ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Advances in Space Research xx (2023) xxx-xxx

ADVANCES IN SPACE RESEARCH
(a COSPAR publication)

www.elsevier.com/locate/asr

Domain randomisation and CNN-based keypoint-regressing pose initialisation for relative navigation with uncooperative finite-symmetric spacecraft targets using monocular camera images

Karl Martin Kajak[a,b,*], Christie Maddock[b], Heike Frei[a], Kurt Schwenk[a]

[a] DLR, Münchener Str. 20, 82234 Weßling, Germany
[b] University of Strathclyde, 75 Montrose St, Glasgow G1 1XJ, United Kingdom

Received June 28, 2023;

Abstract

Vision-based relative navigation technology is a key enabler of several areas of the space industry such as on-orbit servicing, space debris removal, and formation flying. A particularly demanding scenario is navigating relative to a non-cooperative target that does not offer any navigational aid and is unable to stabilise its attitude. This research integrates a convolutional neural network (CNN) and an EPnP-solver in a pose initialisation system. The system's performance is benchmarked on images gathered from the European Proximity Operations Simulator EPOS 2.0 laboratory. A synthetic dataset is generated using Blender as a rendering engine. A segmentation-based pose estimation CNN is trained using the synthetic dataset and the resulting pose estimation performance is evaluated on a set of real images gathered from the cameras of the EPOS 2.0 robotic close-range relative navigation laboratory. It is demonstrated that a synthetic-image-trained CNN-based pose estimation pipeline is able to successfully perform in a close-range visual relative navigation setting on real camera images of a 6-facet symmetrical spacecraft.

Keywords: close-range relative navigation; pose estimation; symmetric uncooperative target; monocular camera; convolutional network ; domain randomisation

1. Introduction

Object pose estimation is usually an important component of close-range visual relative navigation with uncooperative targets. The term ”pose” can refer to the relative position and attitude of an object or just one of these. Visual relative navigation with uncooperative targets is likely an important technology in the future, especially for active debris removal efforts and potentially also for in-orbit servicing. Generally, two tasks can be distinguished in pose estimation systems: initialisation and tracking. Initialisation refers to a situation where no previous information exists about where a sought target is in the sensor field of view. Oestreich et al. (2020), for example, present work on pose initialisation with CNNs. Tracking refers to a situation where previous estimates are

[*] Corresponding author: Karl Martin Kajak
Email addresses: karl.kajak@dlr.de (Karl Martin Kajak), christie.maddock@strath.ac.uk (Christie Maddock), heike.frei@dlr.de (Heike Frei), kurt.schwenk@dlr.de (Kurt Schwenk)

https://dx.doi.org/10.1016/j.jasr.xxxx.xx.xxx
0273-1177/© 2023 COSPAR. Published by Elsevier Ltd All rights reserved.

available (after initialisation or from previous tracking steps) and so the search space for an estimate is smaller. Kelsey et al. (2006) for example presents a tracking-oriented work for relative navigation.

Sensor and algorithm technologies are the backbone of close-range visual relative navigation. Multiple alternatives for sensors have been considered for the task of close-range relative navigation, such as monocular cameras (Sharma et al., 2018), LIDAR sensors (Nocerino et al., 2020), and PMD cameras (Klionovska et al., 2018) among others. Monocular cameras are an attractive option due to their accessibility and widespread use. Technology demonstrations so far have handled close-range relative navigation with monocular cameras via algorithms relying on solutions like template matching (Hannah, 2008) or image processing algorithms (Queen et al., 2008), perhaps seeking to fit lines of a wireframe model to the lines found in the image, for example. However, the general computer vision field has also focused on the application of CNNs for pose estimation (Hodan et al., 2020), also for navigational tasks (Wu et al., 2019). CNNs are a class of artificial neural networks (ANN), typically applied for computer vision problems. They are based on convolution kernels or filters that slide along input features, producing translation-equivariant responses also known as feature maps.

Utilising CNNs for visual relative navigation in space would potentially offer several advantages over the state of the art. To start with, they have the potential to be applicable to targets of various appearances with different dominant visual features, not just ones that have distinguishable lines or silhouettes, which would be necessary for pose estimation systems relying on line filters or silhouette templates. Furthermore, they could potentially operate in various lighting conditions. If this flexibility can be capitalised on, CNNs could form a basis for a general visual relative navigation system that does not require a new integration for every new target with a possibly different appearance. This paper focuses on the integration of a state-of-the-art keypoint-regressing CNN into a visual relative navigation system.

Using CNNs for keypoint regression usually means that the CNN is trained to predict coordinates in the 2D image space. This is done, because keypoints are easier to regress than quaternions, for example. 2D keypoints exist in a continuous Euclidean space, conforming to the continuous Euclidean output space of CNNs, whereas rotation spaces are non-Euclidean and non-linear. In some cases, like with quaternions, the representation is ambiguous, with multiple different quaternions referring to the same rotation (Saxena et al., 2009). Furthermore, rotation representations in real Euclidean spaces of less than five dimensions have been shown to be discontinuous (Zhou et al., 2019). CNNs can only accurately predict continuous spaces (Llanas et al., 2007).

Keypoint-regression-based approaches to pose estimation have been the most accurate approaches featuring CNNs for monocular camera images. For example, this claim is supported by the fact that the top solutions to ESA Pose Estimation Challenge 2019 were keypoint-regression-based (Kisantal et al., 2020). In summary, this competition challenged entrants to estimate the relative position and orientation of a target spacecraft depicted in both synthetic and real monocular camera images.The focus is on the pose initialisation task, as it is the first step of any close-range relative navigation task.

The CNN used here was developed by Hu et al. (2019), which achieved the top score in the ESA Pose Estimation Challenge 2019 on real camera images of spacecraft while having been mostly trained on synthetic images. This is desirable as the aim is to train fully on synthetic images and to achieve robust performance on real camera images.

The issue with training on synthetic images and evaluating on real images is that usually the target and the environment appears different. This difference between the two sets of images is also known as the domain gap problem. Often the light sources, shadowing, image dynamic range, materials, and background are different between synthetic and real images. There are two possible approaches to overcoming this difference: input data manipulation or the adoption of a CNN that has been designed to overcome this issue. When it comes to input manipulation, one can attempt to create a synthetic image dataset that is as similar as possible to the real images (Brochard et al., 2018). This is a non-trivial effort; it is difficult to do when there is limited or no access to real images. Even if real images with the targeted camera are available, they might not cover much of the pose space or lighting conditions necessary to ensure robust operation in these unseen pose or lighting conditions. Furthermore, knowledge about the target objects and the environment could also be limited. Attempting to create photorealistic images of the target object might require knowledge about its materials, exact physical configuration, the lighting, and the rest of the environment, which also participates in the lighting of objects in the scene. However, instead of trying to close the domain gap between the synthetic and real images, success has been found in going in the other direction. Domain randomisation is a principle of trying to present as many different domains as possible in the training image dataset such that the trained CNN learns to see the common features, thus hopefully being applicable also for the image domain where the CNN will eventually be applied. This principle has also been applied for CNNs performing navigational tasks (Loquercio et al., 2020).

Figure 1 depicts an exemplary target for close-range relative navigation with a monocular camera. This target exhibits symmetric shape geometry, which poses an immediate issue when trying to adapt the CNN described in Section 3.4 to this target spacecraft. Figure 1 shows the keypoints that are estimated in the experiments in these sections. Ideally, each 2D image space coordinate predicted by the CNN would always have a fixed correspondence, or in other words, estimate the same uniquely identifiable specific fixed keypoint. However, it is clear from Figure 1 that each of the six keypoints on the main hexagonal body of the spacecraft are not uniquely identifiable as each of the six corners looks very similar to the other ones. However, the real physical mock-up spacecraft in the robotic setup and simulation of course has a unique, single pose mathematically. The problem reveals itself during training of the CNN - trying to optimise the weights of the CNN to predict these visually indistinct keypoints leads to the CNN estimating their average. The six hexagonal body keypoints as shown in Figure 1 would all end up in the centre of the plane they belong on as each visually ambiguous keypoint target drags the predictions in different directions, ultimately having a loss minimum in the centre of the hexagonal front plate. Essentially, one would be trying to teach the CNN to estimate a one-to-many relationship, which it is not prepared to do. In this paper, the issue is tackled with a modified loss function that uses the closest pose among the manually constructed set of ambiguous plausible poses to train the CNN.

In the present work, the term ”pose” is used often, but the focus is mainly on the relative attitude estimation, though the presented methods also provide relative position estimates. This is due to the rotational space being more difficult to estimate, especially due to the domain gap and symmetric target.

The contributions of the paper are as follows. The chosen CNN is trained on domain-randomised synthetic images generated via the open-source software Blender, using a novel loss function. The performance of the resulting CNN is demonstrated on real images gathered from the EPOS 2.0 laboratory with different sets of manually chosen keypoint configurations. Similar accuracy is achieved on synthetic and real images without taking into account any information about the real images.

The paper is structured as follows. Section 2 starts with an overview of the background and existing research in the field. Section 3 expands on the specific methods used in this work. Section 4 presents experiments that have been undertaken with the new CNN-based pose initialisation system and discusses the results. Future work to improve the presented methods is detailed in Section 5. Section 6 concludes the paper.

2. Background and State of the Art

CNNs are starting to be incorporated into visual relative navigation systems in a variety of ways. The same CNN used in the present work was also used by Gerard (2019) in the ESA Pose Estimation Challenge 2019 to rank second in the leaderboard, with the highest score on real images. The post-competition analysis in Kisantal et al. (2020) found that all 20 submitted solutions used deep learning as a part of their pose estimation pipeline.

Sharma et al. (2018) applied a CNN to the task of spacecraft pose estimation from images. The paper documents two primary contributions; the first is a CNN intended for pose initialisation. The CNN is set up to fit a classification task, so the output of the last layer is used in a softmax loss function, which produces a distribution over the class labels. In other words, it is a viewpoint classifer. The classes are tied to a discretisation of the relative attitude space such that each class refers to a particular attitude (a single rotation matrix, for example). During training, the closest pose in terms of the discretized pose space is chosen as the ground truth for the training images. The second major contribution of the paper is the development of a synthetic image generation pipeline to train the CNN. The pipeline is intended to produce an abundance of images that represent noise, colour, and illumination characteristics expected in orbit. The solution is suitable for symmetric objects, in the sense that images containing ambiguous viewpoints will yield some class probabilities that reflect uncertainty between the plausible pose estimates. Of course, the issue remains then that one of these ambiguous solutions will have to be chosen as an initial pose estimate for downstream systems that rely on it. This could be a Kalman filter, for example. Essentially, the ambiguity is passed downstream from the CNN. If this pose estimate would be naively connected to a state estimator, it would seriously disturb the accuracy of the state estimate every time the pose initialiser jumps to an alternative plausible pose estimate. When it comes to closing the domain gap, Sharma et al. (2018) do not explore the robustness of the trained CNN with respect to domain differences. The test dataset is independent from training and validation datasets, but they are sampled from the same domain.

Pasqualetto Cassinis et al. (2022) deploys a CNN-based pose initialization system on monocular camera images from ESA's GNC Rendezvous, Approach, and Landing Simulator (GRALS) testbed. In particular, they manage to achieve successful evaluation on more than 50 % of real camera images after training the CNN on synthetic images. The CNN relies on estimating 2D heatmaps of chosen keypoints on the ENVISAT spacecraft's structure, with one heatmap belonging to each keypoint. The domain generalization is achieved via spacecraft texture, lighting, and background randomization, in addition to standard data augmentation methods. However, it is found that this generalization comes with the caveat of heatmaps starting to predict multiple keypoint locations due to ambiguity. This happens as the network no longer has access to as many specific visual details that could resolve ambiguous poses to one unique pose.

Oestreich et al. (2020) explore questions such as the necessary quantisation of the pose space for classification-based pose estimation, the necessary amount of training images for it, as well as lighting condition impact on pose estimation performance.

It is also worthwhile to look at the development of pose initialisation systems in the general computer vision field. In the process, we will also discuss how and if they consider symmetric objects.

Kehl et al. (2017) use a viewpoint classifier to obtain a pose estimate. Viewpoint classifiers are able to tolerate better a one-to-many type problem as arises with viewpoint ambiguity, but still there are convergence problems. The one-to-many problem means that the same or a very similar input is given different output labels. To circumvent these, the authors select a subset of viewpoints during training such that ambiguous redundant poses are removed. This still requires manual intervention for every new type of object.

Rad & Lepetit (2017) solve this issue for a keypoint-regressing CNN by restricting the keypoint labels to a pose interval where no ambiguity is present. The absolute pose is then obtained by a separate classifier that tries to determine which of the ambiguous pose intervals is present (if at all distinguishable) and this is taken into account in the final pose estimate. This process is not preferable as the intervals are manually designed and every new target would require a new consideration. Furthermore, this does not work for objects with infinite ambiguous viewpoints such as a cylinder.

Corona et al. (2018) use an embedding via a CNN and then select the closest latent space vector from an offline generated database of discrete viewpoints as the final pose estimate using cosine similarity. Latent space refers to a low-resolution representation of features in the narrowest part of the CNN, which contains high-level information about the processed input. However, they additionally classify the order of the symmetry. This symmetry classification still requires manual intervention to label symmetry orders for the target object.

Xiang et al. (2018) approach symmetric objects by using a loss function that focuses on the closest vertices of the 3d model of the object being in agreement between estimate and ground truth orientations. Unfortunately, this approach comes with local minimums during training.

Sundermeyer et al. (2018) presents a method for estimating the pose of an object with a CNN that uses the CNN to encode an image into a latent space vector. This vector then compared to an offline generated database of viewpoints using cosine similarity, yielding a pose estimate. The database format is inherently able to handle ambiguous poses as there is no conflict between storing very similar embedding vectors with different output viewpoints.

Park et al. (2019a) uses a principle they call 'transformer loss' to train their CNN to estimate the pose of a symmetric target. Essentially, the keypoints of the closest plausible ambiguous pose to the predictions of the CNN are chosen for each image in a training batch.

Manhardt et al. (2019) propose a fairly general method where a high number of redundant poses are predicted for an object. Due to the high number of redundancy, the output is expected to reflect a distribution of poses that are plausible given a particular input image. However, the suitability of this approach depends on the output of the CNN. This approach works quite well if the output of the CNN is a rotation representation directly, such as a quaternion. However, many modern pose estimation frameworks actually estimate 2D image space keypoints, which are then resolved to a pose via a Perspective-n-Point solver. However, Manhardt's approach gets complicated when a set of keypoints are estimated. The process gets more intensive as more keypoints are used, and then there is the difficulty of having to move from 2D image space to pose space with all these redundant keypoint regressions.

Hodan et al. (2020) present a new approach to the pose estimation of symmetric objects. Essentially, a CNN is used to segment the surface of the target object in the image into patches. To reiterate, the surface of the target object is divided into some fragments of the whole, and these are then classified in the image per pixel. Per each identified fragment, a regressor is used to estimate the 3D coordinate of the fragment's centre. These 2D-to-3D correspondences are then used with the Progressive-X scheme (Baráth & Matas, 2019) incorporating a robust and efficient PnP-RANSAC algorithm (Fischler & Bolles, 1981). In addition to inherently dealing with symmetric objects, this approach has the added value of not needing specific visual features on the object to relate keypoints to. Rathinam & Gao (2020) compare a viewpoint classifier and an architecture that features an object detector for initial image cropping, a keypoint regressor, and a PnP-solver. They note that to achieve similar results, the viewpoint classifier requires approximately 6-10 times more trainable parameters. However, they do stress that this depends heavily on the setup. Furthermore, this does not take into account the PnP-solver that is still necessary to move from keypoints to an actual pose.

Domain randomisation has been demonstrated to be an effective tool for enabling synthetic-image-trained CNNs to perform in real world conditions in different navigation and movement tasks of different fields. Tobin et al. (2017) demonstrated that an object detector trained on synthetic domain-randomised images can form an effective basis for robotic grasping in real world conditions. Loquercio et al. (2020) demonstrate an autonomous racing drone navigation system trained on domain-randomised synthetic images that is able to fly through a real-world gated racetrack at high speeds.

3. Methods

This section presents the methods used to conduct the research. First, the tools and settings of the synthetic image production pipeline are explained in Section 3.2. Following that, the laboratory yielding real camera images of physical spacecraft is described in Section 3.3. Lastly, the details of the pose estimation pipeline incorporating the CNN are presented in Section 3.4.

3.1. Datasets

For ease of reference, the datasets have been described and assigned unique identifiers in this section. This is to prevent confusion in the description of the experiments. The details of the production of the synthetic datasets are explained in Section 3.2. For completeness, all parameters have also been summarized in Tables 1 and 2.

- **Training dataset A** Includes 15000 images rendered with Blender. Target spacecraft rotations are randomly sampled from the entire pose space. Target range random between 5 - 30 m. Lateral relative displacement random, while being limited to within 1.3 m of the limits of the camera field of view. The largest dimension of the spacecraft is 1.32 m from the centre of rotation, so this limitation ensures the spacecraft is in full view of the camera. The direction of the Sun is randomised. The orientation of the background is randomised.
- **Training dataset B** Includes 15000 images rendered with Blender. The differences with respect to dataset A will be described. Target ranges are now not sampled uniformly with respect to distance, but rather with respect to the size of the target in the image. This is an attempt to improve accuracy at closer ranges as otherwise images where the spacecraft dominates the space of the image are underrepresented. Furthermore, multiple other factors are varied as a part of the domain randomisation scheme. Additionally varied parameters include camera exposure time, background texture brightness, background and target surface texture mixing with random 'Magic' and 'Voronoi' procedural textures of Blender as well as the parameters of these textures, Sun illumination strength, target material colours, roughness, and metallicness, as well as a random Gaussian blur on the images.
- **Test dataset C** Includes 760 images rendered with Blender. Target spacecraft rotations cover entire pose space with 20 degree spacing in all three Euler angles. In terms of the rotation angle around symmetric axis, only a space between two ambiguous poses are covered (60 degree range due to the six-fold finite symmetry of the main hexagonal body). Target range is fixed at 5 meters and is not laterally displaced. The additional domain randomisation variations present in training dataset B are not featured here. Other parameters (lighting, materials, etc) are equivalent to training dataset A.
- **Test dataset D** Same as test dataset C, but at a fixed target range of 17.5 meters.
- **Test dataset E** Same as test dataset C, but at a fixed target range of 30 meters.
- **Test dataset F** This dataset is composed of 3346 real images of a physical target spacecraft mock-up from DLR EPOS 2.0 laboratory at various ranges at less than 25 m. Rotations and positions conform to samplings from representative approach and inspection manoeuvres. Lighting conditions vary, also exhibiting some extremely overlit images. Not a lot of lateral displacement is exhibited as target is mostly centred in camera view.
- **Test dataset G** Synthetic reproductions of poses in test dataset F. Lighting conditions randomised.

Parameter	Dataset A	Dataset B	Dataset C, D, E	Dataset F	Dataset G
Number of images	15000	15000	760	3346	3346
Origin	Blender	Blender	Blender	DLR EPOS 2.0	Blender
Rotation space sampling	Random across entire pose space	Random across entire pose space	Uniformly distributed across entire pose space	Representative approach and inspection maneuver	Representative approach and inspection maneuver
Range sampling	Random between 5-30 m	Random diameter of object, belonging to range 5-30 m	5 m, 17.5 m, and 30 m, respectively, for C, D, E datasets	5.8 - 17 m	5.8 - 17 m
Lateral displacement	Within camera FOV	Within camera FOV	None	Not significant	Not significant
Direction of the Sun	Random	Random	Random	Fixed	Random
Sunlight intensity	”Sun”-type parallel light source with $10W/m^2$ power	”Sun”-type parallel light source with power varying between $0 - 100W/m^2$	”Sun”-type parallel light source with $10W/m^2$ power	”Sun”-type parallel light source with $10W/m^2$ power	”Sun”-type parallel light source with $10W/m^2$ power
Background texture	Random translation and rotation of orbital environment image	Variable-strength mixture of randomly translated, rotated orbital environment image, randomly designed ”Voronoi” and ”Magic” texture patterns, and complete darkness	Random translation and rotation of orbital environment image	DLR EPOS 2.0 laboratory backdrop	Random translation and rotation of orbital environment image
Background brightness	Fixed	Random	Fixed	Fixed - laboratory backdrop	Fixed
Camera exposure time	Blender ”Film exposure” parameter set to 1.0, non-physical unit	Blender ”Film exposure” parameter varies between 0 - 1.0, non-physical unit	Blender ”Film exposure” parameter set to 1.0, non-physical unit	Fixed 5000 μ s	Blender ”Film exposure” parameter set to 1.0, non-physical unit
Camera resolution	2048x2048	2048x2048	2048x2048	2048x2048	2048x2048
Camera field of view	47.6° horizontally and vertically	47.6° horizontally and vertically	47.6° horizontally and vertically	47.6° horizontally and vertically	47.6° horizontally and vertically

Table 1: Summary of the dataset parameters (1/2)

Parameter	Dataset A	Dataset B	Dataset C, D, E	Dataset F	Dataset G
Spacecraft materials	MLI Foil: Principled BSDF, HSV color [1,0.375,0], metallicness 1.0, roughness 0.2. Solar panel: Glossy BSDF, HSV color [0.621,1,0.321], roughness 0.0. Structure: Principled BSDF, HSV color [0,0,1], metallicness 0.0, roughness 1.0	MLI Foil: Principled BSDF, HSV color random, metallicness random between 0.8 - 1.0, roughness random between 0 - 0.2. Solar panel: Glossy BSDF, HSV color random, roughness random between 0 - 0.2. Structure: Principled BSDF, HSV color random, metallicness random between 0.8 - 1.0, roughness random between 0 - 0.2	MLI Foil: Principled BSDF, HSV color [1,0.375,0], metallicness 1.0, roughness 0.2. Solar panel: Glossy BSDF, HSV color [0.621,1,0.321], roughness 0.0. Structure: Principled BSDF, HSV color [0,0,1], metallicness 0.0, roughness 1.0	Real, representative materials	MLI Foil: Principled BSDF, HSV color [1,0.375,0], metallicness 1.0, roughness 0.2. Solar panel: Glossy BSDF, HSV color [0.621,1,0.321], roughness 0.0. Structure: Principled BSDF, HSV color [0,0,1], metallicness 0.0, roughness 1.0
Gaussian blur	7x7 filter, σ up to 0.5	7x7 filter, σ up to 0.5 + up to 5x5 filter size and radius in Blender	None	None	None
HSV-colorspace pixel brightness variation	Up to ±10% in Hue, ±50% in Saturation, ±50% in Value	Up to ±10% in Hue, ±50% in Saturation, ±50% in Value	None	None	None
Random added noise	From normal distribution, σ = 0.1	From normal distribution, σ = 0.1	None	None	None

Table 2: Summary of the dataset parameters (2/2)

3.2. Production of synthetic camera images of spacecraft in relative navigation setting using Blender

The open-source 3D-rendering software Blender is used to generate simulated synthetic images of a representative target spacecraft in relative navigation scenarios, as if a chaser spacecraft was observing it. A few sample images are shown in Figure 2.

3.2.1. Rendering engine

The images are rendered using a ray-tracing engine Cycles . The use of a ray-tracer allows the generation of more realistic shadows and reflections, which are important in relative navigation settings. Often the only source of light in the orbital environment is the Sun and therefore significant parts of the spacecraft can be obscured by shadows. Also, many spacecraft incorporate highly reflective materials and components such as solar panels or Multi-Layer Insulation (MLI).

3.2.2. Background environment

The background of the rendered environment varies depending on the experiments. Training dataset A described in Section 3.1 and used in the first experiment described in Section 4.1 merely feature a static panoramic image taken at an Earth orbit position, produced using SpaceEngine (Romanyuk, 2011). SpaceEngine is a realistic virtual universe simulator. It is an HDR-image, so the stars and Sun are visible simultaneously. This is perhaps not realistic, but it adds more visual information to reflect off of the MLI and the solar panels. The particular height of the orbital position or the lighting environment does not matter too much as the background is merely added to provide somewhat realistic reflections for training. The orientation of the background is randomised along all three axes. Training dataset B described in Section 3.1 and used in the second experiment described in Section 4.2 features modifications to this background baseline. The first modification is that the brightness of the background is allowed to vary between fairly bright and entirely black. The second modification is that Blender's procedural textures *Magic* and *Voronoi* are also mixed in with the SpaceEngine-produced environment image at randomly varying strengths.

Figure 3 provides the Blender setup for generating a background environment. The background environment essentially is composed of multiple shaders. The background environment is a sphere around the spacecraft object onto which a texture is projected. This background texture is a mixture of an Environmental Texture Node output, a Magic Texture Node output, a Voronoi Texture Node output, and a completely black shader. The Environmental Texture Node takes in the equirectangular projection image given in Figure 4 and projects it onto the encompassing background sphere. This image was generated using SpaceEngine. The observing position for this image is near Earth, on a line between the Earth and the Sun. The comprehensive world environment shader emits light into the scene such that the spacecraft reflects light from the Sun, the darker areas, and the Earth itself. The Mix Shader nodes enable mixing the four distinct textures, allowing variations such as a completely dark background, a pure Earth environment texture, a background with randomised Voronoi and/or Magic texture, or any combination in between. The purpose is of this design is to provide the capability to produce dark images, realistic images, and randomised images all at once. The randomisation textures Magic and Voronoi are there to make it more difficult for the CNN to learn a specific surrounding as the CNN is intended to work in any background environment. Figure 5 shows samples of Voronoi and Magic Texture Node outputs. The spatial frequencies, colours, and pattern configurations are variable via the Node parameters.

3.2.3. Lighting and camera sensor

There is a single light source in the scene besides the background which also emits light. This is a parallel ray light source to mimic the Sun. For training dataset A merely the direction of the Sun is randomised. For training dataset B the Sun's light emission strength is also varied between a bright value and nearly completely dark value. Furthermore, the camera exposure time is varied in order to provide globally dark or bright images as well.

3.2.4. Spacecraft

The spacecraft is a simplified version of a physical mock-up described in Section 3.3. It exhibits finite symmetry about its longitudinal or docking axis, though at various multiplicities. There are three material groups used in the model. The front hexagonal plate and column are wrapped in MLI, which is mimicked via a golden reflective material. The folds of the MLI are replicated via Blender's procedural Voronoi and Magic textures applied to the surface normal map, which means that light bounces off the surface similarly to how a folded and crinkled foil would. The second material is the solar panel, mounted on the sides of the hexagonal main body of the spacecraft. This is also a reflective material with a dark blue hue. The third material is a glossy white paint featured on the details of the body and docking adapter. The model is simplified in the sense that it lacks certain MLI cutouts that are featured on the physical mock-up as well as some small scale details like screws that are a part of the original complex CAD drawing of the spacecraft. There are a few further discrepancies. One of them is a missing solar panel plate on one side of the hexagonal body, which leaves a gap where one can look at the interior of the mock-up. The second discrepancy is that the front hexagonal railing on the docking adapter has been deformed slightly (a few centimetres out of the plane of the railing) due to impacting the floor in a previous accident. Overall, these differences should not stand in the way of applying a CNN trained with simulated images on real images of the physical mock-up taken with a camera. They are rather viewed as opportunities to see how local differences impact tested CNN solutions. The above-described setup is used for training dataset A and testing datasets C, D, E, and G. Training dataset B comes with further modifications, though. More precisely, the three materials are randomised per image in various ways. The randomly varied parameters include material colour, metallicness, and reflectivity. Furthermore, random Magic and Voronoi textures with randomly varied magnitudes and other parameters governing the appearance of the procedural textures are mixed in with the base materials to increase the unpredictability of the surface textures.

3.3. Pseudo-real images of physical spacecraft mockups in relative navigation setting and EPOS 2.0 laboratory

The intention of this research is to deploy -CNN-based relative navigation system in realistic settings after training them with synthetic images. The DLR EPOS 2.0 laboratory facilitates the imaging of physical scale model mock-ups with real cameras as well as closing the loop and using computer vision to guide and control relative navigation manoeuvres (Benninghoff et al., 2017). The facility also features the simulation of orbit and attitude dynamics for both chaser and target spacecraft. Training dataset B uses real images recorded at EPOS during representative close range approach and flyby manoeuvres.

3.4. Pose estimation using segmentation-based keypoint regression CNN

A full relative navigation system often requires target object position and orientation estimates to function. A full relative navigation system usually composes of multiple components. For example, there might be a component that determines target object pose information without prior state information, or a pose initialiser. This information could then be fed to a state estimator like a Kalman filter to take advantage of the information contained in previous estimates.

The details of the utilised CNN solution are described in Hu et al. (2019). This publication does not explicitly describe the layers of the CNN and the application of the network to the ESA Kelvins competition was only documented via a presentation. The specific structure of the network is implied by the software repository of the method as received from the author. This repository is not public. Therefore, the CNN layers are given in Table 3. To understand the architecture, a brief explanation of the layers is given here.

A convolutional layer in PyTorch (Paszke et al., 2019) applies 2D convolution filters over an input signal composed of several input channels. Let N be the batch size, C the number of channels, H and W be the height and width of input planes in pixels, respectively. If monocular camera images are processed by a CNN, the input plane is the input image only in the first convolutional layer, after that it will be the output feature maps of the previous layer. In the simplest case, the output value of the layer with input size (N, C_{in}, H, W) and output size $(N, C_{out}, H_{out}, W_{out})$ can be determined by,

$$out(N_i, C_{out_j}) = bias(C_{out_j}) + \sum_{k=0}^{C_{in}-1} weight(C_{out_j}, k) \star in(N_i, k) \tag{1}$$

where $\star$ is the 2D cross-correlation operator. The widths, heights, and channels of an input and output tensor are not necessarily the same, which is why a distinction has been made with subscripts. The stride of a convolutional filter refers to the amount of skipped convolutions when the filter slides over an image. With a stride of 1, the convolutional filters shift by one pixel to calculate the next output, whereas with a stride of 2, they shift by two pixels. Stride allows a resolution reduction for the output feature map compared to the input. For example, if a 3-channel image of size 320x320 is the input tensor to a layer with 54 1x1 convolutional filters at a stride of 2 are applied to it, the output tensor has a width of 160, a height of 160, and 54 channels.

The residual layer maps the feature from before each block (note how Table 3 divides the architecture into repeating boldly outlined blocks) are summed with the feature maps of the layer immediately before the residual layer. To learn about the benefits of residual layers, one can refer to (He et al., 2016), for example. These connections in the CNN allow the use of deeper CNNs as the gradients propagate more easily through the CNN during back-propagation.

The deconvolutional layer applies a 2D transposed convolution operator over an input image composed of several input planes. This module can be seen as the gradient of the convolutional layer with respect to its input. It is also known as a fractionally-strided convolution or a deconvolution (although it is not an actual deconvolution operation as it does not compute a true inverse of convolution). More details on these layers can be found in (Zeiler et al., 2010).

Route layers refer to the concatenation of certain previous feature maps to the current last feature maps. These feature maps have to be the same size. For example, a `route-to-16x16` layer refers to the concatenation of the last feature maps that had that size to the layers immediately before this route, which also have to have the same size.

This CNN architecture was selected given the good performance evaluated using real images of the Tango spacecraft of the PRISMA mission. This aligns with the goal of applying this CNN on the pseudo-real robotic laboratory images. On its input side, the CNN consumes a three-channel colour image of size 256x256 pixels. The CNN outputs a 3D-tensor. Two dimensions of this tensor correspond to the 2D spatial dimensions of the image, essentially dividing the input image into a lower-resolution grid. The third dimension contains the 2D normalised image coordinates of the estimated keypoints as well as two probabilities belonging to ”spacecraft” or ”not spacecraft” classes. This way, each grid cell or ’pixel’ in the output side contains its own estimate for all of the keypoints as well as the class probabilities. This means that there is a large amount of redundancy with respect to the predicted keypoints as each output subpixel of the output tensor produces a complete set of keypoints predictions and a strategy is needed to condense these to a single set of image coordinates for each of the keypoints. In the present work, the predictions of all cells labeled *spacecraft* are used with the OpenCV implementation of EPnP (Lepetit et al., 2009) to obtain the 3D coordinates of the keypoints.

The original approach as published in (Hu et al., 2019) predicts the 2D coordinates of a 3D bounding box around the target object projected into the image. However, the solution that was used for the ESA Pose Estimation Challenge 2019 featured a modification where the predicted keypoints belonged to the geometry of the spacecraft body (Gerard, 2019). This approach was also used by the solution that scored highest on the synthetic dataset in the challenge (Chen et al., 2019), with the justification that it creates a stronger relationship between the keypoints and specific geometric features of the spacecraft. The same approach is adopted in the present work.

		Layer type	Filters	Size / Stride	Output
Darknet-53 from (Redmon & Farhadi, 2018)		Convolutional	32	3x3	256x256
		Convolutional	64	3x3 / 2	128x128
	1x	Convolutional	32	1x1	
		Convolutional	64	3x3	
		Residual			128x128
		Convolutional	128	3x3 / 2	64x64
	2x	Convolutional	64	1x1	
		Convolutional	128	3x3	
		Residual			64x64
		Convolutional	256	3x3 / 2	32x32
	8x	Convolutional	128	1x1	
		Convolutional	256	3x3	
		Residual			32x32
		Convolutional	512	3x3 / 2	16x16
	8x	Convolutional	256	1x1	
		Convolutional	512	3x3	
		Residual			16x16
		Convolutional	1024	3x3 / 2	8x8
	4x	Convolutional	512	1x1	
		Convolutional	1024	3x3	
		Residual			8x8
Head from (Hu et al., 2019)	2x	Convolutional	512	1x1	
		Convolutional	1024	3x3	8x8
		Convolutional	512	1x1	8x8
		Convolutional	256	1x1	8x8
		Deconvolutional	256	2x2 / 2	16x16
		Route to 16x16			
	2x	Convolutional	256	1x1	
		Convolutional	512	3x3	16x16
		Convolutional	256	1x1	16x16
		Convolutional	128	1x1	16x16
		Deconvolutional	128	2x2 / 2	32x32
		Route to 32x32			
	3x	Convolutional	128	1x1	
		Convolutional	256	3x3	32x32
		Convolutional	2 + 2*# of keypoints	1x1	32x32

Table 3: Architecture of segmentation-based keypoint regression CNN.

In relative navigation scenarios, target spacecraft can appear at various distances from the observing spacecraft. The CNN used in the present work performs more effectively when the input image contains most to all of the spacecraft. Therefore, the target spacecraft is cropped in the input images for the experiments using the ground truth class segmentation image also produced by Blender. In this image, pixels corresponding to the spacecraft have a maximum intensity value and the background pixels have a zero value. This of course means that the location of the spacecraft in the image is presumed to be known. In a full navigation system this is obviously not a realistic expectation and would therefore require a solution such as training the CNN on all kinds of target distances without cropping or using an object detector to crop the image. In this work it is done as a simplification to study the pose estimation pipeline's domain generalisation characteristics and suitability for symmetric target objects.

3.5. Training a keypoint regressor for symmetric targets

The symmetry problem introduced in Section 1 can be solved if the relationship of the input and output of the CNN can be reduced to a one-to-one relationship. This has been done in a simple way in Park et al. (2019a), for example, where during training the closest of all of the ambiguous pose solutions is selected for loss calculations. This should pull the estimated keypoints to one of the six ambiguous pose solutions in this case, though one would not be able to control which of them. A similar approach is adopted in this work, though it has to be adapted with some modifications. It's a preferred solution as it is a simple change to get CNNs otherwise not designed for symmetric objects specifically to perform with symmetric objects. Since the CNN predicts one full set of coordinates and classes for each cell of the subgrid that the output is divided into it yields highly redundant predictions. One has two options then during training - to select the closest ambiguous pose solution for the entire group of keypoint estimates or to let each cell in the subgrid approach its own closest solution.

In the present work, the total loss used during training is a modified version of the loss used by Hu et al. (2019). Two loss components are considered and modelled: a keypoint regression loss, and a focal loss for the class labels. The CNN does not directly regress coordinates in the same coordinate system as the image pixels. Instead, each output subgrid cell predicts a 2D offset vector from the centre of that cell to each keypoint on the spacecraft. Each 2D keypoint g_i on the spacecraft in normalised image space can be expressed as $g_i = c + h_i(c)$, where c is the centre of each cell and $h_i(c)$ is the corresponding 2D offset vector from the centre of the cell. The CNN is trained to regress the 2D offset vector expressed as $h_i(c) = g_i - c$. The keypoint regression loss is therefore given by,

$$\mathcal{L}_{xy,1} = \underset{h_t}{\operatorname{argmin}} \sum_{h \in M} \sum_{i=1}^{N} |h_t - h_p| \tag{2}$$

$$\mathcal{L}_{xy,2} = \sum_{h \in M} \underset{h_t}{\operatorname{argmin}} \sum_{i=1}^{N} |h_t - h_p| \tag{3}$$

where h_t and h_p refer to the training target and predicted 2D keypoint offset vectors from grid cell centres, respectively, M refers to the output subgrid pixels that predict that they are a part of the spacecraft class rather than the background class, N is the set of keypoints on the spacecraft that have been selected for localisation and $| \cdot |$ refers the the L1 loss.

The focal loss for class labels is a dynamically weighted version of cross-entropy, as presented by Lin et al. (2020).

3.6. Error metrics

The error metrics used in the work are unusual from the point of view of comparing CNNs performing a certain task as they are dependent on the target object, but they are useful when trying to determine the performance of this CNN as a part of a navigation system. The focus is on producing metrics that are intuitive for a navigation engineer.

- **Longitudinal axis projection error** In the case of the target object featured in this work, there is one axis that is more important than the others. It is the symmetry axis and also the axis along which one would approach the docking adapter. This error is calculated by projecting the predicted longitudinal axis onto the ground truth longitudinal axis and extracting the angle between them via the cosine law.

- **Lateral axis projection error** The error is calculated the same way as the longitudinal axis projection error, but instead an axis is used that is perpendicular to the longitudinal axis. This error expresses rotation error about the symmetry axis, which is an important quantity to estimate during the approach with the docking adapter.

- **Position error magnitude** The absolute position error magnitude is also calculated to determine the quality of the position estimate.

4. Experiments

This section presents the experiments, their results, and the underlying reasoning for them.

4.1. Experiment 1: baseline pose estimation results with synthetic images

The first experiment focuses on evaluation of the CNN trained using the loss in (2). The CNN is trained on training dataset A and tested on test datasets C, D, and E. The results of the experiment are presented in Figure 7 in terms of the metrics explained in Section 3.6.

The first thing to note is that the loss function attains meaningful results compared to the symmetry-unaware loss as presented by Hu et al. (2019). The results for that arent shown here, but training would collapse all keypoints onto the symmetry axis, which are not valid inputs for the EPnP-solver.

The largest rotational errors occur at close range, while position error is reduced closer to the target.

The lateral axis projection error figure displays four distinguishable peaks. Since we use an axis projection as an input to an arc-cosine function to calculate a rotational error, the six symmetry-caused ambiguous solutions can be distinguished as four separate peaks here. A -60 or 60 degree rotation error about the symmetry axis are both going to show as positive 60 degree lateral axis projection error, and the same for -120 and 120 degrees of error. Thus there are four, not six peaks corresponding to the hexagonal main body of the spacecraft. This demonstrates that the pose estimation system is able to identify one of the six plausible ambiguous poses due to symmetry of the main body of the spacecraft.

It is interesting to note that the ambiguous poses are not identified in a balanced way as some of the lateral axis projection error peaks dominate over others. A similar pattern of the 120-degree error being dominating shows up in later experiments as well. It could be due to the EPnP-solver, as the ordering of the keypoints given by the CNN do not play a role in its inputs.

Another aspect to note is that the highest longitudinal axis error is not actually centred at zero at any distance, and this also shows in later experiments. No experiments have been conducted to study what causes this exactly, but there are a few sources that could contribute to this phenomenon. The keypoint estimates might be biased in a way that shifts pose estimate, or it could also be a result of the EPnP-solver. Furthermore, there are cases where the predicted pose is actually flipped such that the symmetry axis directions are opposite between ground truth and predicted pose. To put it differently, the pose estimation system predicts that it is looking at the back of the spacecraft when it's looking at the front, or vice versa. Both of these phenomenons could be due to the choice of keypoints as for the case of the observer being exactly on the symmetry axis, the keypoint locations on the image are indistinguishable between the cases of looking at the front or back of the hexagonal main body.

Overall, the pose estimation system trained with the loss given by Equation (2) successfully yields one of the plausible symmetry-caused ambiguous poses. However, some outlying poses are seen with reversed symmetry axes.

4.2. Experiment 2: pose estimation results with synthetic images featuring domain randomisation

The second experiment is largely the same as the first one with a few changes. This time, the CNN was trained using loss of Equation (2) on training dataset B, featuring domain randomisation and a different sampling of target ranges, designed to make the apparent size of the target in the pictures uniformly represented in the dataset (more images at closer ranges). The CNN was tested on datasets C, D, and E. The apparent diameter of the target spacecraft was uniformly sampled from an interval that results in relative ranges between 5 to 30 meters. This change was inspired by the results of the first experiment in an attempt to improve accuracy at closer ranges. The results are presented in Figure 8.

The performance in terms of the lateral and longitudinal axis projection errors has not changed in an significant way according to the histogram metrics. The inclusion of more images captured at close range has not had an appreciable effect on improving rotational accuracy at close range. Making the dataset more challenging via domain randomisation has not caused an accuracy drop across any of the tested ranges.

In summary, the experiment is encouraging the use of domain randomisation as it does not necessarily incur an accuracy penalty. Furthermore, the inclusion of more images at close range has not improved close-range accuracy, so the issue does not stem from dataset statistics.

4.3. Experiment 3: a more flexible loss function for a symmetric target object

The third experiment is the same in terms of used datasets as the second experiment - the CNN is trained on dataset A and tested on datasets C, D, and E. However, here the loss given by Equation (3) was used for training.

Compared to the second experiment trained with the first loss function, the outlying longitudinal axis projection error maximums decrease in all cases. Looking at the inlying pose estimates, the accuracy has also improved in all cases, also notably in the close-range 5-meter range case. The improvement is likely due to the less constraining nature of the second loss, allowing all output cells to seek their own closest pose loss minimum. This result has inspired the use of the second loss function in the fourth experiment.

4.4. Experiment 4: estimating pose from real camera images

The fourth and last experiment focuses on evaluation on real camera images. The CNN is trained on training dataset B and evaluated on test dataset F (real camera images) and G (synthetic copies of the poses represented in the real camera image dataset). The loss function given by Equation 3 is used again. The results are shown in Figure 10.

Contrasting Figures 10 a and c reveals the expected result that the relative attitude prediction about the symmetry axis is slightly more accurate for the case of synthetic images, which are a part of the domain that the CNN was trained on. Comparing the longitudinal axis projection errors in Figures b and d show that in the case of the real image dataset, there are pose estimates, where the symmetry axis is pointing in the wrong direction.

The domain randomisation process has been successful in this case as the CNN trained on the domain randomized images of training dataset B gives mostly reasonable pose estimates, similar to how the CNN performs on the synthetic dataset. However, it has also been demonstrated that the domain randomization procedure is inadequate for covering the specific lighting issues related to a too high exposure time setting on the camera sensor as these produce symmetry-axis flipping pose estimation errors.

4.5. Summary of experiment results

Table 4 summarizes the accuracy and recall metrics of the presented experiments 1-4. All errors are calculated with respect to the closest plausible ambiguous pose solution out of the set of six ambiguous poses due to the hexagonal main body structure of the satellite. Recall is determined as the percentage of pose estimates with less than 30° degrees of error. Furthermore, the accuracy metrics are only calculated on pose estimates that pass this accuracy threshold criteria. This threshold was chosen as the maximum plausible error in terms of the set of possible ambiguous poses due to symmetry. Anything larger than this is definitely a case of failure.

Experiment	Lateral axis error [°]	Longitudinal axis error [°]	Quaternion error [°]	Recall
Experiment 1, 5 meter range	2.55 ± 2.68	2.58 ± 2.88	3.27 ± 3.14	100 %
Experiment 1, 17.5 meter range	0.75 ± 0.69	0.77 ± 0.72	0.94 ± 0.80	99.86 %
Experiment 1, 30 meter range	1.14 ± 2.38	1.10 ± 1.97	1.33 ± 2.26	98.81 %
Experiment 2, 5 meter range	2.08 ± 2.36	2.05 ± 1.55	2.66 ± 2.43	99.86 %
Experiment 2, 17.5 meter range	1.17 ± 1.08	1.33 ± 0.93	1.59 ± 1.19	100 %
Experiment 2, 30 meter range	1.35 ± 1.59	1.50 ± 1.22	1.80 ± 1.67	99.73 %
Experiment 3, 5 meter range	1.57 ± 1.01	1.75 ± 1.21	2.13 ± 1.25	99.86 %
Experiment 3, 17.5 meter range	0.91 ± 0.72	1.15 ± 0.82	1.36 ± 0.92	100 %
Experiment 3, 30 meter range	1.04 ± 1.01	1.37 ± 1.87	1.60 ± 1.88	100 %
Experiment 4, real camera images, ranges 6 - 17 meters	1.26 ± 1.56	1.52 ± 2.44	1.81 ± 2.45	71.66 %
Experiment 4, synthetic images, ranges 6 - 17 meters	0.77 ± 0.64	0.89 ± 1.01	1.09 ± 1.00	100 %

Table 4: Summary of the accuracy and recall metrics of the presented experiments 1-4.

4.6. Dissecting outliers from experiments 1-4

Some outlying pose estimates can be seen in the error histograms of all experiments. However, the combined effect of domain randomisation in training images and the more flexible loss function of Equation (3) seem to have a reducing effect in terms of outliers. This section will explore some of the outlying cases seen in experiments 1-4.

The first discovered failure mode is such that the symmetry axis is flipped in the case of some pose estimates, most prominently in the case of the real image dataset F in the fourth experiment. Most of the images in the representative approach image datasets F and G are such that the chaser approaches the target along the symmetry axis towards the front column with the docking adapter. From this viewpoint, the projected keypoints are quite similar for the two relative attitudes where the column is in the front or back. Some adverse lighting conditions with intense reflections cause the CNN to predict the keypoints for the wrong pose as demonstrated in Figure 11. The predicted keypoints fit better the situation where the column with the docking adapter is on the other side of the spacecraft, despite some visible details from the front octagonal rail that clearly correspond to the front of the spacecraft. Admittedly, the camera images with these conditions are perhaps unnecessarily difficult as an exposure correction could mitigate the issue, but it's still desirable for the CNN to not make this mistake in conditions where distinguishing visual details are available to make the correct pose prediction. Figure 12 presents a comparison of the lighting conditions of the spacecraft and the corresponding longitudinal axis error. The lighting of the spacecraft has been measured as the mean of the pixel intensities over the part of the image corresponding to the spacecraft. The coverage of the synthetic dataset is wider in terms of the mean brightness, but that has not been enough to trigger the same sorts of erroneous pose estimates as the real image dataset. It is likely that the synthetic image production pipeline has not been able to suitable reproduce the sort of local lighting issues as seen in the real image dataset. To illustrate this further, Figure 13 presents histograms of pixel intensities belonging to the spacecraft across the entire synthetic and real camera image datasets. It is clear that the real camera image dataset features many images with maximum brightness values, given that the histogram in Figure 13a shows a 40 percent likelyhood of nearly maximum brightness pixels on the spacecraft throughout the dataset. This corresponds with the assessment that the exposure time for the camera has been too high.

The second failure mode is to do with the keypoint estimates from the CNN not agreeing on which symmetrically ambiguous pose to estimate. An example of this failure mode is shown in Figure 14, which shows the failed predictions on a real camera image from the EPOS robotic navigation laboratory. First of all, the predictions of the keypoints belonging to the hexagonal main body

Solution	$e_q[°]$	PnP
Team ”UniAdelaide” (Chen et al., 2019)	0.41 ± 1.50	Yes
Team ”EPFL_cvlab” (Gerard, 2019)	0.91 ± 1.29	Yes
Team ”pedro_fairspace” (Proença & Gao, 2020)	2.49 ± 3.02	No
SLAB baseline (Park et al., 2019b)	2.62 ± 2.90	Yes
Our solution (experiment 3, 5m range)	2.24 ± 3.11	Yes
Our solution (experiment 3, 15m range)	1.36 ± 0.92	Yes
Our solution (experiment 3, 30m range)	1.60 ± 1.88	Yes
Our solution (experiment 4)	1.91 ± 3.34	Yes

Table 5: Comparison of the solution presented in this paper with the solutions of the 2019 Pose Estimation Challenge.

are spread between several ground truth keypoints. Secondly, the predicted keypoint cluster belonging to the tip of the docking column has moved away from the actual tip of the column toward the back of the spacecraft. Most likely, the rather intense lighting conditions on the front of the spacecraft as discussed for the case of the first failure mode has confused the CNN in terms of whether it is seeing the front or back of the spacecraft, and that is why it has predicted an average location between front and back positions for the front column.

The problematic lighting conditions featured in the real camera dataset may not necessarily be encountered in operation as the camera automatic exposure compensation mode has been turned off and is likely set such that the image is overexposed. Furthermore, the camera features a high dynamic range mode that is not used for this dataset. Likely, the CNN would perform better if these more optimal camera modes were used. However, these borderline cases help to understand the limits of the CNN in terms of estimating the pose in various conditions.

4.7. Comparison to state-of-the art

This section compares the performance of the presented method against the performance of the solutions submitted to the ESA Pose Estimation Challenge 2019 as presented in (Kisantal et al., 2020). The mean and standard orientation errors of the challenge solutions and the solution as presented here are presented in Table 5. The comparison is not direct due to a number of reasons. Firstly, the image datasets that the pose initializers were subjected to are different - the Pose Estimator Challenge utilized the SPEED dataset for training and evaluation. The first three solutions of the competition cannot be trained or evaluated on the dataset presented in this work as they are not prepared to handle symmetric targets. The second important difference is that for our solution the rotational error has been calculated with respect to the closest ambiguous pose solution, as the presented solution is yet unable to provide the correct unique pose solution out of the six possible answers. This capability is left for future work. The third important difference is that in the case of the present work, the solution is ”helped” by pre-cropping the images rather than using an object detector as many of the challenge solutions to crop the image. The fourth important distinction is that in the case of our solution, training was stopped at a point where keypoint estimation loss was still reducing, whereas the challenge solutions were likely pushed to limits in terms of the attainable accuracy. Regardless of these differences, comparing the presented solution to the performance of the 2019 Pose Estimation Challenge solutions is valuable in order to show that the performance has not dropped significantly as a result of the loss function modification.

5. Future work

There are further developments planned to improve the presented pose initialisation system. The first drawback of the system is that it yields one of the plausible ambiguous poses due to symmetry. However, the spacecraft does feature a few mechanical components on the docking adapter that allow unique identification of the pose. The next step in this respect is to include functionality that allows to identify these details such as to yield the correct unique pose at least when the chaser is close to the target. The second drawback of the system is related to the application of the pose initialisation system to the real camera images. Here, multiple steps are planned. First, adjustments to the domain randomisation procedure should be explored with the intention to capture the intense local lighting phenomena seen in the real camera image dataset, which were the cause of multiple failure modes. Secondly, it should be explored whether the keypoint predictions can be made more reliable in the cases where the adverse light conditions obscured most of the docking adapter column, yet still revealing some details where a human observer could correctly identify the front side of the spacecraft.

6. Conclusions

The experiments conducted demonstrate that the keypoint-regression CNN combined with a EPnP-solver have the potential to be viable for a spacecraft target exhibiting finite symmetry in a relative navigation setting. It was demonstrated that a domain randomisation procedure enables to train the pose estimation system on synthetic images, and successfully evaluate it on real camera images without a significant loss in pose estimation accuracy. On the other hand, 30 percent of the pose estimates were associated with a 180-degree-flip of the symmetry axis. As these flips were only seen in the real camera images and not the synthetic images, it was concluded that the domain randomisation procedure presented is not adequate for dealing with local overlit regions due to camera exposure time being too high and with the sun reflecting directly into the camera. A modified loss function that allows convergence on selected ambiguous poses due to symmetry enables the CNN to converge on the ground truth keypoints successfully. However, multiple undesirable phenomena were demonstrated in this respect. Firstly, the selection of the keypoints is non-trivial, and can have an impact on nature of the specific failure modes of the system. For example, estimating the corner keypoints on the front octagonal docking railing is less successful than the corner keypoints of the main hexagonal body, because the column railing is sometimes fully hidden and has a certain geometric relationship to the hexagonal body as well. Secondly, a high intensity local reflection off the MLI on the front side of the spacecraft with the docking column could shift the entire entire group of redundant keypoint estimates toward an alternative estimate due to visual ambiguity. This phenomenon makes it clear that the spread of the redundant keypoint estimates is not a good enough indicator that the pose estimate is actually wrong, and therefore an alternative source of uncertainty must be found for self-diagnostic capability. Furthermore, it points to the fact that the redundant estimates are not actually fully independent and all point toward the same problematic answer despite being situated in different locations spatially in the output tensor. In other words, the spatial location of the keypoint estimates does not correlate with a more reliable estimate locally in an area of the input image with better visibility of details of the spacecraft body.

7. Acknowledgements

This work was funded under European Commission H2020 MSCA ITN Stardust-R, grant agreement 813644.

References

Baráth, D., & Matas, J. (2019). Progressive-X: Efficient, anytime, multi-model fitting algorithm. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 3779–3787). doi:10.1109/ICCV.2019.00388.

Benninghoff, H., Rems, F., Risse, E.-A., & Mietner, C. (2017). European Proximity Operations Simulator 2.0 (EPOS) - a robotic-based rendezvous and docking simulator. *Journal of large-scale research facilities JLSRF*, *3*, A107. doi:10.17815/jlsrf-3-155.

Brochard, R., Lebreton, J., Robin, C., Kanani, K., Jonniaux, G., Masson, A., Despré, N., & Berjaoui, A. (2018). Scientific image rendering for space scenes with the SurRender software. URL: https://www.airbus.com/sites/g/files/jlcbta136/files/2021-11/article_SurRenderSoftware_Airbus.pdf Presentation at 69th International Astronautical Congress (IAC) in Bremen, Germany.

Chen, B., Cao, J., Parra, A., & Chin, T.-J. (2019). Satellite pose estimation with deep landmark regression and nonlinear pose refinement. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* (pp. 2816–2824). doi:10.1109/ICCVW.2019.00343.

Corona, E., Kundu, K., & Fidler, S. (2018). Pose estimation for objects with rotational symmetry. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (pp. 7215–7222). doi:10.1109/IROS.2018.8594282.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, *24*(6), 381–395. URL: https://doi.org/10.1145/358669.358692. doi:10.1145/358669.358692.

Gerard, K. (2019). Segmentation-driven satellite pose estimation. https://indico.esa.int/event/319/attachments/3561/4754/pose_gerard_segmentation.pdf. Presentation at ESA Kelvins Day 2019 at ESTEC in Noordwijk, The Netherlands.

Hannah, S. J. (2008). ULTOR (registered trademark) passive pose and position engine for spacecraft relative navigation. In *SPIE Defense and Security Symposium*. volume 6958. doi:10.1117/12.777193.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). doi:10.1109/CVPR.2016.90.

Hodan, T., Baráth, D., & Matas, J. (2020). EPOS: Estimating 6D pose of objects with symmetries. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 11700–11709). doi:10.1109/CVPR42600.2020.01172.

Hu, Y., Hugonot, J., Fua, P., & Salzmann, M. (2019). Segmentation-driven 6D object pose estimation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 3380–3389). doi:10.1109/CVPR.2019.00350.

Kehl, W., Manhardt, F., Tombari, F., Ilic, S., & Navab, N. (2017). SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. In *2017 IEEE International Conference on Computer Vision (ICCV)* (pp. 1530–1538). doi:10.1109/ICCV.2017.169.

Kelsey, J., Byrne, J., Cosgrove, M., Seereeram, S., & Mehra, R. (2006). Vision-based relative pose estimation for autonomous rendezvous and docking. In *2006 IEEE Aerospace Conference* (pp. 2038–2057). volume 1. doi:10.1109/AERO.2006.1655916.

Kisantal, M., Sharma, S., Park, T. H., Izzo, D., Märtens, M., & D'Amico, S. (2020). Satellite pose estimation challenge: Dataset, competition design, and results. *IEEE Transactions on Aerospace and Electronic Systems*, *56*(5), 4083–4098. doi:10.1109/TAES.2020.2989063.

Klionovska, K., Ventura, J., Benninghoff, H., & Huber, F. (2018). Close range tracking of an uncooperative target in a sequence of photonic mixer device (PMD) images. *Robotics*, *7*(1). URL: https://www.mdpi.com/2218-6581/7/1/5. doi:10.3390/robotics7010005.

Lepetit, V., Moreno-Noguer, F., & Fua, P. (2009). EPnP: An accurate O(n) solution to the PnP problem. *International Journal of Computer Vision*, *81*, 155–166. doi:10.1007/s11263-008-0152-6.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2020). Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(2), 318–327. doi:10.1109/TPAMI.2018.2858826.

Llanas, B., Lantarón, S., & Sainz, F. J. (2007). Constructive approximation of discontinuous functions by neural networks. *Neural Processing Letters*, *27*, 209–226. doi:10.1007/s11063-007-9070-9.

Loquercio, A., Kaufmann, E., Ranftl, R., Dosovitskiy, A., Koltun, V., & Scaramuzza, D. (2020). Deep drone racing: From simulation to reality with domain randomization. *IEEE Transactions on Robotics*, *36*(1), 1–14. doi:10.1109/TRO.2019.2942989.

Manhardt, F., Arroyo, D. M., Rupprecht, C., Busam, B., Birdal, T., Navab, N., & Tombari, F. (2019). Explaining the ambiguity of object detection and 6D pose from visual data. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 6840–6849). doi:10.1109/ICCV.2019.00694.

Nocerino, A., Opromolla, R., Fasano, G., & Grassi, M. (2020). Analysis of lidar-based relative navigation performance during close-range rendezvous toward an uncooperative spacecraft. In *2020 IEEE 7th International Workshop on Metrology for AeroSpace (MetroAeroSpace)* (pp. 446–451). doi:10.1109/MetroAeroSpace48742.2020.9160326.

Oestreich, C., Lim, T. W., & Broussard, R. (2020). On-orbit relative pose initialization via convolutional neural networks. In *AIAA Scitech 2020 Forum* AIAA 2020-0457. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2020-0457. doi:10.2514/6.2020-0457.

Park, K., Patten, T., & Vincze, M. (2019a). Pix2Pose: Pixel-wise coordinate regression of objects for 6D pose estimation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 7667–7676). doi:10.1109/ICCV.2019.00776.

Park, T. H., Sharma, S., & D'Amico, S. (2019b). Towards robust learning based pose estimation of noncooperative spacecraft. In *2019 AAS/AIAA Astrodynamics Specialist Conference* (pp. 3667–3686). volume 171.

Pasqualetto Cassinis, L., Menicucci, A., Gill, E., Ahrns, I., & Sanchez-Gestido, M. (2022). 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. doi:10.1016/j.actaastro.2022.04.002.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc.

Proença, P. F., & Gao, Y. (2020). Deep learning for spacecraft pose estimation from photorealistic rendering. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 6007–6013). doi:10.1109/ICRA40945.2020.9197244.

Queen, S., Naasz, B., Burns, R., Eepoel, J., Hannah, J., & Skelton, E. (2008). The HST SM4 relative navigation sensor system: Overview and preliminary testing results from the Flight Robotics Lab. *The Journal of the Astronautical Sciences*, *57*, 457–483. doi:10.1007/BF03321512.

Rad, M., & Lepetit, V. (2017). BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. *2017 IEEE International Conference on Computer Vision (ICCV)*, (pp. 3848–3856).

Rathinam, A., & Gao, Y. (2020). On-orbit relative navigation near a known target using monocular vision and convolutional neural networks for pose estimation. In *2020 International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS)*. URL: https://openresearch.surrey.ac.uk/esploro/outputs/99524421202346.

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *ArXiv*, *abs/1804.02767*. URL: https://arxiv.org/abs/1804.02767. doi:10.48550/arXiv.1804.02767.

Romanyuk, V. (2011). Space Engine - a universe simulator. https://spaceengine.org/. Accessed: 2020-01-05.

Saxena, A., Driemeyer, J., & Ng, A. Y. (2009). Learning 3-D object orientation from images. In *2009 IEEE International Conference on Robotics and Automation* (pp. 794–800). doi:10.1109/ROBOT.2009.5152855.

Sharma, S., Beierle, C., & D'Amico, S. (2018). Pose estimation for non-cooperative spacecraft rendezvous using convolutional neural networks. In *2018 IEEE Aerospace Conference* (pp. 1–12). doi:10.1109/AERO.2018.8396425.

Sundermeyer, M., Marton, Z.-C., Durner, M., Brucker, M., & Triebel, R. (2018). Implicit 3D orientation learning for 6D object detection from RGB images. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer Vision – ECCV 2018* (pp. 712–729). Cham: Springer International Publishing.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 23–30). doi:10.1109/IROS.2017.8202133.

Wu, D., Zhuang, Z., Xiang, C., Zou, W., & Li, X. (2019). 6D-VNet: End-to-end 6DoF vehicle pose estimation from monocular RGB images. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 1238–1247). doi:10.1109/CVPRW.2019.00163.

Xiang, Y., Schmidt, T., Narayanan, V., & Fox, D. (2018). PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. In *Proceedings of Robotics: Science and Systems*. Pittsburgh, Pennsylvania volume 14. doi:10.15607/RSS.2018.XIV.019.

Zeiler, M. D., Krishnan, D., Taylor, G. W., & Fergus, R. (2010). Deconvolutional networks. In *2010 IEEE Computer Society Conference on Computer Vision and*

Pattern Recognition (pp. 2528–2535). doi:10.1109/CVPR.2010.5539957.

Zhou, Y., Barnes, C., Lu, J., Yang, J., & Li, H. (2019). On the continuity of rotation representations in neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 5738–5746). doi:10.1109/CVPR.2019.00589.

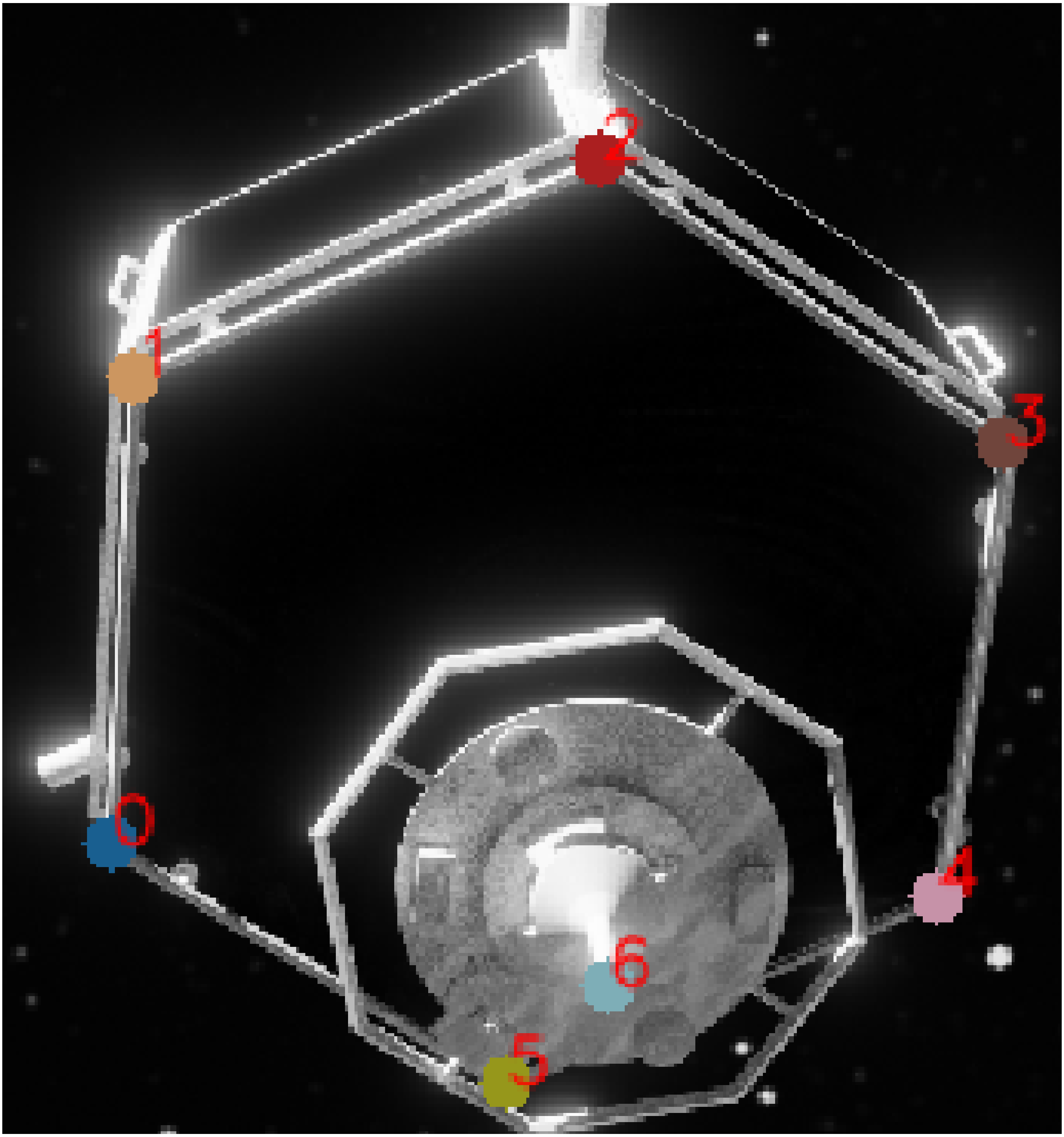
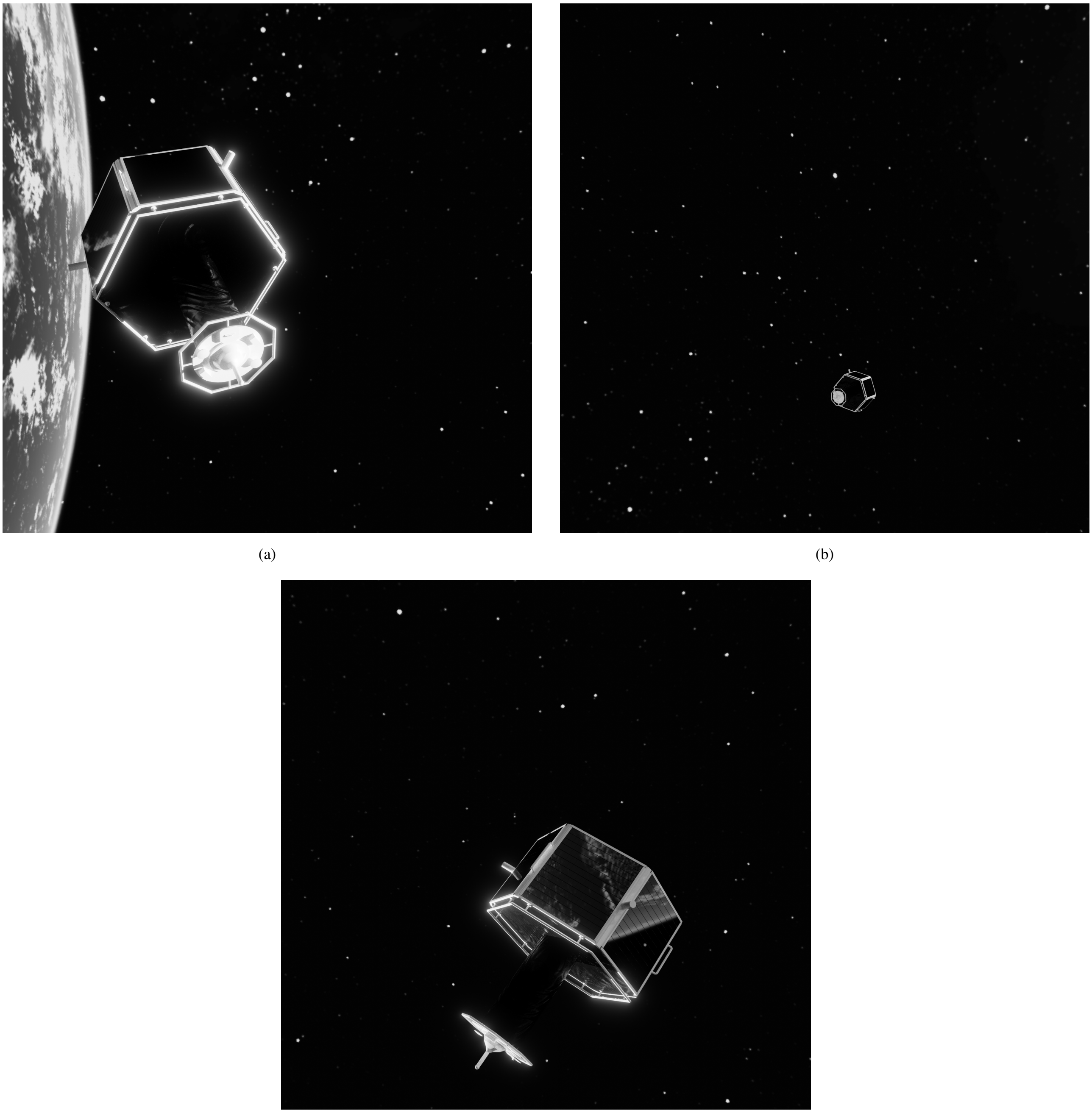


Fig. 1: Estimated keypoints on the spacecraft body as marked with coloured circles.

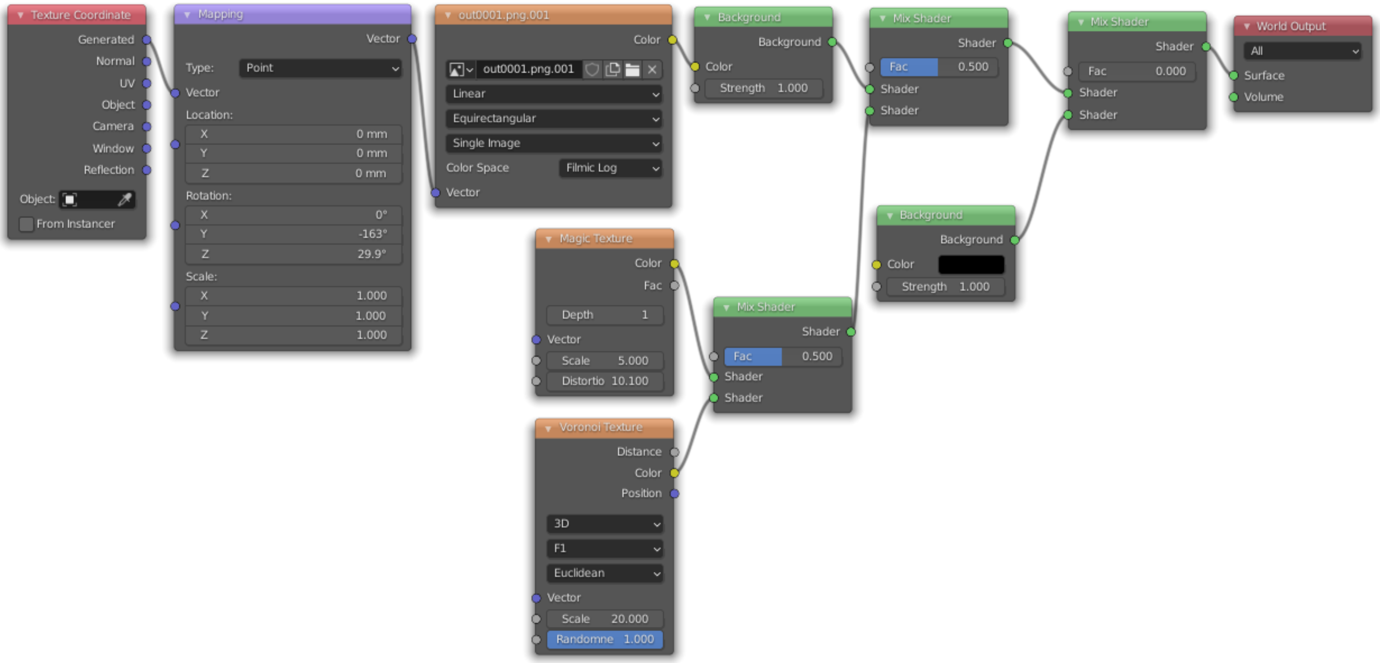


(a)

(b)

(c)

Fig. 2: Several views of the representative target spacecraft as rendered in Blender.



Texture Coordinate: Object: □ From Instancer

Mapping: Type: Point
Location: X 0 mm, Y 0 mm, Z 0 mm
Rotation: X 0°, Y -163°, Z 29.9°
Scale: X 1.000, Y 1.000, Z 1.000

out0001.png.001: out0001.png.001; Linear; Equirectangular; Single Image; Color Space: Filmic Log

Background: Strength 1.000

Mix Shader: Fac 0.500

Mix Shader: Fac 0.000

World Output: All

Background: Color ■; Strength 1.000

Magic Texture: Depth 1; Scale 5.000; Distortio 10.100

Mix Shader: Fac 0.500

Voronoi Texture: 3D; F1; Euclidean; Scale 20.000; Randomne 1.000

Fig. 3: Node setup for background environment in Blender

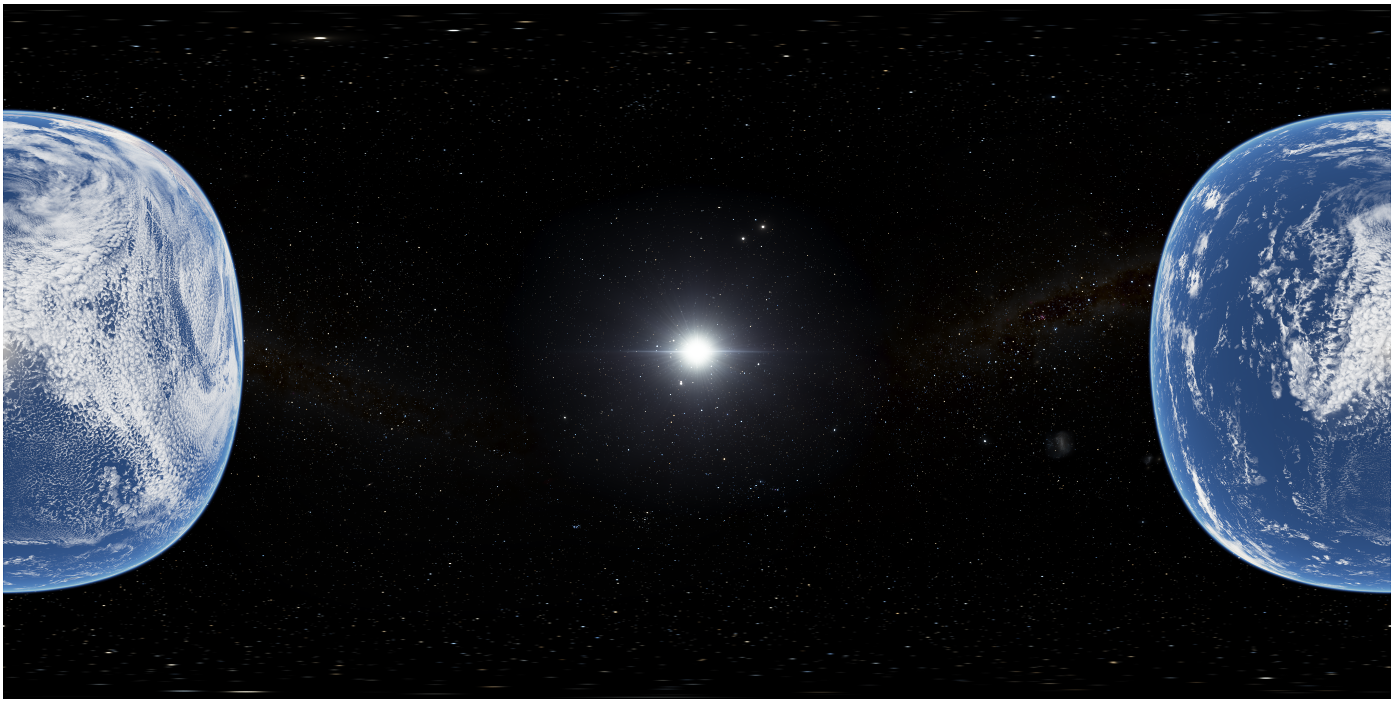


Fig. 4: Equirectangular projection of the surrounding environment texture, generated via SpaceEngine

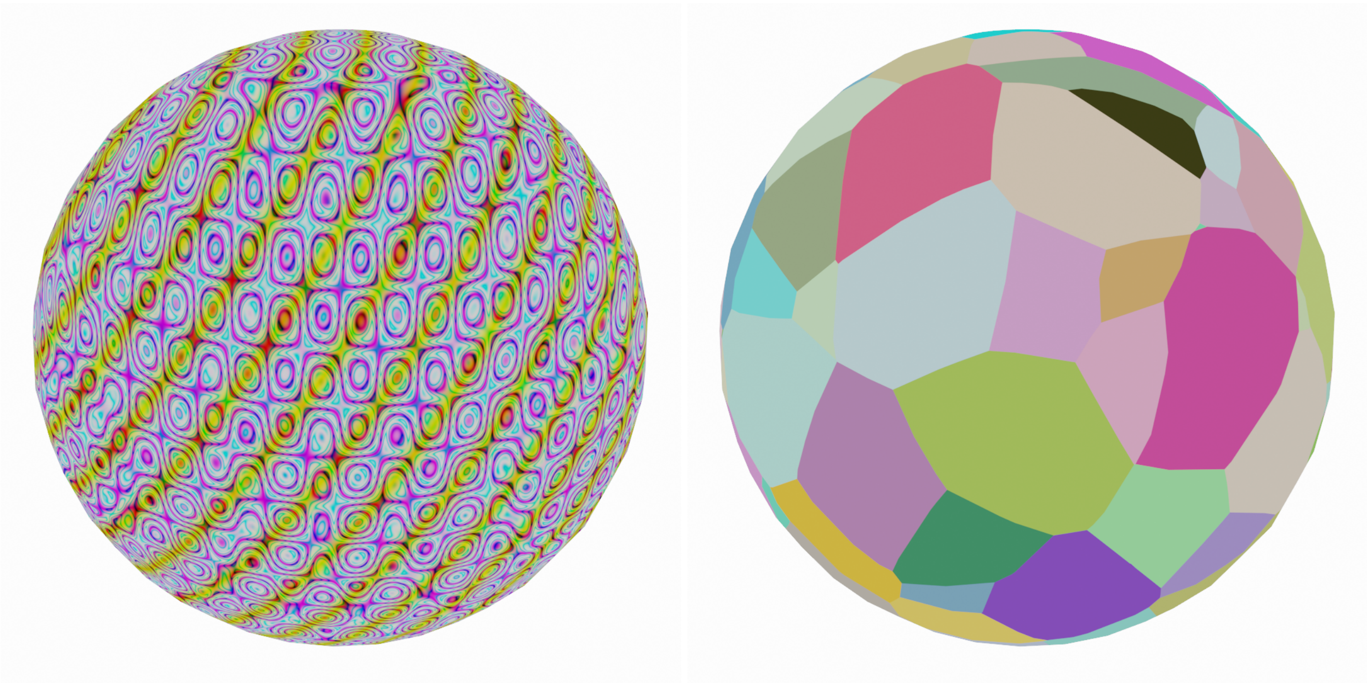


Fig. 5: Example outputs of Magic Texture Node (left) and Voronoi Texture Node(right)

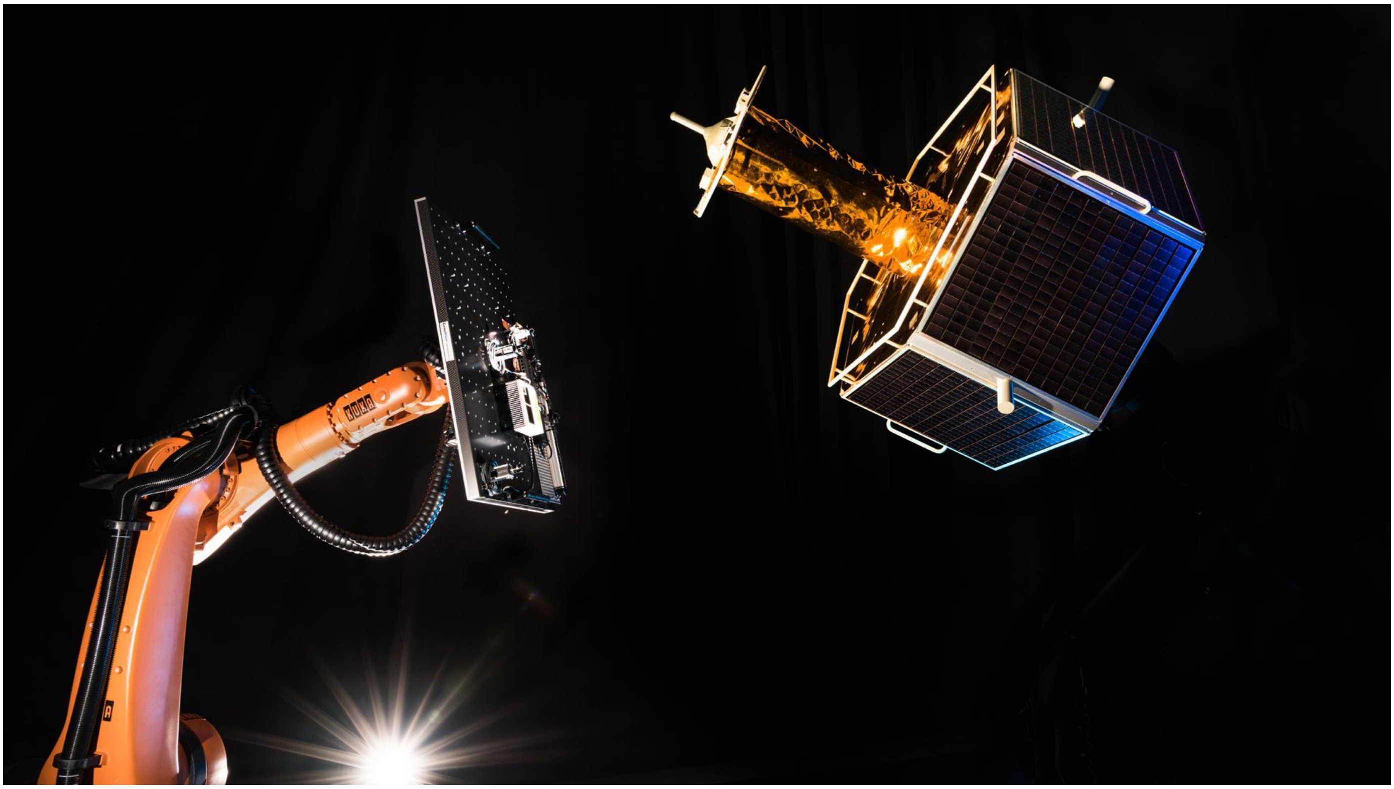
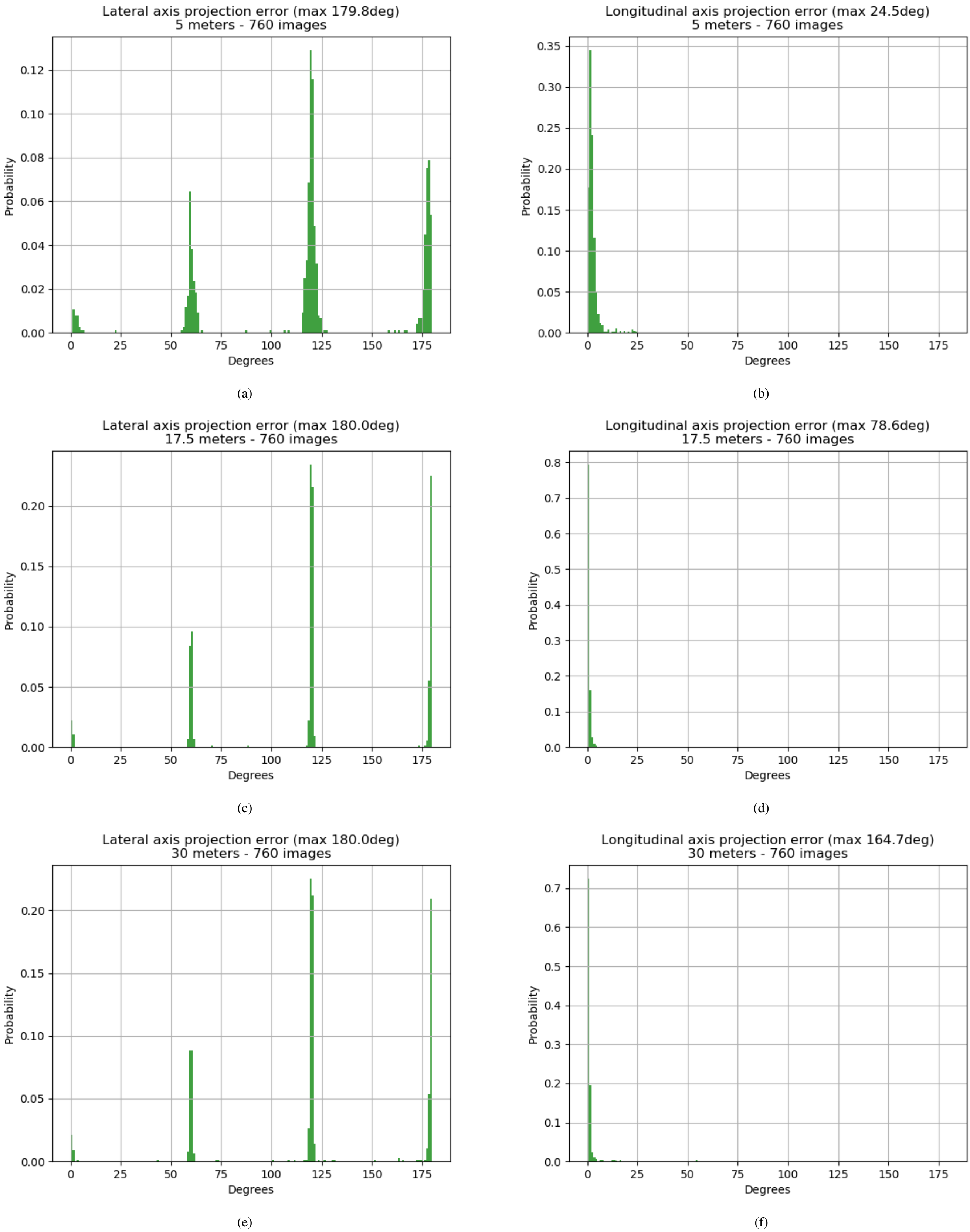


Fig. 6: A view of the robotic arm system that simulates proximity manoeuvres via movement of a sensor package and a physical spacecraft mock-up at the EPOS 2.0 laboratory. Photo: DLR, CC-BY 3.0



(a)

(b)

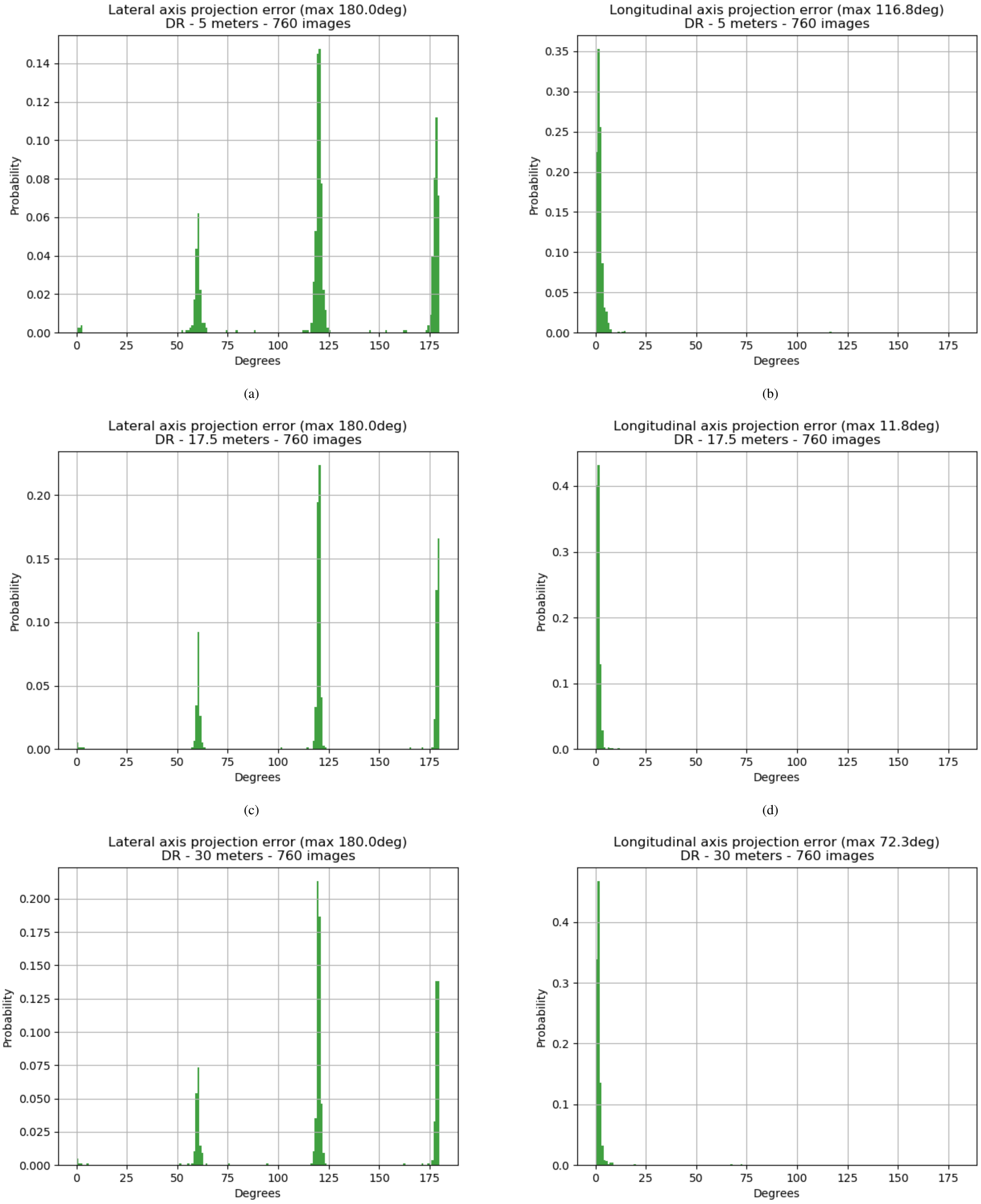
(c)

(d)

(e)

(f)

Fig. 7: Comprehensive results of the first experiment. Figures a, c, and e correspond to lateral axis projection error histograms at ranges 5, 17.5, and 30 m, respectively. Figures b, d, and f correspond to longitudinal axis projection error histograms at ranges 5, 17.5, and 30 m, respectively.



(a)

(b)

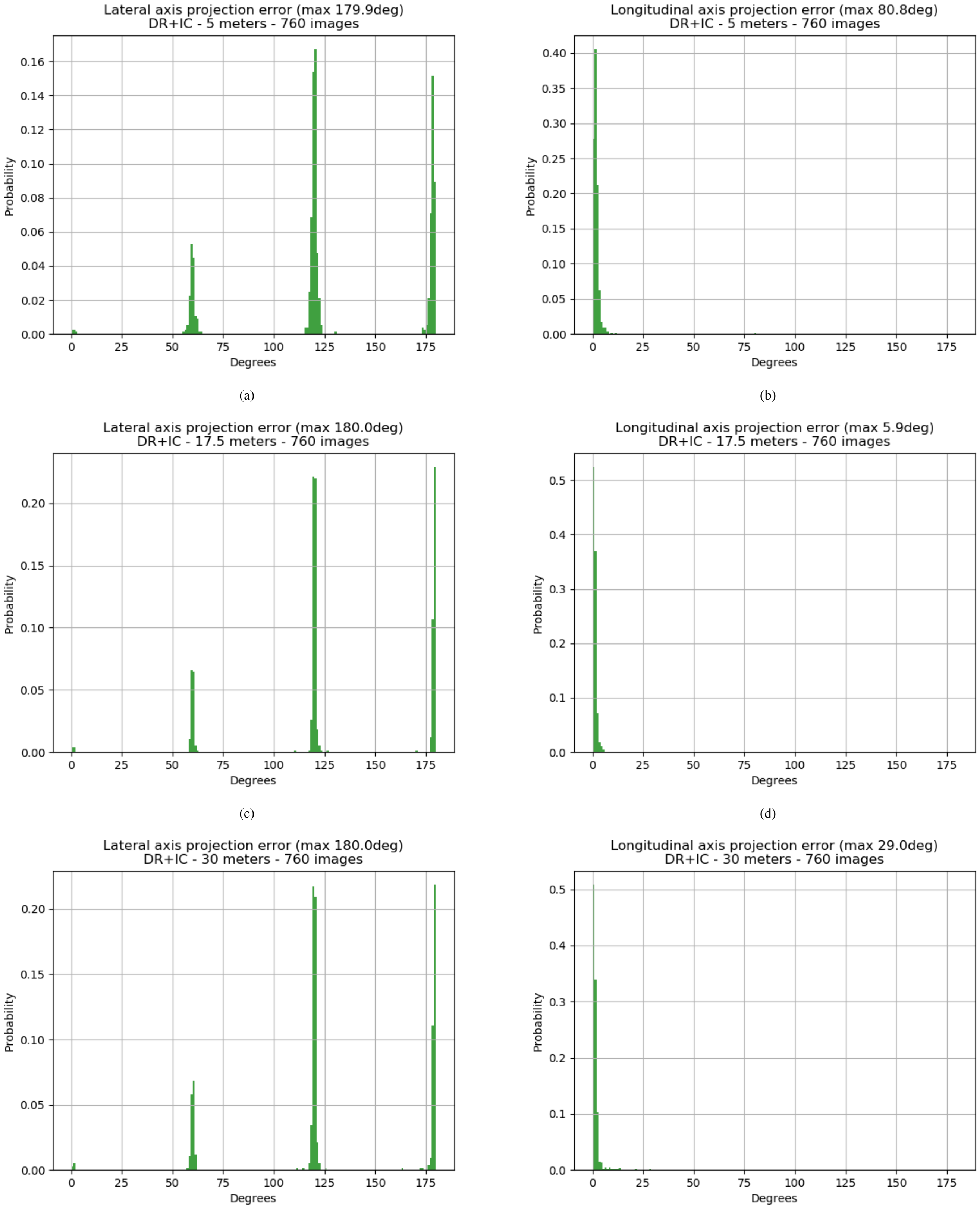
(c)

(d)

(e)

(f)

Fig. 8: Comprehensive results of the second experiment. Figures a, c, and e correspond to lateral axis projection error histograms at ranges 5, 17.5, and 30 m, respectively. Figures b, d, and f correspond to longitudinal axis projection error histograms at ranges 5, 17.5, and 30 m, respectively.



(a)

(b)

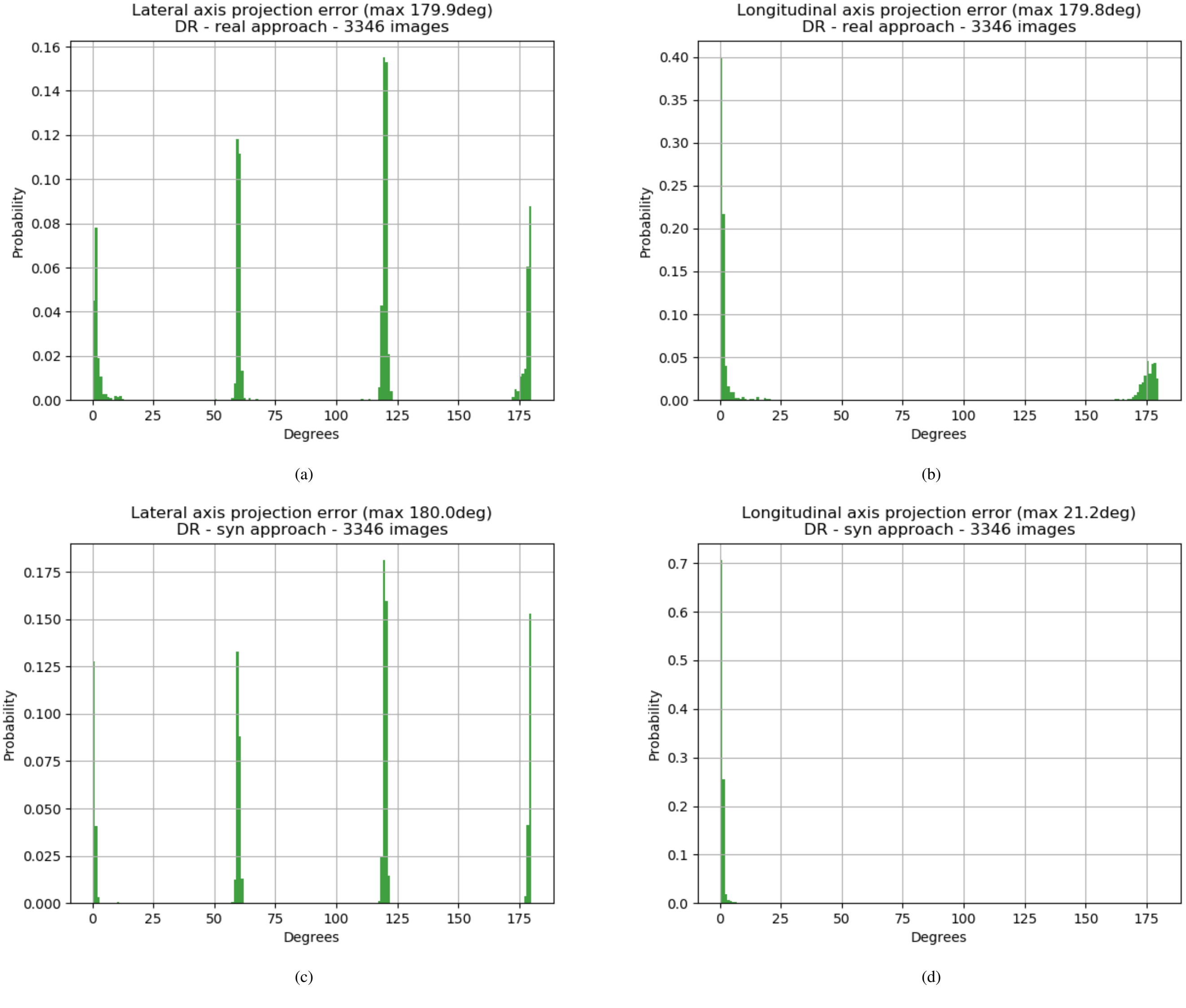
(c)

(d)

(e)

(f)

Fig. 9: Comprehensive results of the third experiment. Figures a, c, and e correspond to lateral axis projection error histograms at ranges 5, 17.5, and 30 m, respectively. Figures b, d, and f correspond to longitudinal axis projection error histograms at ranges 5, 17.5, and 30 m, respectively.



(a)

(b)

(c)

(d)

Fig. 10: Comprehensive results of the fourth experiment. Figure a and c present lateral axis projection errors for real and synthetic data for a representative approach maneuver, respectively. Figures b and c present longitudinal axis projection errors for real and synthetic data, respectively.

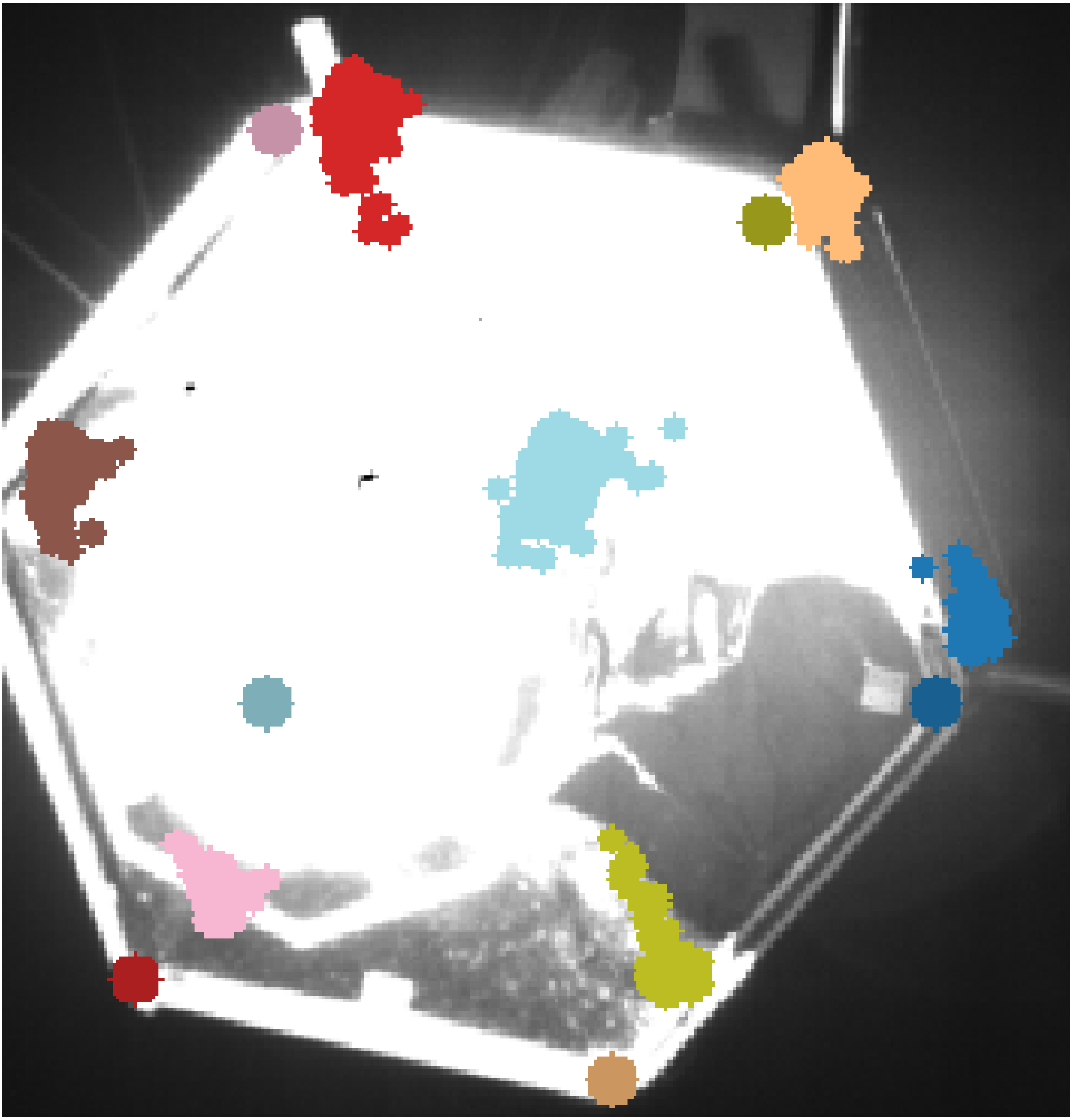


Fig. 11: A failure mode of the CNN, where the CNN predicts keypoints for the wrong pose due to intensely adverse lighting conditions. The large colored circles correspond to the grount truth keypoints, and the smaller clusters of points correspond to the keypoints predicted by all the cells on the output side of the CNN.

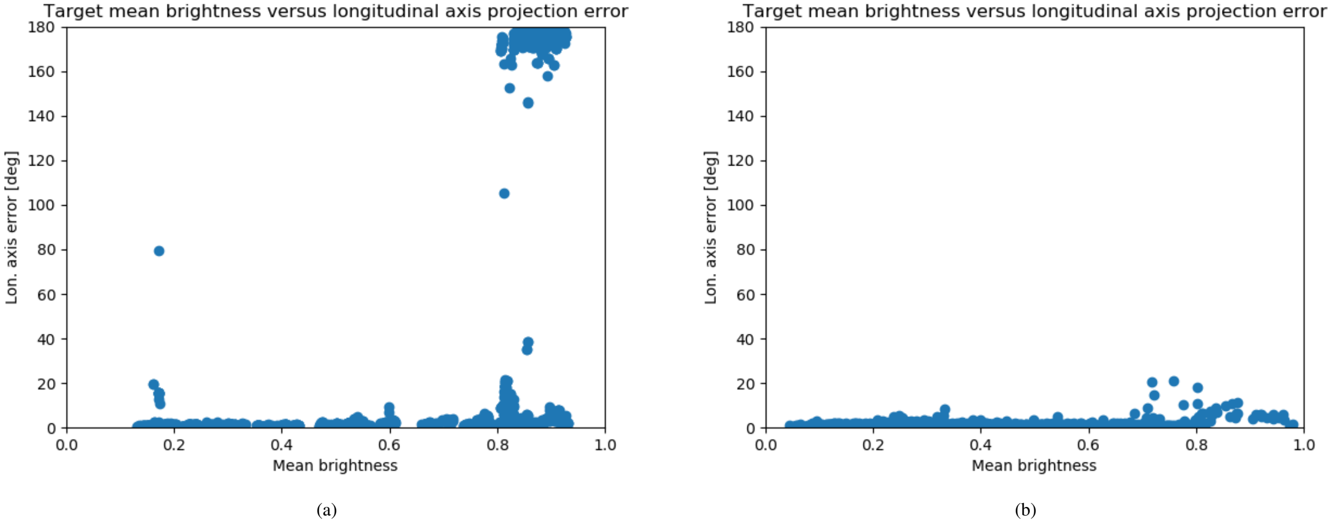


Fig. 12: Comparison of longitudinal error versus mean brightness of the spacecraft for the synthetic and real image dataset. Figure a corresponds to the real camera image dataset and Figure b corresponds to the synthetic image dataset. Pixel brightness ranges have been normalized to range from 0 to 1.

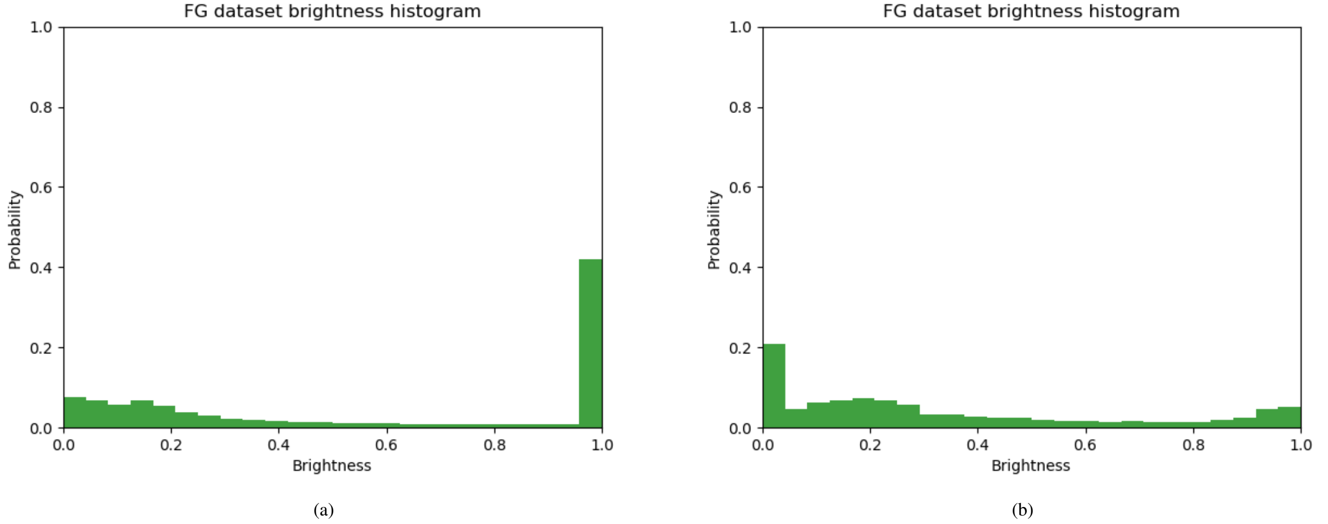


Fig. 13: Comparison of spacecraft pixel intensity histograms across the whole dataset for the synthetic and real image dataset. Figure a corresponds to the real camera image dataset and Figure b corresponds to the synthetic image dataset. Pixel brightness ranges have been normalized to range from 0 to 1.

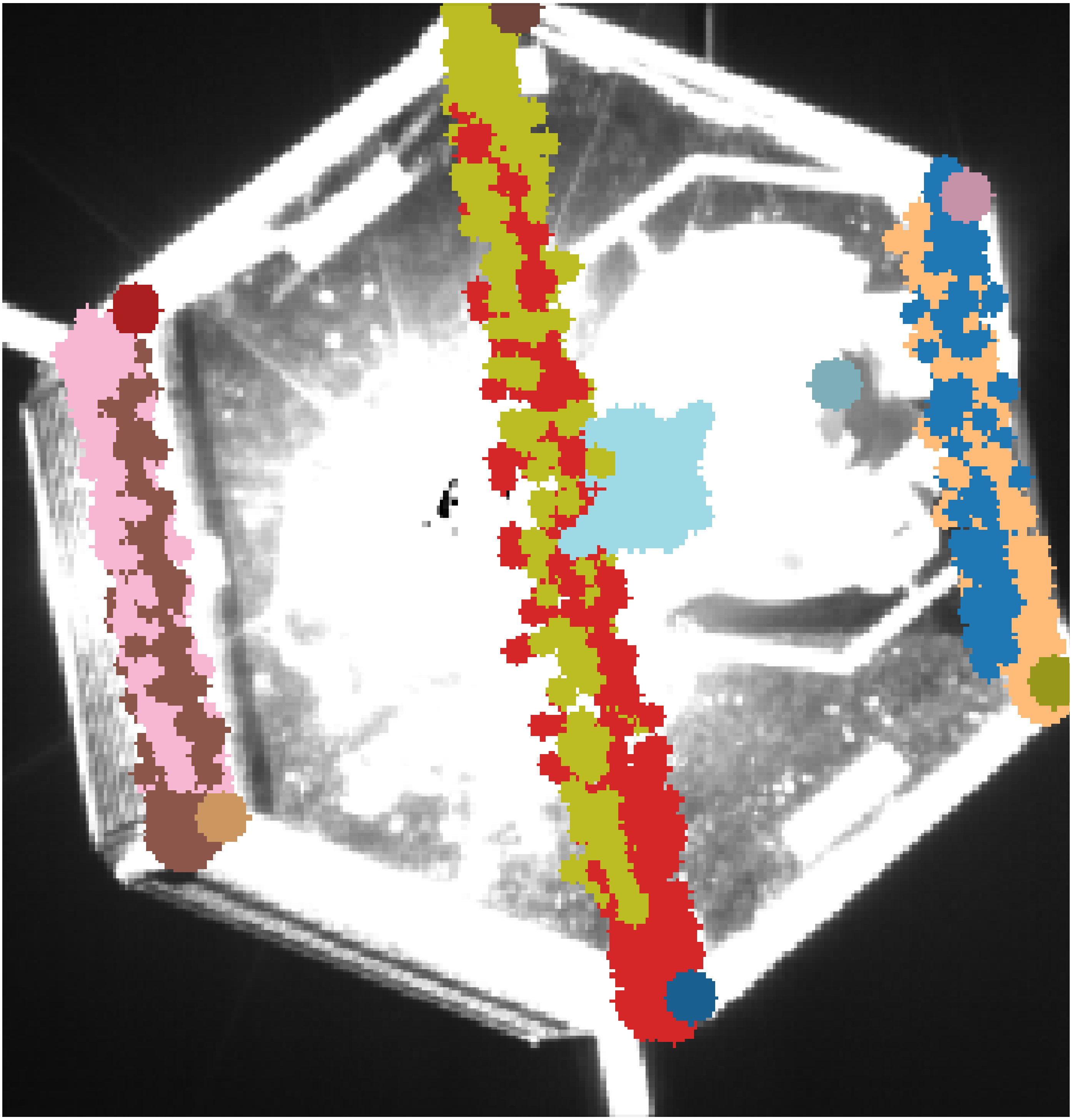


Fig. 14: A failure mode of the CNN, where it is unable to decide which of the ambiguous poses to estimate, spreading the keypoints between the two possibilities. The large colored circles correspond to the grount truth keypoints, and the smaller clusters of points correspond to the keypoints predicted by all the cells on the output side of the CNN.