CNN-Based Visual Navigation: Optimization Strategies for Monocular Pose Estimation in Proximity Operations

IAC–24–C.1.88170

CNN-based Visual Navigation: Optimization Strategies for Monocular Pose Estimation in Proximity Operations

Lucrezia Lovaglio[a*], Antonio D' Ortona[b], Fabrizio Stesina[c], Sabrina Corpino[d]

[a] *Mechanical and Aerospace Engineering Department, Politecnico di Torino, 10129 Turin, Italy, lucrezia.lovaglio@polito.it*
[b] *Mechanical and Aerospace Engineering Department, Politecnico di Torino, 10129 Turin, Italy, antonio.dortona@polito.it*
[c] *Mechanical and Aerospace Engineering Department, Politecnico di Torino, 10129 Turin, Italy, fabrizio.stesina@polito.it*
[d] *Mechanical and Aerospace Engineering Department, Politecnico di Torino, 10129 Turin, Italy, sabrina.corpino@polito.it*
[*] *Corresponding author*

Abstract

Proximity operations are becoming increasingly more important for current and future space missions, particularly On-Orbit-Servicing (OOS) and Active-Debris Removal (ADR) ones. In this framework, a high-accuracy estimation of the relative pose (position and attitude) between spacecraft is required to successfully and safely achieve complex proximity operations like inspection, rendezvous, and docking. Visual navigation has recently become one of the most popular techniques for this purpose, thanks to the availability of increasingly compact, precise, and reliable monocular cameras. Traditional approaches relying on hand-engineered feature matching do not guarantee robustness or sufficient generalization, whereas Convolutional Neural Network (CNN)-based architectures have demonstrated improved robustness, noise rejection, and resilience to unseen scenarios. Despite their potential, these algorithms do not frequently reach the desired accuracy due, among others, to the employment of heuristic approaches in the choice of hyperparameters and the unavailability of an adequate large dataset.

This work proposes a CNN-based architecture for non-cooperative spacecraft monocular pose estimation exploiting optimization techniques to overcome these limits, improve performances and reduce the computational effort. This is achieved through the usage of a robust analytical method to select the best set of hyperparameters to minimize the pose loss function and the enhancement of the dataset for better feature learning. Moreover, the relationship between hyperparameters and the objective function (pose loss) is investigated, as well as the impact of different sets of hyperparameters on the CNN performance. A Blender® based synthetic dataset of approximately 25,000 synthetic images of an uncooperative target is generated to train the CNN. Such images are used to emulate representative proximity scenarios to validate the proposed approach.

The obtained results show that the proposed algorithm achieves centimeter-level position accuracy and near-degree-level attitude accuracy, maintaining, at the same time, high robustness against changes of illumination conditions and background textures.

Keywords: On-Orbit Servicing (OOS), rendezvous and docking, pose estimation, visual navigation, Convolutional Neural Network (CNN), hyperparameter optimization

Acronyms/Abbreviations

Active Debris Removal (ADR)
Convolutional Neural Network (CNN)
Pose EStimation Network (PEN)
Perspective-n-Point (PnP)
Spacecraft Pose EStimation Dataset (SPEED)
Waking Safety Ellipse (WSE)
Final Approach (FA)
Testebed for Rendezvous and Optical Navigation (TRON)
Space Rendezvous Laboratory (SLAB)
In-Orbit Demonstration missions (IOD)
On-orbit Servicing missions (OOS)
Hyperparamter Optimization (HPO)
Tree-structured Parzen Estimator (TPE)

1. Introduction

The last years observed a dramatic increment in the small satellites market due to the improvement of the operational capabilities achieved by this type of spacecraft while maintaining lower cost and quicker schedule compared to larger spacecraft. Small-satellite missions are now used for a range of applications, including Space exploration [1], On-Orbit Servicing (OOS) [2], In-Orbit Demonstration (IOD) [3], and remote sensing [4]. One of the most interesting CubeSat missions involves inspecting a mothership by maneuvering around it [5]. In the framework of these missions, the most challenging operational capability [6] to be completed is often the retrieval. The satellite should approach the Target vehicle and achieve

the retrieval position. Proximity operations, including rendezvous and retrieval, impose stringent constraints that significantly influence the design of critical subsystems, such as attitude determination and control [7], propulsion systems, retrieval mechanisms [8], and guidance, navigation, and control [9] [10] (GNC). To address these challenges, highly precise relative navigation techniques are required to complete the mission for their accuracy. Nowadays, digital cameras are compact, accurate, noninvasive, and inexpensive making visual navigation the optimal approach for precisely estimating relative position and attitude - pose estimation - .

Traditional pose estimation methods are based on hand-engineered features (e.g. corners, blobs, edges) described using feature descriptors and detected using feature detectors (e.g. SIFT, SURF,etc) to detect features in the 2D images and to use their 3D correspondences to compute the relative pose [11], [12]. Unfortunately, these methods struggled in a space environment under varying illumination conditions, low signal-to-noise ratios, and the high contrast often found in space imagery, leading to inaccurate target state estimation in many situations. These factors have driven the development towards Convolutional Neural Networks, with impressive results, such as higher robustness and resilience to noise and unseen scenarios. A CNN for space application was proposed in [13] that extracts 2D keypoints on the image which can be used in conjunction with corresponding 3D model coordinates (landmarks) to compute relative pose via the Perspective-n-Point (PnP) problem showing promising results. Another example can be found in [14] where the landmarks detection is combined with geometric optimisation, associating the keypoints to their corresponding 3D points on an a priori reconstructed 3D model, then solving for the object pose using non-linear optimisation.

However, the majority of Deep Learning-based approaches still depend heavily on labeled data, which is difficult to obtain. While generating synthetic data and acquiring data in laboratory settings are considered the most feasible methods for training and testing these algorithms, there is a significant performance drop when applied to real-world test images compared to training images. This issue is known as the Domain Gap.

1.1 Domain Gap

One of the primary issues with applying ML models to images is that the training process requires a huge quantity of specific-application data. In the context of Space Exploration, this gap refers to the significant difference between training data and the data encountered during actual operations in Space and a consequent obstacle in obtaining an exhaustive evaluation of the trained model *before* the deployment of the mission itself.

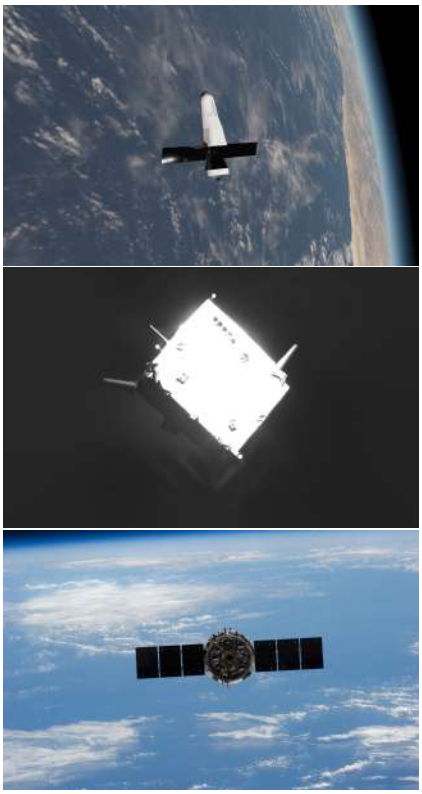
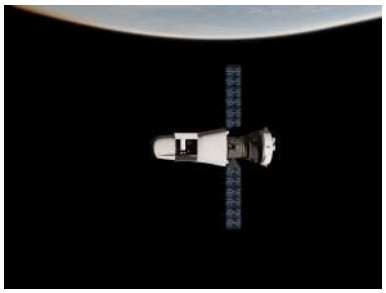


Fig. 1. Starting from the upper image, a synthetic image of a target satellite, in the middle a HIL image (SPEED+, *sunlamp*), and on the bottom a real image of a target satellite(mission *Cygnus*, NASA, February 2014).

Addressing Domain Gap is crucial for ensuring the success of space missions, as systems designed and tested in a too different environment may not perform optimally or may even fail when deployed in Space. Several strategies have been developed to bridge the Domain Gap. Among the most popular is the development of a benchmark dataset, SPEED, which later became SPEED+ [15]. It is composed by 59.960 *synthetic* photos, 6,740 HIL images for emulating diffuse light in Earth's orbits (*lightbox*), and 2,791 HIL images for simulating the sun's influence on the spacecraft, (*sunlamp*).[16] These images are produced at the Testebed for Rendezvous and Optical Navigation (TRON) facility at Space Rendezvous Laboratory (SLAB) at Stanford University by using two 6 Degrees-of-Freedom (DOF) robot arms, one holding a mockup of the spacecraft, and the other moving a camera along a linear rail. These images can be considered as surrogates of spaceborne images but they do not cover all possible scenarios, although making a great contribution to solving the

problem. Although the images in these datasets are invaluable for addressing the gap, they cannot account for every possible scenario encountered in space. To further minimize the it, real spaceborne images from missions, such as NASA's *Cygnus* mission [17], are used. However, due to differences in target geometry and environmental factors, results obtained from these real images may not always be consistent. Additional techniques, such as data augmentation and texture randomization, are frequently employed to make datasets as domain-agnostic as possible, further enhancing the robustness of the models.

The proposed work seeks to enhance the accuracy and robustness of pose estimation through a CNN-based approach combined with advanced optimization techniques. Specifically, the study investigates hyperparameter optimization as a method to identify the optimal set of hyperparameters, thereby minimizing the pose loss function. Additionally, the dataset is refined and augmented to improve feature learning.
The structure of the paper is organized as follows: Section 2 addresses the challenge of the Domain Gap and outlines the creation of the dataset, along with the enhancement techniques employed, such as data augmentation. Section 3 provides a numerical evaluation of the proposed method by comparing the performance of the trained model with and without the application of the suggested optimization techniques. This section also presents simulation results and discusses key performance metrics that demonstrate the improvement in accuracy and robustness. Finally, Section 4 concludes the paper by summarizing the key contributions and findings of the study, as well as proposing potential directions for future research and implementation.



(a) Final Approach dataset

(b) Walking Safety Ellipse dataset

Fig. 2. Examples of images from the two datasets

Table 1. Dataset settings

Dataset	
Walking Safety Ellipse	**Final approach**
20000 RBG images	5000 RBG images
600 to 100 m	80 to 8 m
2048x1536 pixels	2048x1536 pixels
70-15-15 ratio	70-15-15 ratio

2. Dataset

The dataset is composed of around 25000 synthetic RGB images divided into two main blocks: 20000 are generated in the context of Walking Safety Ellipses (WSE) for target observation, spanning from 600 to 100 m away from the target distances, and 5000 RGB images are produced in the context of the Final Approach (FA) trajectory during the rendezvous maneuver, spanning from 80 to 8 m. In both cases, the images are generated placing the chaser on the corresponding trajectory, considering random illumination conditions with a resolution of 2048x1536 pixels.

Also, they are divided using a 70-15-15 ratio for training, validation, and testing respectively, to avoid overfitting.

2.1 Dataset Generation

The custom dataset is developed using Python APIs of Blender. For this specific application, a CAD model of the target is placed into a scenario and Blender's, built-in camera model is used to visualize the camera's field of view. While this method may lack physical and radiometric accuracy, it serves as an effective means to assess feature extraction algorithms in ideal conditions. The environment is built by incorporating a realistically scaled Earth, a dynamic sun, and trajectory input from STK or MATLAB simulations. The Earth comprises three distinct elements - terrain, atmosphere, and clouds - each with a customized shader. *"The atmosphere is modeled to replicate Rayleigh scattering and atmospheric pressure decay, attributing color to the sky during the day and imparting a reddish hue during sunset. Volu-*

metric effects in the cloud layer are simulated through a semi-transparent texture enveloping the globe, while the placement of the sun mesh aligns with the sun vector, accompanied by Blender's lens flare effects when visible from the camera[18]".
Blender's API also generates *ground truth* labels based on the known position and orientation of the target. Despite its benefits, this approach has some limitations. The synthetic nature of the images leads to inaccuracies, and the rendering process is time-consuming. Additionally, issues like total black images or those generated during eclipses complicate feature recognition. These challenges make dataset cleaning necessary, for which a CNN is used to filter out problematic images and improve dataset quality.

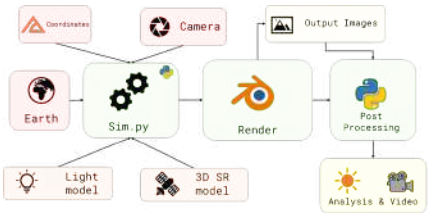


Fig. 3. Dataset creation pipeline from [18]

Fig. 4. On the right side, a random image generated in normal conditions, in the center a random image generated in eclipse conditions, and on the left side, a random image generated without the presence of the target.

2.2 Data Augmentation

In addition to utilizing the developed synthetic dataset of 25,000 images to train the network for pose estimation, Data Augmentation techniques have also been applied to further enhance model accuracy. For these reasons, Data Augmentation is fundamental to broaden the dataset, enhance the performances and reduce the overfitting on synthetic images. The used approach in this work is to use Albumentations [19], a python library widely used in computer vision tasks to increase the quality of trained samples from already existing data.
The applied effects can be divided into style-augmentation, like *Blur* or *Texture Randomization* and more standard ones, like *Flip* and *Resize*.

Fig. 5. Example of input image

Table 2. List of the Data Augmentation and equivalent commands in Albumentations. Table inspired by [16].

Data Augmentation	
Augmentation Techniques	**Commands**
Resize	*A.Resize*
Horizontal Flip	*A.HorizontalFlip*
Vertical flip	*A.VerticalFlip*
Noise	*A.OneOf(A.GaussNoise,A.ISONoise)*
Blur	*A.OneOf(A.MotionBlur, A.MedianBlur,A.GlassBlur)*
Brightness and Contrast	*A.RandomBrightnessContrast*
Texture Randomization*	

*`github.com/philipjackson/style-augmentation`[20]

The implemented augmentations, showed in Table 2 have a probability of 0.5 for activation each.

(a) Vertical flip (b) Horizontal flip (c) Blur

Fig. 6. Example of applied transforms

endcomment

3. Pose Estimation Network

This work proposes a CNN-based architecture for non-cooperative spacecraft pose estimation using a multiple-heads architecture.

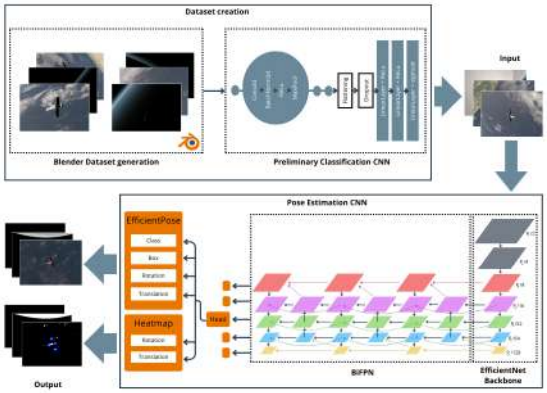


Fig. 7. Pose Estimation Network architecture

It consists of a three-step process:

- The number of keypoints on the item to be detected is selected. In this application, a set of 6 keypoints is picked, geometrically corresponding to spacecraft's **head**, the **tail** and the **corners** of its two solar panels. Subsequently, the 3D coordinates are retrieved from the CAD model of the target and the ground-truth (2D pixel coordinates of the keypoints) is computed through an inverse PnP problem [18]

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & u0 \\ 0 & f_y & v0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

  where u and v are obtained from the multiplication of the intrinsic camera matrix K with the extrinsic camera matrix [R|t] (it is the one that is normally unknown and has to be estimated) and the 3D coordinates of the keypoints in the target Reference Frame.
  The intrinsic camera matrix's parameters,(f_x and f_y), describe the camera's focal lengths, whereas γ represents the skew coefficient. $u0$ and $v0$ are the principal point coordinates.
  In the extrinsic camera matrix, r_{ij} represents the rotation matrix elements, whereas t_x, t_y and t_z are translation vector components of the camera.

- Once the dataset has been generated, a first CNN is used is to clean it from black images and the ones generated in eclipse, in which no features can be extracted. The built network is a binary classifier, trained to recognize the presence of the target in the image.

- The images are then fed into a second CNN, which architecture is inspired by [16], consisting of multiple prediction heads that are used for different purposes, such as **bounding box detection**, **keypoints prediction** and **direct pose regression**. These heads are connected between each other thanks to the Shared Feature Encoder composed by the EfficientNet backbone and the Bi-directionalFeature Pyramid Network. The presence of multiple heads has the aim to collect more generalized features that can add robustness to the prediction.

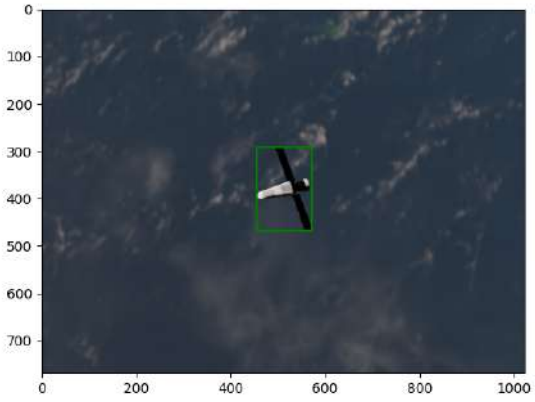


Fig. 8. EfficientPose bounding box prediction

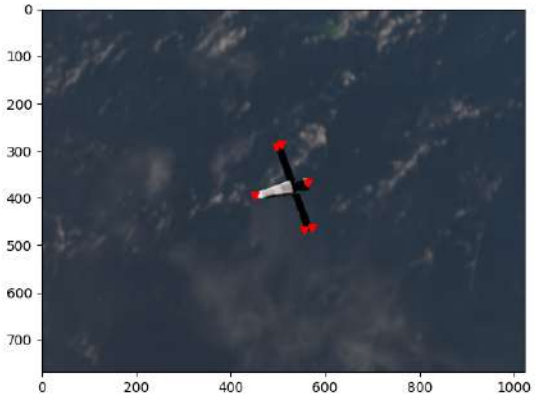


Fig. 9. EfficientPose keypoints prediction

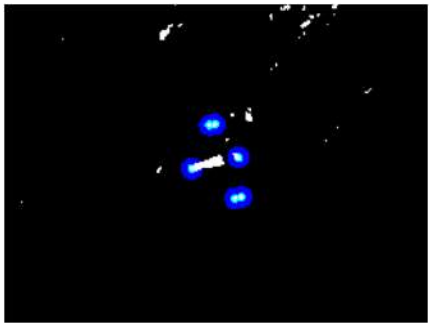


Fig. 10. Heatmap keypoints prediction (overlay of single keypoint heatmaps)

4. Hyperparameters Optimization

Hyperparameters play a crucial role in determining the performance of CNN-based algorithms. However, the process of identifying the best hyperparameters can be challenging and typically involves subjective and ineffective trial-and-error methods, as they are mostly tuned using a heuristic approach. While these methods are simple and easy to implement, they rely on subjective and sometimes inefficient trial-and-error procedures that frequently fail to fully exploit the potential of CNN architectures, resulting in suboptimal performance.

Hyperparameter optimization (HPO) provides a systematic and automated strategy to overcome these restrictions, improving the performance of CNN-based algorithms. By leveraging optimization techniques such as grid search, random search, or Bayesian optimization, HPO systematically explores the hyperparameter space to identify configurations that maximize performance metrics. This systematic exploration mitigates the risk of missing crucial hyperparameter interactions and assures a more exhaustive search for optimal configurations. Furthermore, HPO techniques allow for the introduction of domain knowledge and prior information into the optimization process, allowing for more informed decisions about hyperparameter settings. E.g. Bayesian optimization can efficiently leverage past evaluations to adaptively adjust the search space and prioritize promising regions, leading to faster convergence and improved performance.

This work specifically explores hyperparameter optimization as a method to improve the accuracy and robustness of the proposed CNN-based architecture. The goal is to identify the optimal set of hyperparameters for training the network, which in turn minimizes the pose loss function and enhances overall model performance. **Optuna** library [21] is used as hyperparameter optimization framework.

Fig. 11. `https://github.com/optuna/optuna/tree/master`

It is a popular choice since it is easy to use and has a wide range of features for efficient experimentation. Its lightweight, versatile, and platform-agnostic architecture ensures compatibility across various tasks with minimal setup requirements. Moreover, Optuna adopts state-of-the-art algorithms for sampling hyperparameters and efficiently pruning unpromising trials, optimizing the exploration process. Additionally, rapid visualization tools allow users to easily review optimization histories, offering significant insights into the optimization process.

In the presented HPO, a *study* instance is created. The study aims to identify the ideal set of hyperparameter values (e.g. optimizer, scheduler, etc.) that minimize or maximize the stated objective function across several trials (e.g., 30). In this specific case, the objective function is the pose loss, computed at each validation step, while the hyperparameters search space comprises:

- *Scheduler*:options include *exponential*,*polynomial*, *cosine decay*, *step*, and *cosine warmup*
- *Learning rate*:the available range spans from 1e-5 to 1e-3. The proposed limits represents standard values, known to be effective for this parameter in similar tasks
- *Learning rate factor*: also in this case, standard values are considered, defining a range from 0.1 to 0.9
- *Batch size*: parameter ranges from a minimum of 4 to a maximum of 8 images per GPU. This range accommodates standard values while considering hardware limitations and optimization for parallel processing
- *Number of epochs*:it is set from 100 to 150 epochs. The objective is to identify an optimal balance between training duration and model performance, considering computational constraints
- *Optimizer*: options include *Adam*, *AdamW*, and *SGD*. While Adam is commonly used, exploring alternative optimizers accounts for po-tential interactions between hyperparameters and optimizer choice.
- *Weight decay interval amplitude*: this parameter defines the amplitude of the weight decay interval in case of *step* scheduler. It can be chosen between 1 and 2

- *Epochs for learning rate decay*: this parameter defines the number of epochs after which the learning rate decays. It can be chosen between 0 and the end epoch

Hyperparameter optimization	
Hyperparameter	**Values**
Learning rate	[1e-5;1e-3]
Learning rate factor	[0.1;0.9]
Batch size	[4;8]
Number of epochs	[100;150]
Optimizer	*Adam, SGD, AdamW*
Weight decay interval amplitude	[1;2]
Epochs for learning rate decay	[0;End epoch]

Table 3. Summary of chosen hyperparameters to be tuned and relative search spaces

Along with the definition of the parameters, the pruning method and sampler are also defined to help efficiently allocate computational resources and terminating trials that are unlikely to yield promising results.

The chosen pruning method is the **MedianPruner**. It prunes trials that are less promising based on the median of the intermediate values observed. In other words, a trial is pruned if its intermediate value is worse than the median of the intermediate values of the previous trials at the same step.

The **Tree-structured Parzen Estimator (TPE)** optimizer is adopted as sampler. It balances exploration (searching for promising regions of the search space) and exploitation (focusing on regions that are likely to contain the optimum) by using a tree-structured approach:

1. *Division of Search Space*: the Search Space is divided into regions or segments, often represented in a hierarchical or tree-like structure, each node in the tree corresponding to a particular region.

2. *Exploration and Exploitation*: TPE iteratively explores different regions, guided by the tree structure, to find promising areas. It prioritizes regions that have been less explored to balance exploration and exploitation.

3. *Probabilistic Modeling*: TPE uses a probabilistic model within each region to estimate the performance of hyperparameter configurations, focusing the search on those with higher predicted performance.

4. *Tree Structure Update*: As the search progresses, TPE updates the tree dynamically based on performance data, improving resource allocation by concentrating on regions with better potential for performance gains.

Once the hyperparameters, pruner, and sampler are defined, the model is built, a guess is made, and the objective function is initialized. Before optimization begins, random seeds are set, and configuration parameters are updated based on trial suggestions to ensure reproducibility. The training process then enters its main loop, where it iterates over epochs, adjusts the learning rate, and evaluates model performance on the validation set. If the pose estimation loss does not meet the pruning criteria, the trial is pruned to save time and resources.

5. Results and Discussion

5.1 Trial-and-error VS Hyperparameter optimization

Two training sessions are conducted with different sets of hyperparameters. The first session uses trial-and-error values, while the second session employs values obtained from a hyperparameter optimization process. Although the second session runs for 100 epochs, only the first 30 epochs are reported for comparison with the duration of the first session.

Pose Estimation Network - Hyperparameter settings	
Hyperparameters	**Values**
Learning rate	0.000533
Learning rate steps	[44,65]
Learning rate factor	0.2775
Batch size	7
Number of epochs	100
Number of validation epochs	*One after each training epoch*

Table 4. Summary of PEN hyperparameters and their values for the hyperparameter optimization approach

Pose Estimation Network - Hyperparameter settings	
Hyperparameters	**Values**
Learning rate	0.01
Learning rate steps	[15,18]
Learning rate factor	0.1
Batch size	8
Number of epochs	30
Number of validation epochs	*One after each training epoch*

Table 5. Summary of PEN hyperparameters an their values for the trial and error approach

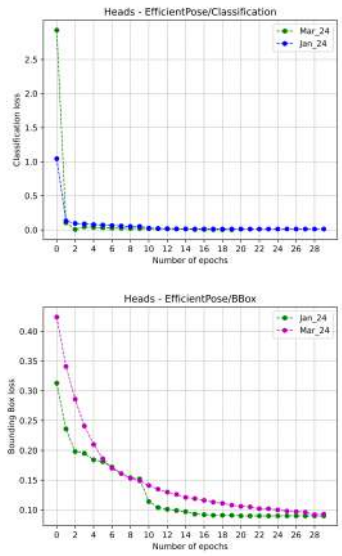


Fig. 12. Classification and Bounding Box loss comparison between trial and error approach (Jan24) and hyperparameter optimization (Mar24)

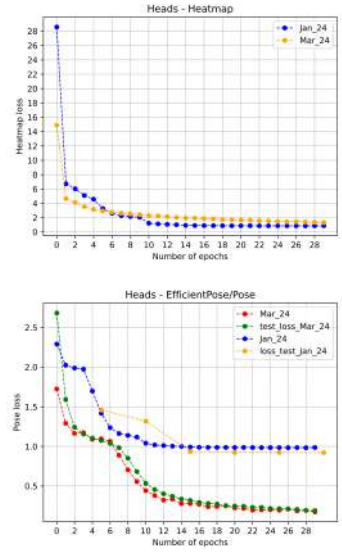


Fig. 13. Heatmaps and EfficientPose loss comparison between trial and error approach and hyperparameter optimization

Although the detection and Heatmaps techniques show similar behavior in both sessions, the first setting has a slight advantage in bounding box detection and classification, as it has earlier learning rate drop settings at 15 and 18 steps [16]. The biggest difference between the two models is noticed in the pose estimation, where the first model stabilizes at a loss of 0.92, while the second model follows a descending trend, resulting in a loss value of 0.19.

Synthetic dataset + Data Augmentation					
Resolution	**#Rej**	**IoU [-]**	E_t **[m]**	E_q **[deg]**	$E_{\textbf{pose}}$ **[-]**
1024x768	15	0.907	1.38	12.07	0.922
1024x768	1	0.925	0.498	2.727	0.186

Table 6. Performance comparison of PEN performances using trial and error approach (first row) and hyperparameter optimization (second row)

The above table compares model performance before and after applying hyperparameter optimization. The parameters include resolution (1024x768), number of rejections (#Rej), Intersection over Union (IoU), translation error (E_t) in meters, quaternion error (E_q) in degrees, and overall pose error (E_{pose}). The second model shows significant improvements, with only 1 rejection, an increased IoU of 0.935, reduced translation and quaternion errors of 0.42m and 2.386 degrees, respectively, and a dramatically lower pose error of 0.050.

Optuna provides a range of visualization tools to help users analyze and interpret optimization results. These tools include visualizations for understanding the search space, tracking the performance of trials, and identifying the influence of individual hyperparameters. In particular, the *feature importance plot* and *contour plot* are chosen to visualize some obtained results.

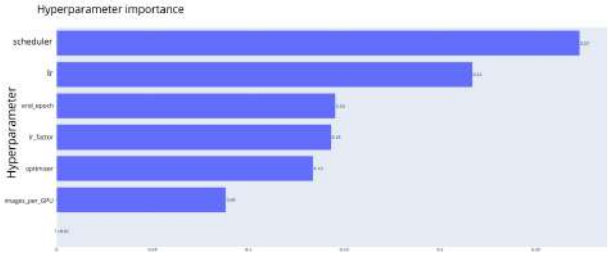


Fig. 14. Feature importance plot

The *feature importance plot* offers insights into the relative importance of different hyperparameters in influencing model. The major contribution is given by the scheduler (0.27). The value of the learning rate follows with a variance of 0.22 up to the batch size that turns out to be the hyperparameter that less impacts the performances of the model.

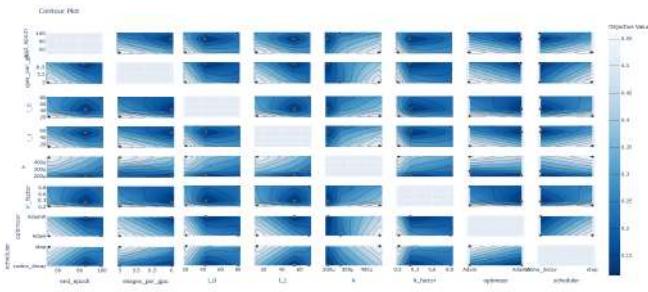


Fig. 15. Contour plot

A *contour plot* depicts instead the relationship between two hyperparameters and the objective function, as shown by the color gradient.By analyzing the contour plot, it is possible to detect how variations in the hyperparameters influence changes in the objective function, thereby identifying promising regions within the hyperparameter space.A bar on the graph's side specifies the objective value scale, which ranges from white (about 0.65 as the loss value) to dark blue (roughly 0.09). Darker regions within the plot corresponds to the best values for each hyperparameter (e.g. 7 as batch size or 100 as number of epochs) since they correlate to lower values of the objective function.

5.2 WSE dataset VS Final Approach dataset

The results of the two different datasets are compared in the table and graphs below. The first dataset is the one corresponding to the Walkin Safety Ellipses, while the second dataset is the one represented the Final Approach of the chaser and the target. The results are obtained using the same hyperparameters - obtained from the hyperparameter optimization study - and the same network configuration. It is important to note that the upper graph shows 100 epochs, while the lower graph represents only 50 epochs. Even though the final approach dataset has a lower pose loss than heatmap loss values, its curve follows the same descending trend as the lower graph. The main difference lies in the values of loss heads (0.085 VS 0.005 for pose loss). This can be attributed to many reasons: firstly, the final approach dataset is trained using pre-trained weights from the general dataset, allowing it to learn faster. Secondly, while the resolution is the same, the final approach dataset has a less cluttered scene (the Earth is barely visible in the beackground) with adjustments made to create a more photorealistic dataset. Also, following a straight-line trajectory, the pose of the final approach dataset is *easier* to learn. Both datasets are constituted of synthetic images.

Synthetic dataset + Data Augmentation					
Resolution	**#Rej**	**IoU [-]**	E_t **[m]**	E_q **[deg]**	E_{pose} **[-]**
1024x768	1	0.935	0.418	2.386	0.085
1024x768	1	0.992	0.213	0.035	0.005

Table 7. Performance comparison of PEN performances between WSE dataset (first row) and Final Approach dataset (second row)

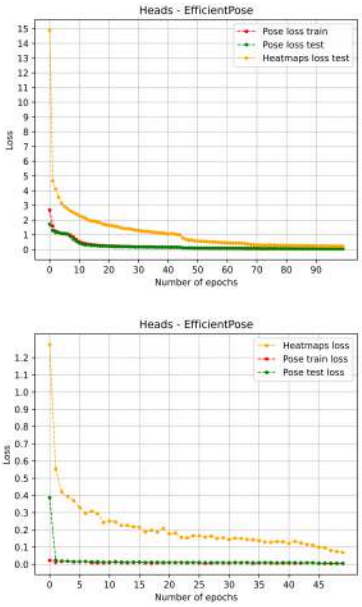


Fig. 16. Heatmaps VS EfficientPose loss comparison on the general dataset on the left and the final approach on the right

6. Conclusions

Accurate determination of spacecraft positioning and orientation is critical for proximity space operations. This study presents a CNN-based architecture that aims to improve non-cooperative but known spacecraft pose estimation by effectively addressing the challenges associated with real-time applications and computational constraints. The network demonstrates impressive performance metrics, achieving centimeter-level position accuracy and near-degree-level attitude precision, while exhibiting robust resilience to varying lighting conditions and background textures.

Significant challenges remain, particularly regarding the need for extensive datasets and the time-consuming process of dataset creation, as well as the domain gap between synthetic and real-world data. Future improvements will focus on expanding the dataset to further en-

hance the network's performance by incorporating realistic space mission disturbances, as well as adding a segmentation head, along with the target's CAD model to boost accuracy while maintaining acceptable computational load. Testing the design on real hardware could provide valuable insights into its performance on onboard computers, validating the design, and refining hyperparameter optimization techniques to further improve performance under real-time constraints.

By analyzing and optimizing hyperparameters, this research not only improves the CNN's performance but also contributes valuable insights into the correlation between hyperparameters and the pose loss function. The synthetic dataset of 25,000 images generated via Blender serves as a critical foundation for training the network, showcasing the effectiveness of simulation-based training in achieving high accuracy levels. This work lays a groundwork for advancing pose estimation techniques in non-cooperative space scenarios, highlighting the potential for future innovation in this field.

References

[1] M. A. Viscio *et al.*, "Interplanetary cubesats mission to earth-sun libration point for space weather evaluations," vol. 2, Jan. 2013.

[2] S. Corpino *et al.*, "Space rider observer cube - sroc: A cubesat mission for proximity operations demonstration," in *Proceedings of the 73rd International Astronautical Congress, IAC 2022*, Code 190266, Paris, 2022.

[3] P. Lepcha *et al.*, "Assessing the capacity and coverage of satellite iot for developing countries using a cubesat," *Applied Sciences*, vol. 12, no. 17, p. 8623, 2022. DOI: 10.3390/app12178623. [Online]. Available: https://doi.org/10.3390/app12178623.

[4] N. Schwartz *et al.*, "6u cubesat deployable telescope for optical earth observation and astronomical optical imaging," in *Proc. SPIE 12180, Space Telescopes and Instrumentation 2022*, 2022. DOI: 10.1117/12.2627248. [Online]. Available: https://doi.org/10.1117/12.2627248.

[5] J. Bowen, M. Villa, and A. Williams, "Cubesat based rendezvous, proximity operations, and docking in the cpod mission," in *Proceedings of the Small Satellite Conference*, Logan, Utah, Aug. 2015.

[6] M. Richard-Noca *et al.*, "Developing a reliable capture system for cleanspace one," in *Proceedings of the International Astronautical Congress*, Guadalajara, Mexico, Oct. 2016.

[7] M. Pecorilla and F. Stesina, "Novel extended kalman filter + h∞ optimal output feedback control configuration for small satellites with high pointing stability requirements," *International Journal of Robust and Nonlinear Control*, vol. 34, no. 5, pp. 2991–3010, 2024. DOI: 10.1002/rnc.7119.

[8] F. Branz, L. Olivieri, F. Sansone, and A. Francesconi, "Miniature docking mechanism for cubesats," *Acta Astronautica*, vol. 176, pp. 510–519, 2020. DOI: 10.1016/j.actaastro.2020.06.042. [Online]. Available: https://doi.org/10.1016/j.actaastro.2020.06.042.

[9] F. Stesina, "Tracking model predictive control for docking maneuvers of a cubesat with a big spacecraft," *Aerospace*, vol. 8, no. 8, p. 197, 2021. DOI: 10.3390/aerospace8080197. [Online]. Available: https://doi.org/10.3390/aerospace8080197.

[10] F. Stesina, S. Corpino, C. Novara, and S. Russo, "Docking manoeuvre control for cubesats," *Journal of the Astronautical Sciences*, vol. 69, no. 2, pp. 312–334, Apr. 2022. DOI: 10.1007/s40295-022-00307-1.

[11] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, 2018, pp. 1–10. DOI: 10.1109/ICOMET.2018.8346440.

[12] D. Barath and Z. Kukelova, *Relative pose from sift features*, 2022. arXiv: 2203.07930 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2203.07930.

[13] T. H. Park, S. Sharma, and S. D'Amico, "Towards robust learning-based pose estimation of noncooperative spacecraft," *CoRR*, vol. abs/1909.00392, 2019. arXiv: 1909.00392. [Online]. Available: http://arxiv.org/abs/1909.00392.

[14] B. Chen, J. Cao, A. Parra, and T.-J. Chin, "Satellite pose estimation with deep landmark regression and nonlinear pose refinement," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct. 2019.

[15] T. H. Park, M. Märtens, G. Lecuyer, D. Izzo, and S. D'Amico, "Speed+: Next-generation dataset for spacecraft pose estimation across domain gap," in *2022 IEEE Aerospace Conference (AERO)*, 2022, pp. 1–15. DOI: 10.1109/AERO53065.2022.9843439.

[16] T. H. Park and S. D'Amico, "Robust multi-task learning and online refinement for spacecraft pose estimation across domain gap," *Advances in Space Research*, 2023, ISSN: 0273-1177. DOI: https://doi.org/10.1016/j.asr.2023.03.036. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0273117723002284.

[17] K. Black, S. Shankar, D. Fonseka, J. Deutsch, A. Dhir, and M. R. Akella, "Real-time, flight-ready, non-cooperative spacecraft pose estimation using monocular imagery," 2021. [Online]. Available: https://arxiv.org/abs/2101.09553.

[18] G. D. D'Ortona, "Relative visual navigation based on CNN in a proximity operation space mission," Materials Research Forum LLC, Jan. 2023.

[19] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020. DOI: 10.3390/info11020125. [Online]. Available: https://www.mdpi.com/2078-2489/11/2/125.

[20] P. T. Jackson, A. Atapour-Abarghouei, S. Bonner, T. P. Breckon, and B. Obara, "Style augmentation: Data augmentation via style randomization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 83–92.

[21] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.