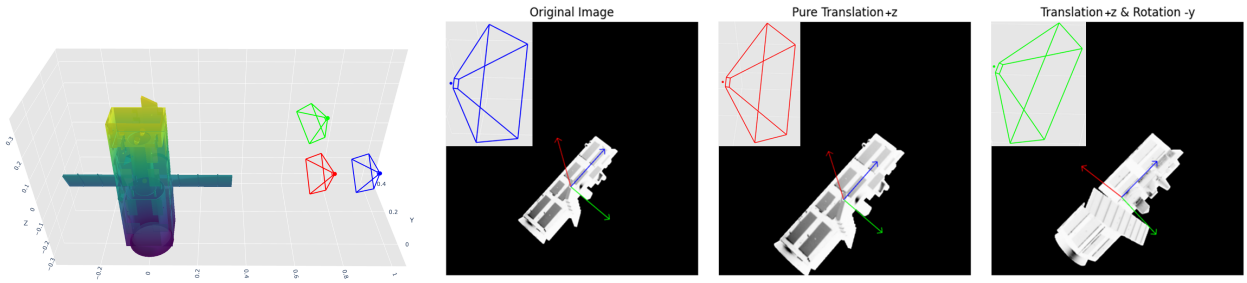
As shown, the homogeneous transformed image point can be obtained by an *affine* transform A of the original homogeneous image point. The transformed 2D image point is obtained by dividing the homogeneous point by its z coordinate, which cancels the scale factor λ.

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} = \begin{bmatrix} \cos\alpha\, u - \sin\alpha\, \dfrac{f_x}{f_y}\, v + p_x(1 - \cos\alpha) + \sin\alpha\, \dfrac{f_x}{f_y}\, p_y \\ \sin\alpha\, \dfrac{f_y}{f_x}\, u + \cos\alpha\, v + p_y(1 - \cos\alpha) - \sin\alpha\, \dfrac{f_y}{f_x}\, p_x \end{bmatrix} \iff \hat{\mathbf{x}} \neq f(\lambda) \tag{8}$$

This shows that a relative rotation exclusively around the boresight axis can be represented with an affine matrix without the need for precise depth information. Crucially, this means that it can be perfectly recreated with VS.

Next, we proceed in a similar way for a translation in the direction of z (boresight axis). For the sake of demonstration, we also include a translation in the x and y axes; in these axes, however, the translation must be proportional to the depth λ of the corresponding 3D point as we will see below. In general, the image transformation associated with an arbitrary translation in x and y is depth-dependent. Therefore, we model the translations in x and y as linearly proportional to the depth with a constant factor β, so $\Delta x = \beta_x\, \lambda$ and $\Delta y = \beta_y\, \lambda$.

DOI: 10.13009/EUCASS2025-011



(a) Camera poses from right in 3D.

(b) Images from each pose.

Figure 6: Comparison of poses with similar C-L2 but different BDD values. The green view (right) is harder to synthesize from the blue view (left) due to both translation in z and rotation in y of the camera. Compared to the green view, the red view (center) is perceptually closer to the blue view due to only being translated in z, despite both the red and green views having a similar C-L2 distance to the blue view. This anisotropy is captured by the BDD.

$$\hat{\mathbf{p}}_C = \mathbf{p}_C + \Delta\mathbf{p} = \begin{bmatrix} \frac{\lambda}{f_x}(u - p_x) + \beta_x \lambda \\ \frac{\lambda}{f_y}(v - p_y) + \beta_y \lambda \\ \lambda + \Delta z \end{bmatrix} \tag{9}$$

$$\hat{\mathbf{x}}_{\text{hom}} = K\,\hat{\mathbf{p}}_C = \begin{bmatrix} \lambda(u - p_x) + f_x\beta_x\lambda + p_x(\lambda + \Delta z) \\ \lambda(v - p_y) + f_y\beta_y\lambda + p_y(\lambda + \Delta z) \\ \lambda + \Delta z \end{bmatrix} \tag{10}$$

Again, we divide by the homogeneous component to obtain the transformed image point:

$$\hat{\mathbf{x}} = \frac{1}{\lambda + \Delta z}\begin{bmatrix} \lambda(u - p_x) + f_x\beta_x\lambda + p_x(\lambda + \Delta z) \\ \lambda(v - p_y) + f_y\beta_y\lambda + p_y(\lambda + \Delta z) \end{bmatrix} = \begin{bmatrix} \frac{\lambda}{\lambda+\Delta z}(u - p_x) + \frac{\lambda}{\lambda+\Delta z}f_x\beta_x + p_x \\ \frac{\lambda}{\lambda+\Delta z}(v - p_y) + \frac{\lambda}{\lambda+\Delta z}f_y\beta_y + p_y \end{bmatrix} \tag{11}$$

$$\text{If } \lambda \gg \Delta z \implies \hat{\mathbf{x}} \approx \begin{bmatrix} u + f_x\beta_x \\ v + f_y\beta_y \end{bmatrix} = \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} \iff \hat{\mathbf{x}} \neq f(\lambda) \tag{12}$$

This shows that the resulting transformed point is independent of the depth when the change in depth is small or when the depth itself is large. This approximation is justified when the depth is sampled densely in a dataset; in this case, changes in depth between nearby views are negligible. For nearest neighbors, this is the case for our datasets.

Lastly, we repeat the same procedure for any relative rotation about the out-of-plane camera x and y axes, which we expect to depend on precise depth information. For this example, we choose the x axis, but the procedure is equivalent for the y axis or both axes simultaneously.

$$\hat{\mathbf{p}}_\mathbf{C} = R\,\mathbf{p}_\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \lambda \begin{bmatrix} \frac{u}{f_x} - \frac{p_x}{f_x} \\ \frac{v}{f_y} - \frac{p_y}{f_y} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{\lambda}{f_x}(u - p_x) \\ \cos\alpha\,\frac{\lambda}{f_y}(v - p_y) - \lambda\sin\alpha \\ \sin\alpha\,\frac{\lambda}{f_y}(v - p_y) + \lambda\cos\alpha \end{bmatrix} \tag{13}$$

$$\hat{\mathbf{x}}_\mathbf{hom} = K\,\hat{\mathbf{p}}_\mathbf{C} = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\lambda}{f_x}(u - p_x) \\ \cos\alpha\,\frac{\lambda}{f_y}(v - p_y) - \lambda\sin\alpha \\ \sin\alpha\,\frac{\lambda}{f_y}(v - p_y) + \lambda\cos\alpha \end{bmatrix} \iff \underline{\hat{\mathbf{x}} = f(\lambda)} \tag{14}$$

As expected, when the homogeneous image point $\hat{\mathbf{x}}_\mathbf{hom}$ is divided by its z component to get the transformed image point $\hat{\mathbf{x}}$, it depends non-linearly on λ which means that this is difficult to represent using VS as the transformed image point depends on precise depth information.

This shows that the C-L2 metric is an inappropriate choice as a distance norm for VS because of its isotropic behavior with respect to the x,y and z axes and its ignorance of isolated relative rotation. An approach to solving this is to take the sum of the normalized C-L2 metric together with the magnitude of the relative attitude, this was used for evaluation by the ESA-ACT + SLAB SPEC competitions[31] ("e_{pose}") and authors who followed.[2,8,25,35]

This combined metric has the problem that the relative weights of position and attitude errors must be chosen manually. In general, the magnitudes of position and attitude are both real numbers but they have different dimensions (distances vs. angles); simply adding them is imprecise. In addition, attitude is bounded and can therefore easily be normalized while position is potentially unbounded. The BDD resolves these issues by being defined solely in the attitude space S^3 and considering only the rotations that "matter", namely the out-of-plane axes of rotation.

DOI: 10.13009/EUCASS2025-011

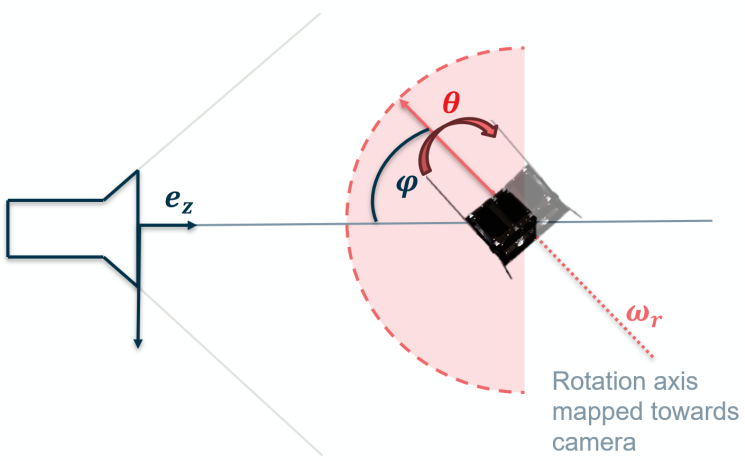


Figure 7: Illustration of the angles ϕ and θ of the BDD.

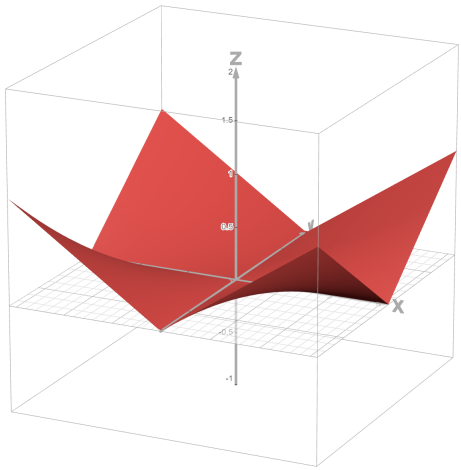


Figure 8: 3D plot of the BDD, $x = \phi \in (-\frac{\pi}{2}, \frac{\pi}{2})$, $y = \theta \in (-\pi, \pi)$, $z = BDD$.

5.3 Definition of the Boresight Deviation Distance

$$BDD : S^3 \times S^3 \to \mathbb{R}, \quad BDD(q_1, q_2) = \frac{|\theta|}{\pi}\left(1 - \left|\frac{2|\phi|}{\pi} - 1\right|\right), \quad \phi \in (-\frac{\pi}{2}, \frac{\pi}{2}], \theta \in (-\pi, \pi], \; BDD \in [0, 1] \tag{15}$$

$$\text{where} \quad q_1 \sim -q_1, \; q_2 \sim -q_2 \quad (S^3 \to SO(3)) \tag{16}$$

$$q_r = r_r \; + \mathbf{v_r} = [r_r, q_{rx}, q_{ry}, q_{rz}] = q_1 q_2^* \tag{17}$$

$$\theta = 2 \cdot \text{atan2}(|\mathbf{v_r}|, r_r), \tag{18}$$

$$\boldsymbol{\omega_r} = [\omega_x, \omega_y, \omega_z] = \frac{\mathbf{v_r}}{\sin(\frac{\theta}{2})} \text{ if } \theta \neq 0 \text{ else } \boldsymbol{\omega_r} = \mathbf{0} \tag{19}$$

$$\boldsymbol{\omega_{r+}} = [\omega_x, \omega_y, |\omega_z|] \quad \mathbf{e_z} = [0, 0, 1] \tag{20}$$

$$\phi = \text{atan2}(|\mathbf{e_z} \times \boldsymbol{\omega_{r+}}|, \mathbf{e_z}^T \boldsymbol{\omega_{r+}}) \tag{21}$$

The input to the BDD consists of two poses. First, the relative rotation q_r between the attitude components is computed. This is converted to an axis angle representation, of which θ is the angular magnitude. The rotation axis of the relative rotation is obtained as $\boldsymbol{\omega_r}$ and mapped to the semi-sphere pointing towards the camera, resulting in $\boldsymbol{\omega_{r+}}$. Lastly, ϕ is the angle between $\boldsymbol{\omega_{r+}}$ and the camera boresight axis $\mathbf{e}_z$, computed with the numerically stable atan2(...).

The BDD is intuitively understood by considering the illustration in 7. In essence, the BDD encodes how far the target's rotation axis is from the boresight axis. To this end, the angle ϕ between the target's relative rotation axis and the boresight axis is combined with the magnitude θ of that rotation. Thus, the BDD vanishes when either of them is zero as both of these extremes can accurately be recreated using VS. This captures the previously shown anisotropy.

5.4 Advantages of the Boresight Deviation Distance

The advantages of the BDD, particularly with respect to C-L2, are the following:

1. It is inherently normalized between 0 and 1, while C-L2 is potentially unbounded and needs manual tuning (w.r.t. the experimental setup, like shown by[24]) to normalize. This is advantageous for defining a density metric as simply the inverse of the BDD. This property also makes it dataset-agnostic: a given BDD value has the same meaning across different datasets without any manual tuning or thresholding necessary.

2. It is anisotropic w.r.t. the six degrees of freedom of camera poses and only keeps track of the most important components. In contrast, C-L2 does not take into account the relative attitudes between camera poses. For C-L2, one must manually add a rotational term which degrades the properties of the resulting metric.

3. It offers an upper bound for VS. Any target with a BDD > 0.5 w.r.t. the source is not possible to synthesize. This is due to the fact that at greater values previously unseen faces of the target geometry always become visible.

DOI: 10.13009/EUCASS2025-011

4. Lastly, the BDD fulfills the necessary conditions for a metric between members of the rotation group $SO(3)$:

$$(1) \quad BDD(R,R) = 0 \tag{22}$$

$$(2) \quad R_1 \neq R_2 \implies BDD(R_1,R_2) > 0 \tag{23}$$

$$(3) \quad BDD(R_1,R_2) = BDD(R_2,R_1) \tag{24}$$

$$(4) \quad BDD(R_1,R_3) \leq BDD(R_1,R_2) + BDD(R_2,R_3) \tag{25}$$

Where R is a member of $SO(3)$. Due to the double cover of the unit quaternions, the antipodal points of S^3 (negation of a given unit quaternion $q \to -q$) must be identified by the map $q \to R$, as shown in 16. In practice, this is achieved by ensuring that the scalar part of the relative quaternion q_r is always positive.

5.5 Density calculation

One of the possible applications of VS is "densifying" existing datasets after they have been acquired or even planning an acquisition beforehand based on the desired density. To achieve this, a density metric must be established which serves to measure how densely the set of all poses is covered by a dataset.

Thanks to property 1 the BDD can be conveniently used to define this density metric. The density of a set of poses can be defined as the inverse of the size of the largest empty BDD-ball which fits into the set of poses. A BDD-ball is defined as $\{R : BDD(R_{\text{query}}, R) < d\}$ where d is the size and R_{query} the center of the ball. The density is then: $\rho_{BDD} := \frac{1}{BDD_{max}}$ where BDD_{max} is the size of the largest ball which does not contain any poses of a dataset ("empty"). BDD_{max} is referred to in the following as **LB-BDD**. This measure can be spatially visualized as a cone, see 11.

Using the LB-BDD means that the density of a dataset is not evaluated on a nearest-neighbor basis only. Doing so would unfairly bias in favor of trajectory-type datasets which are dense in the space of the trajectory but sparse in the whole space. The LB-BDD avoids this issue. In practice, to compute the LB-BDD, an ideal dense baseline sampling must be defined. We generate this sampling with a blue-noise-like pose generation algorithm inspired by "Mitchell's Best Candidate Algorithm".[21] The LB-BDD of a set of poses is then computed with respect to this ideal sampling using a min-max greedy nearest-neighbor algorithm.

6. View Synthesis Methods

In this chapter, we describe the general principles of view synthesis (6.1) and the two methods we chose to synthesize novel views (6.2 and 6.3).

6.1 View Synthesis

VS renders novel 2D views from an existing dataset. Crucially, VS does not render, it generates new images from existing ones. Therefore, this method cannot deliver additional information beyond what is already present in the dataset, barring any optional data augmentation. What VS instead offers is to exploit the available data more fully.

In general, to synthesize novel views we transform their nearest-neighbor (in terms of the BDD) that already belongs to the dataset to the queried novel pose. The justification for choosing the BDD for this purpose is shown formally in 5.2 and is also shown empirically in 3. The nearest-neighbor (NN) search is performed by building a k-d-tree using the BDD, the tree can then quickly be queried to find the existing NN of a desired novel pose.

In this work, we use exclusively analytic, as in non-generative, methods for VS. The two proposed methods are named Homography and 3D Transform. A third one based on View Morphing[29] has been tested but needs further improvements in accuracy and runtime to be viable. This will be done in future work. An important note concerning our VS methods is that they are not able to synthesize lighting changes between views, only geometric ones.

6.2 Homography Transform

The first method is called the Homography Transform. It is based on the projective image transformation of the same name. It is the faster, but less accurate, of the two VS methods presented in this work.

The algorithm for the Homography Transform is described in 1 and is adopted from.[13] The principle of this method is based on an approximation of the image as the projection of a planar scene. In our implementation, the depth to the corresponding world points is assumed to be constant, i.e. the normal vector of the plane is assumed to be parallel to the boresight axis. Note that the plane in 9 is inclined only for visual purposes; this is not done in the implementation.

We use the `warpPerspective` routine from `OpenCV`, which executes a perspective transformation. Optionally, the algorithm allows for providing the mask and 2D keypoint locations for the input image to obtain their transformed counterparts. We use the transformed mask to evaluate the accuracy of the synthesized view.

DOI: 10.13009/EUCASS2025-011

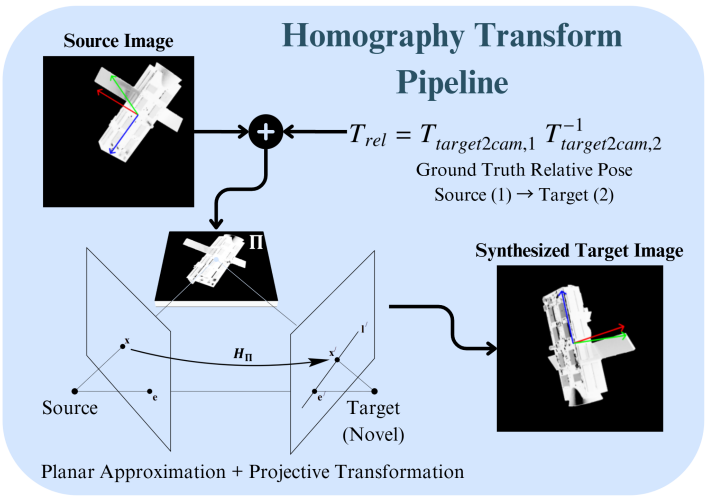


Figure 9: Principle behind the Homography Transform. Homography figure adapted from.[13]

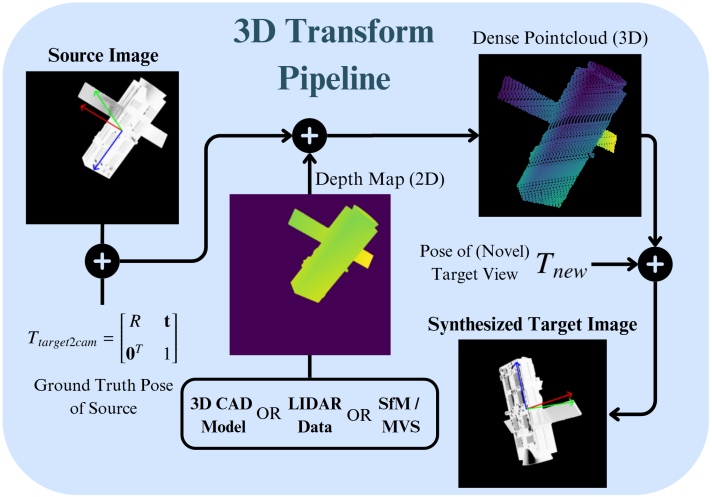


Figure 10: Principle behind the 3D Transform.

Algorithm 1 Homography Transform

Require: Source image I_S, source pose T_S, intrinsics K, optionally source mask M
Require: Target view's desired pose T_T
Ensure: Synthesized target image $I_{T,\text{synth}}$

1: **Plane & distance** $\mathbf{n} \leftarrow (0,0,1)^T, \quad d \leftarrow \|\mathbf{t_S}\|$
2: **Relative pose** $D \leftarrow T_T\ T_S^{-1} \qquad R_{12} \leftarrow D_{1:3,1:3},\ \mathbf{t}_{12} \leftarrow D_{1:3,4}$
3: **World homography** $H \leftarrow R_{12} + \dfrac{\mathbf{t}_{12}\,\mathbf{n}^T}{d}$
4: **Image–plane map** $G \leftarrow K\,H\,K^{-1}$
5: **Warp** $I_{T,\text{synth}} \leftarrow \text{warpPerspective}(I_S, G)$
6: **Transformed mask** $M' \leftarrow \text{warpPerspective}(M, G, \text{border} = 0)$
7: **return** $I_{T,\text{synth}}$, optionally M'

6.3 3D Transform

The 3D Transform addresses the key limitation of the Homography Transform, that being its inaccurate depth approximation, by leveraging available depth data (e.g., a depth-map rendered with a 3D CAD model or LIDAR capture). If neither is available, the 3D Transform cannot be used; in that case, the Homography Transform shall be used. If the latter delivers unsatisfactory results a multi-reference method like View Morphing shall be used.

Unlike the Homography Transform's single perspective warp, the 3D Transform operates on each pixel individually, which may leave gaps in the target view. Strategies for filling these gaps will be discussed below. In our implementation, depth data comes from a depth-map rendered using the 3D CAD model of the target. The rendering of the depth-map in our implementation is done with `pyrender`'s `OffscreenRenderer`. The grayscale intensity value of each pixel in the depth-map gives the z-coordinate in the camera reference frame to the underlying 3D world point of the same pixel; the 3D point is then found by evaluating the back-projected camera ray at that depth value.

Because pixels are handled one-by-one, two distinct 3D points can project to the same target pixel, creating "intra-image" gaps. (Extra-image gaps i.e. out-of-focus-areas[22] are resolved by masking the input image). Intra-image gaps are filled using morphological operations and interpolation: a sliding 5×5 convolution window gathers valid neighbors (detected via a mask that saves where source pixels land in the target image) and computes color values for the gaps.

See 2 for a detailed description of the 3D Transform's algorithm.

One possibility to improve the algorithm is to not detect gaps only based on the transformed coordinates of the source pixels but to fill all the pixels belonging to the target model in the synthesized image. These pixels could be detected using the target image's mask of the target model which can easily be obtained by thresholding the target image's depth-map. In this way, disocclusions that arise from new faces of the target model becoming visible, which are ignored in the above algorithm, would also be filled. However, this is not done this in this work to not bias the results of the IoU metric (which would always be 1 otherwise, as the whole mask would be filled).

DOI: 10.13009/EUCASS2025-011

Algorithm 2 3D Transform with interpolation

Require: Source image I_S, source pose T_S, intrinsics K, 3D model of target object O
Require: Target view's desired pose T_T
Ensure: Synthesized target image $I_{T,\text{synth}}$
Render source depth map D_S using O, K and T_S
for each pixel $\tilde{p} = (u, v, 1)$ in I_S **do**
  Camera coordinates $P_c \leftarrow D_S(K^{-1}\,\tilde{p})$ {evaluate ray at depth map pixel value}
  World coordinates $P \leftarrow R_S^\top(P_c - \mathbf{t_S})$
  Target image coordinates $\tilde{p'} \leftarrow K\,(R_T P + \mathbf{t_T})$
  $I_{T,\text{synth}}(\tilde{p'}) \leftarrow I_S(\tilde{p})$
end for
Valid mask $V_T = -\mathbf{1}$, override with transformed pixels
for each gap $h \in \{\, h \mid V_T(h) = -1 \ \wedge D_S(h) \geq 0\}$ **do**
  Interpolate $I_{T,\text{synth}}(h)$ from neighboring valid pixels in a 5×5 window
end for
return $I_{T,\text{synth}}$

7. Validation

To validate VISY-REVE, we create a general performance model by means of a Monte Carlo (MC) campaign in which, for each draw, a synthesized image is compared with the actual existing image. The MC results are then used to derive performance model specifications that show the limits of VS and that can inform its real-world use. To this end, first, the necessary quality metrics to evaluate VS will be defined (7.1). Then, the MC process is explained in detail (7.2). The results of MC are described in 8. The practical application performance of VS will be analyzed through two test-cases which model possible real-life use-cases. Their results are described in 8.2.1 and 8.2.2.

7.1 Quality metrics

Two different types of metrics are needed to evaluate VS. The first comprises quality metrics that evaluate how similar a synthesized image is to the actual image. The second includes distance metrics that measure the distance between poses; these should also be predictive of the accuracy of VS, as explained in 5.1. Together, they can be used to derive the performance model. As distance metric, the BDD is chosen, which is designed for this purpose.

As quality metrics, multiple ones are chosen to cover different aspects of VS accuracy. The chosen metrics are all full-reference, i.e. they compare a synthesized image with its ideal counterpart. We opt for a balance between optical, i.e. metrics that identify radiometric differences; structural, i.e. metrics that identify geometric differences; and functional, i.e. metrics that represent the functioning of VBN algorithms. The chosen metrics are the following:

1. Optical, **SSIM**. The Structural Similarity metric.[34] The SSIM is computed using images cropped to the bounding boxes of the target model to remove bias w.r.t. its size.
2. Structural, **IoU**. The Intersection over Union between two masks of the target model: The transformed source mask for the synthesized view M' and the mask for the actual view M_T, formally: $IoU(M', M_T) = \frac{M' \cap M_T}{M' \cup M_T}$.
3. Functional, **KPS-L2**, Mean of the Euclidean reprojection errors between ground-truth 2D keypoints $\mathbf{u}$ and as detected by a convolutional neural network (CNN) $\hat{\mathbf{u}}$. Keypoints are landmarks defined on the 3D geometry of the target model that can be projected into each image based on the ground-truth pose. We opt for detecting keypoints in the synthesized images with a CNN instead of directly using their transformed coordinates as the CNN detections also encode the visual fidelity of the synthesized images, not just their geometric accuracy. Formally: $KPS\text{-}L2 = \frac{1}{K}\sum_{i=1}^{K} \|\hat{\mathbf{u}}_i - \mathbf{u}_i\|_2$ where K is the number of keypoints.
4. Functional, **KPS-VBN**. Ratio between: 1. The mean of the Euclidean distances between the ground-truth 3D keypoints $\mathbf{X}$ and the back-projected CNN-detected 2D keypoints $\hat{\mathbf{X}}$. The back-projection is done using the ground-truth pose. This is divided by: 2. The Euclidean distance between the camera and the target model, resulting in a relative range metric. This kind of metric is typical for VBN missions, a common baseline is 1%. It makes the error in the keypoints invariant to the range, as shown by.[14] Formally: $KPS\text{-}VBN = \frac{\frac{1}{K}\sum_{i=1}^{K} \|\hat{\mathbf{X}}_i - \mathbf{X}_i\|_2}{\|\mathbf{t}\|_2}$.

The neural network employed for metrics 3. and 4. uses the HRNet architecture[32] which is state-of-the-art for pose estimation by keypoint detection in VBN.[7,35] It is trained to regress heatmaps for the 2D projections of the 3D

DOI: 10.13009/EUCASS2025-011

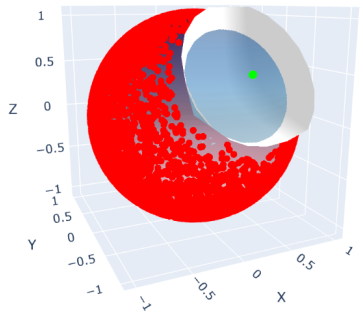


Figure 11: A LB-BDD of 0.2 visualized as a white cone. Each red point marks a camera position, normalized to the unit sphere, ball center marked in green (technically a line).

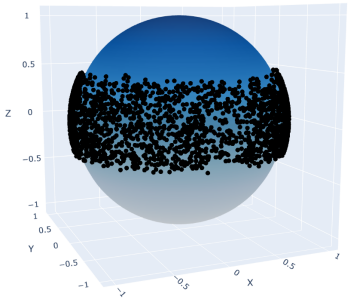


Figure 12: Initial ESA-AIRBUS dataset with missing poles. Camera positions normalized to unit sphere.

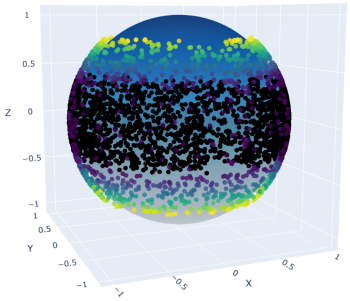


Figure 13: Densified ESA-AIRBUS with new views up to a BDD of 0.1. Brighter means a higher BDD from source to target.

keypoints. The number of keypoints chosen is 8 for SWISSCUBE, 9 for SPEED+ and 16 for ESA-AIRBUS, their positions are chosen manually to coincide with salient features on the target models. The network contains around 63.5 million parameters. For training and inference, the input images are normalized and resized to 288×384 pixels with an output heatmap size for each keypoint of 144×192. The network was trained for 20 epochs with a learning rate of 0.0005, achieving a final PCK test score (percentage of correct keypoints) of at least 0.995 for all datasets, which means that 99.5% of detected keypoints fall within 5% of the heatmap width w.r.t. their ground truth locations.

7.2 Implementation

The MC process uses a strategy we name **sample replacement**, similar to.[27] This involves continuously drawing new pairs of samples from a dataset. The first image of the pair, named *target*, is synthesized from the second one, named *source*. The synthesized and actual target images are then compared with the quality metrics described above. The MC is carried out with a sample size of 10.000 sample replacements for each dataset. The pairs are chosen randomly, albeit so that their BDD is below 0.5, see 3. The background is masked out on all source images to reduce bias.

8. Results

8.1 Performance Model

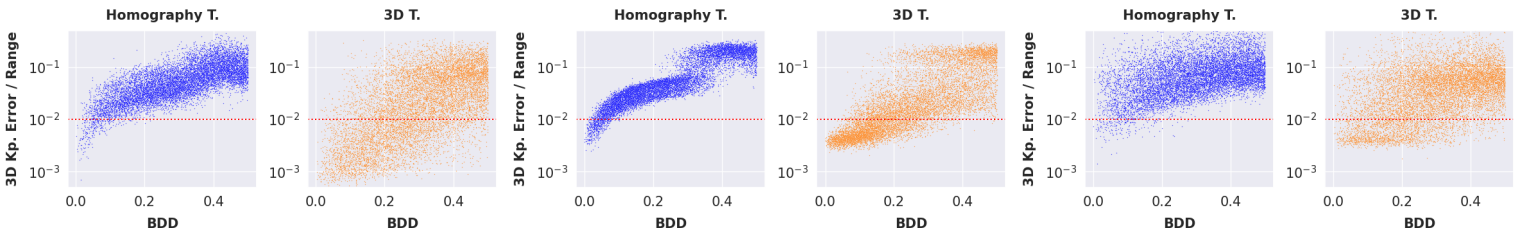
The results of the MC are shown in detail in 2 and are visualized for the KPS-VBN metric in 14. Table 2 shows the maximum LB-BDD the respective datasets could have so that VS would fulfill the quality requirements with a 3σ confidence. Using confidence intervals is necessary because of occasional outliers. As an example for the 1% range requirement (KPS-VBN metric 4), using the best VS method, the 3D Transform, the results are: 0.087 for ESA-AIRBUS, 0.024 for SPEED+ and 0.001 for SWISSCUBE (higher is better). The actual LB-BDD values of these datasets are 0.31 for ESA-AIRBUS, 0.0045 for SPEED+ and 0.0071 for SWISSCUBE (lower is better), see 5.5.

This means that for SPEED+, since $0.024 > 0.0045$, **every possible novel pose** can be synthesized and fulfill the 1% range-requirement with greater than 99.7% confidence. For ESA-AIRBUS and SWISSCUBE this is not the case, namely because ESA-AIRBUS has a very high LB-BDD with its missing poles as shown in 12 and for SWISSCUBE because of the small target, which is difficult to recognize by our CNN for even small deviations induced by VS.

Dataset	**KPS-VBN** $3\sigma - 1\%$		**IoU** $3\sigma - 0.9$		**SSIM** $3\sigma - 0.9$	
	HOM	**3DT**	**HOM**	**3DT**	**HOM**	**3DT**
ESA-AIRBUS	0.044	0.087	0.0304	0.097	-	-
SPEED+	0.016	0.024	0.0044	0.016	0.0041	0.0041
SWISSCUBE	0.001	0.001	0.001	0.001	0.0025	0.001

Table 2: 3-sigma MC results. The values in the cells are LB-BDD. For KPS-VBN we measure the maximum BDD under which 3σ% of sample replacements fulfill the 1% VBN range-requirement. For IoU and SSIM we measure the max. BDD under which 3σ% of sample replacements are above 0.9. HOM is Homography, 3DT is 3D Transform. Higher is better. Values rounded to two significant figures. SSIM requirement for ESA-AIRBUS never fulfilled.

DOI: 10.13009/EUCASS2025-011



(a) SPEED+ (b) ESA-AIRBUS (c) SWISSCUBE

Figure 14: MC results for the KPS-VBN quality metric ("3D Kp. Error / Range"). The red horizontal lines mark the 1% VBN relative range-requirement. Note that the y axis is logarithmic, its limits and scale have been kept the same across all plots. Table 2 can be interpreted as vertical cutoff lines in these plots at the BDD values given in the table.

8.1.1 Correlations BDD vs. C-L2

We empirically show the predictive capabilities of the BDD compared to C-L2 w.r.t. the accuracy of VS. For this comparison, we calculate the Pearson Correlation Coefficients[11] between the quality metrics (KPS-L2, IoU, SSIM) and the BDD / C-L2 for the MC sample replacement results. The correlations values are shown in 3. The entries in bold mark the distance metric with the higher correlation.

The BDD generally outperforms C-L2 in the KPS-L2 quality metric, which best represents the current use-case of VBN algorithms. Only for the SSIM C-L2 is mostly superior. This is relativized by the fact that the correlations for the SSIM are generally lower and that for all metrics, when the C-L2 is better, the margins are small. We can conclude that the BDD is indeed more representative of the mechanism of VS and that it is thus the better distance metric. In 17 we visually show this for SPEED+, also compared with the rotation magnitude and SPEC metrics.

Distance Metric	Dataset	KPS-L2		IoU		SSIM	
		HOM	3DT	HOM	3DT	HOM	3DT
BDD	ESA-AIRBUS	**0.83**	0.62	-0.94	-0.76	-0.37	-0.35
	SPEED+	**0.60**	**0.45**	**-0.92**	**-0.54**	-0.10	**-0.27**
	SWISSCUBE	**0.29**	**0.27**	**-0.58**	-0.34	**0.08**	-0.08
C-L2	AIRBUS	0.82	0.62	-0.94	**-0.77**	**-0.39**	**-0.36**
	SPEED+	0.19	0.16	-0.66	-0.36	**-0.12**	-0.25
	SWISSCUBE	0.06	0.11	-0.53	**-0.35**	0.02	**-0.12**

Table 3: Correlation table BDD & C-L2. HOM is Homography Transform, 3DT is 3D Transform. Higher is better.

8.2 Functional Validation

8.2.1 Test-case 1: Real-Time Trajectory Generation

One of the main use-cases of VISY-REVE is the real-time in-the-loop synthesis of images, only needing a sparse dataset as input. VISY-REVE can be used as a replacement for synthetic rendering or laboratory acquisition to close the loop for testing image processing algorithms. Combined with a GNC stack, VS therefore enables real-time **novel trajectory synthesis**.[33] Note also that VS is even able to synthesize images that lie outside the convex hull of the dataset. For best results, however, the lighting position should remain the same or similar across the dataset.

We demonstrate this use-case qualitatively: we synthesize a trajectory with small deviations from an existing one, see 15. Pose estimation is done for both trajectories using the same CNN as before, combined with PnP. The errors are shown in 16. As can be seen, the poses can accurately be reconstructed from the synthesized images most of the time.

8.2.2 Test-case 2: Densifying Datasets

The second promising use-case of VISY-REVE is to *densify*, i.e. increase the number of samples in a dataset in an optimal manner. To this end, first, regions of poor sampling in the dataset are identified using the BDD and the ideal sampling described in 5.5. Then, VS is used to populate these regions with images and ground-truth labels.

The methodology of identifying under-sampled regions using the BDD can also be employed at dataset creation-time. In that case, the image poses to acquire can be defined w.r.t. the density requirements for accurate VS found in 2. Thereby, R.T. or synthetic acquisition campaigns are shortened and gaps can be filled using VS on-line or off-line.

DOI: 10.13009/EUCASS2025-011

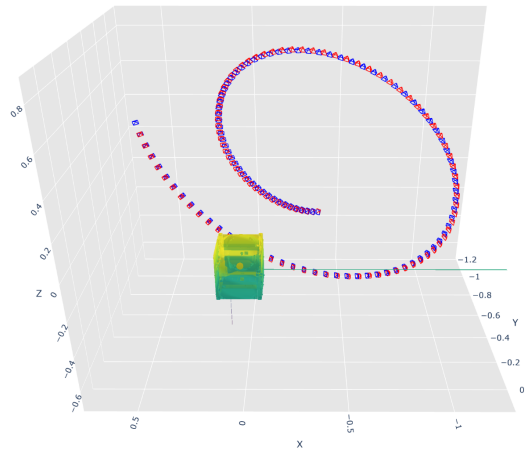


Figure 15: Existing trajectory (blue) and synthesized trajectory (red) from SWISSCUBE.

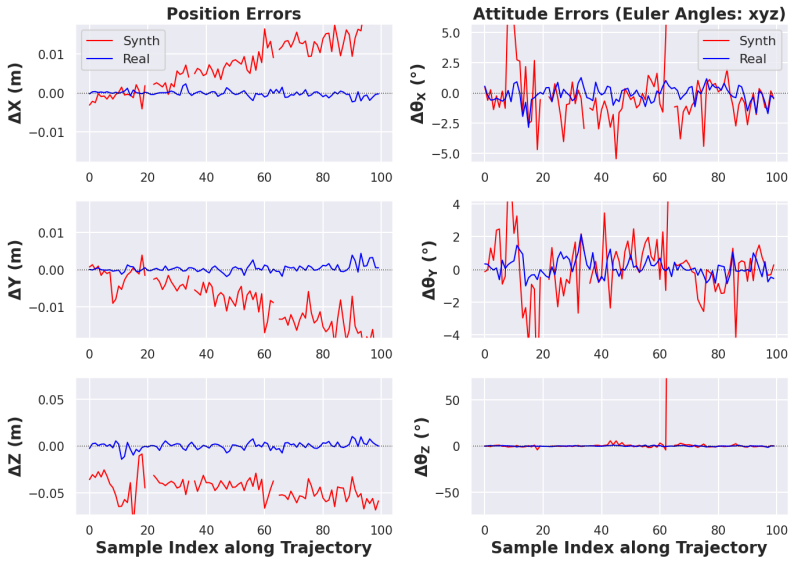


Figure 16: HRNet + PnP performance for trajectories in left image. Y-axes clipped to 3σ bounds of the combined errors.

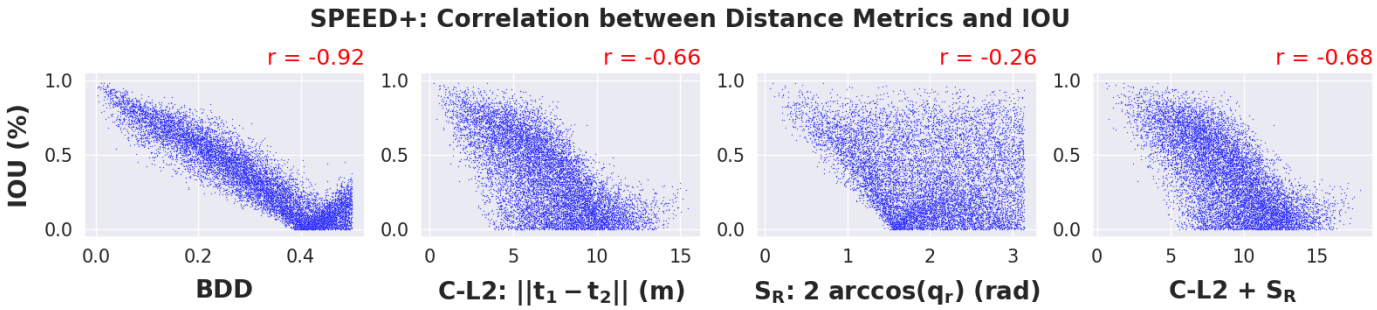


Figure 17: Correlation results of the IoU metric for SPEED+. It can be clearly seen that the BDD better predicts the IoU than any other distance metric. Red numbers indicate Pearson Correlation Coefficients. Compare 3.

We demonstrate this use-case in 13. Here, the ESA-AIRBUS dataset is densified using the 3D Transform and its density value ρ_{BDD} is increased from 3.33 to 5 (LB-BDD: 0.3 to 0.2), an increase of 50%. This exceeds the 3σ limit (cf. 2). However, for ESA-AIRBUS the 2σ KPS-VBN 1% range-requirement with the 3D T. is fulfilled up to a LB-BDD of 0.11. Therefore, we see that the synthesized images will fulfill the requirement more than 95% of the time.

8.2.3 Runtime Analysis

The runtime of VS is analyzed. The tests are performed on a laptop with an AMD Ryzen 5 3500U CPU at 1400 clock speed, 16 GB RAM, AMD Radeon(TM) Vega 8 GPU. The algorithms are all run single-threaded. The benchmark is the synthesis of a random cubic spline trajectory around the origin of 10 m length containing 500 samples for the SPEED+ dataset with an image size of 1200x1920 pixels. The results are rounded to three significant figures.

The results for the Homography Transform is an 0.476s average, including rendering the depth-map (to mask the background on the input image) and including loading the image from disk. Without masking the input, so without depth-map rendering, this drops to 0.0119s on average. For the 3D Transform with interpolation, it takes 0.852s on average, including rendering the depth map (necessary to perform the transform and to mask the background) and including loading the image from disk. Without interpolation, this drops to 0.699s on average.

9. Discussion

Through the MC campaign and the resulting performance model, it has been successfully shown that for SPEED+, VS has the necessary accuracy to synthesize **all possible novel poses** while fulfilling the 1% range-requirement for the CNN-based keypoint detection error with greater than 99.7% confidence. For the SWISSCUBE and ESA-AIRBUS

DOI: 10.13009/EUCASS2025-011

datasets, not every novel pose can be synthesized at this level, although for both datasets, every NN can be synthesized to fulfill the requirement, as the average NN-BDD values for these datasets are near zero.

The results of the two test-cases indicate that the presented VISY-REVE pipeline, namely dataset evaluation and VS, is useful for real-life validation of VBN image processing algorithms, although more quantitative experiments are needed to confirm this. Lastly, we have shown that the BDD is a superior metric for choosing NN views for VS as it better represents the drop in quality incurred by an increase in distance.

We acknowledge that having chosen a CNN as a validation technique may have negatively impacted the presented findings, as the results are sensitive to the quality of the trained network. In particular, there is a domain gap between synthesized and actual images, which is not present during training. This gap is due to deviations in the images induced by VS, namely deformations for the Homography Transform and gaps for the 3D Transform. Using data augmentation as shown by[7,24,25] may improve the findings as the CNN would learn to better generalize across domains.

The *complete source code of our work is released publicly*; please see the project page for more information.

10. Further Work

Speed is a crucial factor for the adoption of VISY-REVE. For this work, all algorithms were implemented in Python for a single-threaded CPU runtime and were not yet sufficiently optimized. For better performance, the algorithms would have to be converted to C/C++. GPU acceleration is also viable, as the VS methods are easily parallelizable.

To further the development of the BDD, a closed-form solution for a desired rotation given a BDD and a query rotation should be developed which would be useful to avoid having to use rejection sampling as in this work. The BDD should also be enhanced to take into account foreshortening effects associated with translation of the camera.

Additionally, using more and potentially better VS methods is possible. In particular, using e.g. View Morphing which can combine multiple source views and their information to yield the target view might result in better accuracy as this would enable synthesizing lighting differences associated to changes in pose. Furthermore, we plan to implement modern VS methods such as NeRF,[20] which are also capable of synthesizing lighting changes.

Lastly, as mentioned before, the CNN used for validation should be trained with photometric and geometric data augmentation to see if accuracy on synthesized images improves, particularly for closed-loop-like trajectories as in 15.

11. Acknowledgments

The results presented in this paper have been achieved during a research stage at the European Space Agency (ESA). The views expressed in this paper can in no way be taken to reflect the official opinion of the European Space Agency.

The authors thank the TEC-ECG section of ESA for all the inspiring conversations and teachings that helped bring this work to life. In particular, we want to thank Paul Duteïs for his generous help and continuous availability, as well as Massimo Casasco for his trust and the opportunity to pursue this work.

References

[1] Alexander Amini, Igor Gilitschenski, Jacob Phillips, Julia Moseyko, Rohan Banerjee, Sertac Karaman, and Daniela Rus. Learning robust control policies for end-to-end autonomous driving from data-driven simulation. *IEEE Robotics and Automation Letters*, 5:1143–1150, 4 2020.

[2] Kevin Black, Shrivu Shankar, Daniel Fonseka, Jacob Deutsch, Abhimanyu Dhir, and Maruthi R. Akella. Real-time, flight-ready, non-cooperative spacecraft pose estimation using monocular imagery. 1 2021.

[3] P. Bodin, R. Noteborn, R. Larsson, T. Karlsson, S. D'Amico, J-S. Ardaens, M. Delpech, and J-C. Berges. *Prisma formation flying demonstrator: overview and conclusions from the nominal mission*. AAS 12-072; 35th Annual AAS Guidance and Control Conference, 8 February 2012, Breckenridge, 2012.

[4] Antonio Canclini, Francesco Malapelle, Marco Marcon, Stefano Tubaro, and Andrea Fusiello. View-synthesis from uncalibrated cameras and parallel planes. *Signal Processing: Image Communication*, 79:40–53, 2019.

[5] L. Pasqualetto Cassinis, R. Fonod, E. Gill, I. Ahrns, and J. Gil Fernandez. Cnn-based pose estimation system for close-proximity operations around uncooperative spacecraft. In *AIAA Scitech 2020 Forum*, volume 1 PartF. American Institute of Aeronautics and Astronautics Inc, AIAA, 2020.

[6] Lorenzo Pasqualetto Cassinis, Alessandra Menicucci, Eberhard Gill, Ingo Ahrns, and Manuel Sanchez-Gestido. On-ground validation of a cnn-based monocular pose estimation system for uncooperative spacecraft: Bridging domain shift in rendezvous scenarios. *Acta Astronautica*, 196:123–138, 7 2022.

[7] Pasqualetto Cassinis. Monocular vision-based pose estimation of uncooperative spacecraft. 2022.

DOI: 10.13009/EUCASS2025-011

[8] Bo Chen, Jiewei Cao, Alvaro Parra, and Tat-Jun Chin. Satellite pose estimation with deep landmark regression and nonlinear pose refinement. 8 2019.

[9] Simone D'Amico, N.A. Amico, Mathias Benn, and John L. Jørgensen. Pose estimation of an uncooperative spacecraft from actual space imagery. *International Journal of Space Science and Engineering*, 2:171, 2014.

[10] Jeff Delaune. Vision-based navigation for mars helicopters ingenuity & mars science helicopter, 2021.

[11] David Freedman, Robert Pisani, and Roger Purves. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007.

[12] Jesús Gil-Fernández, David Gonzalez-Arjona, Joao Branco, Francisco Cabral, Jesus Gil-Fernandez, Thomas Voirin, Irene Huertas, Denis Rebuffat, and Bruno Sousa. Vision-based guidance, navigation and control system for phobos sample return mission, 2015.

[13] Richard. Hartley and Andrew. Zisserman. *Multiple view geometry in computer vision Richard Hartley, Andrew Zisserman*. Cambridge University Press, 2013.

[14] Yinlin Hu, Sebastien Speierer, Wenzel Jakob, Pascal Fua, and Mathieu Salzmann. Wide-depth-range 6d object pose estimation in space. 4 2021.

[15] Jérémy Lebreton, Ingo Ahrns, Roland Brochard, Christoph Haskamp, Matthieu Le Goff, Nicolas Menga, Nicolas Ollagnier, Ralf Regele, Francesco Capolupo, and Massimo Casasco. Training datasets generation for machine learning: Application to vision based navigation. 9 2024.

[16] Jérémy Lebreton, Roland Brochard, Matthieu Baudry, Grégory Jonniaux, Adrien Hadj Salah, Keyvan Kanani, Matthieu Le Goff, Aurore Masson, Nicolas Ollagnier, Paolo Panicucci, Amsha Proag, and Cyril Robin. Image simulation for space applications with the surrender software.

[17] Zhi-Song Liu, Marie-Paule Cani, and Wan-Chi Siu. See360: Novel panoramic view interpolation. *IEEE Transactions on Image Processing*, 31:1857–1869, 2022.

[18] I. Martin, M. Dunstan, and M. Sanchez Gestido. Planetary surface image generation for testing future space missions with pangu. In *2nd RPI Space Imaging Workshop, Sensing, Estimation, and Automation Laboratory*, 2019.

[19] Aurore Masson, Paul Duteis, Christoph Haskamp, Ingo Ahrns, Roland Brochard, Keyvan Kanani, and Remi Delage. Airbus ds vision based navigation solutions tested on liris experiment data. *7th European Conference on Space Debris*, 2017.

[20] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020.

[21] Don P. Mitchell. Spectrally optimal sampling for ray tracing. *SIGGRAPH Comput. Graph.*, 25(4):157–164, July 1991.

[22] Adriano Q. De Oliveira, Thiago L.T.Da Silveira, Marcelo Walter, and Claudio R. Jung. A hierarchical superpixel-based approach for dibr view synthesis. *IEEE Transactions on Image Processing*, 30:6408–6419, 2021.

[23] Tae Ha Park and Simone D'Amico. Adaptive neural network-based unscented kalman filter for robust pose tracking of noncooperative spacecraft. 6 2022.

[24] Tae Ha Park and Simone D'Amico. Bridging domain gap for flight-ready spaceborne vision, 2024.

[25] Tae Ha Park and Simone D'Amico. Robust multi-task learning and online refinement for spacecraft pose estimation across domain gap. *Advances in Space Research*, 73(11):5726–5740, June 2024.

[26] Tae Ha Park, Marcus Märtens, Gurvan Lecuyer, Dario Izzo, and Simone D'Amico. Speed+: Next-generation dataset for spacecraft pose estimation across domain gap. 10 2021.

[27] Juan Ignacio Bravo Perez-Villar, Alvaro Garcia-Martin, Jesus Bescos, and Juan C. SanMiguel. Test-time adaptation for keypoint-based spacecraft pose estimation based on predicted-view synthesis. *IEEE Transactions on Aerospace and Electronic Systems*, 2024.

[28] Frank Schnitzer, Arne Sonnenburg, and Manuel Sanchez Gestido et al. Lessons-learned from on-ground testing of image-based non-cooperative rendezvous navigation with visible-spectrum and thermal infrared cameras.

[29] Steven M Seitz and Charles R Dyer. Physically-valid view synthesis by image interpolation, 1995.

[30] Sumant Sharma and Simone D'Amico. Pose estimation for non-cooperative rendezvous using neural networks. 6 2019.

[31] Sumant Sharma, Tae Ha Park, Dario Izzo, Marcus Märtens, and Simone D ' Amico. Satellite pose estimation challenge: Dataset, competition design and results.

[32] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition, 2020.

[33] Qitai Wang, Lue Fan, Yuqi Wang, Yuntao Chen, and Zhaoxiang Zhang. Freevs: Generative view synthesis on free driving trajectory, 2024.

[34] Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13:600–612, 4 2004. SSIM.

[35] Zi Wang, Minglin Chen, Yulan Guo, Zhang Li, and Qifeng Yu. Bridging the domain gap in satellite pose estimation: a self-training approach based on geometrical constraints, 2022.

[36] Jiale Xu, Jia Zheng, Yanyu Xu, Rui Tang, and Shenghua Gao. Layout-guided novel view synthesis from a single indoor panorama, 2021.