International Journal of Aeronautical and Space Sciences
https://doi.org/10.1007/s42405-023-00703-3

IJASS International Journal of Aeronautical and Space Sciences

ORIGINAL PAPER

Transformer Network-Aided Relative Pose Estimation for Non-cooperative Spacecraft Using Vision Sensor

Jamal Ahmed[1] · Awais Arshad[1] · Hyochoong Bang[1] · Yoonhyuk Choi[2]

Received: 10 November 2023 / Revised: 17 December 2023 / Accepted: 20 December 2023

Abstract
The objective of the proposed work is to perform monocular vision-based relative 6-DOF pose estimation of the non-cooperative target spacecraft relative to the chaser satellite in rendezvous operations. In this work, the convolutional neural network (CNN) is replaced by the high-resolution transformer network to predict the feature points of the target satellite. The self-attention mechanism inside the transformer provides the advantage of overcoming the inadequacies of the translation equivariance, 2D neighborhood awareness, and long-range dependencies in CNN. First, the 3D model of the target satellite is reconstructed using the inverse direct linear transform (IDLT) method. Then, the pose estimation pipeline is developed with a learning-based image-processing subsystem and geometric optimization of the pose solver. The image-processing subsystem performs target localization using CNN-based architecture. Then, the key points detection network performs regression to predict 2D key points using the transformer-based network. Afterward, the predicted key points based on their confidence scores are projected onto the corresponding 3D points, and the pose value is computed using the efficient perspective-n-point method. The pose is refined using the non-linear iterative Gauss–Newton method. The proposed architecture is trained and tested on the spacecraft pose estimation dataset and it shows superior accuracy both in translation and rotation values. The architecture has shown robustness against the drastically changing clutter background and light conditions in the space images due to the self-attention mechanism. Moreover, this method consumes less computation resources by using fewer floating-point operations and trainable parameters with low input image resolution.

Keywords Monocular vision · Pose estimation · Perspective-n-point · Gauss–Newton method · Convolution neural network · Transformer

1 Introduction

In recent times, spacecraft proximity operations have gained importance with the increase in interest in the field of space robotics. Space missions like Apollo, Hubble Telescope, and the International Space Station have performed proximity operations in orbit and are serviced by the onboard astronauts that travel to these space stations from the Earth. However, due to huge budget requirements, the demand arises to develop unmanned autonomous systems that can perform proximity operations [1–4]. In addition, the increasing interest in space exploration and space applications has given a huge boost to the launch of new satellites in low earth orbit. Few satellites complete their life cycle and go to graveyard orbit or re-enter the earth's atmosphere. Many satellites become non-functional before the end of life and turn into space debris [5]. As a result, there is a lot of congestion in space and the possibility of collision between non-cooperative and working spacecrafts has increased enormously. Currently, research is in progress to develop debris removal methods and ESA is working on a Clearspace-1 mission to remove space debris with the help of a robotic arm and has a plan to launch it in 2026 [6].

A key initial step to perform the above-mentioned rendezvous operations is to estimate the relative position and orientation of the non-cooperative target satellite relative to the servicing spacecraft [7]. For close to medium-range rendezvous operations, different vision sensors are used for pose estimation of the non-cooperative target; some are active sensors like LIDARS and some are passive sensors like stereo

✉ Hyochoong Bang
hcbang@kaist.ac.kr

[1] Korea Advanced Institute of Science and Technology, Daejeon, Korea

[2] Agency for Defense Development, Daejeon, Korea

Published online: 17 March 2024

and monocular sensors [8]. In this work, a monocular camera is chosen over the others because it is a passive sensor and less susceptible to sun illumination and it has low power and mass budget. It is the same sensor that is used for star tracker and it is already available on chaser satellite. However, complex image-processing is required to get the desired pose [9].

The problem of estimating the pose of the non-cooperative spacecraft using a monocular vision sensor has been addressed in the literature through different approaches [10–15]. The first approach is to use model-based pose estimation in which 3-D model points of the target object are known. In this method, first, the 2-D image is captured by the camera on the chaser satellite and then the image-processing algorithm extracts the key points in the image. Then project the 3D model points onto the 2D key points to get the pose value by reducing the reprojection error through the pose solver. The key points can be extracted either through conventional or learning-based methods.

Sharma et al. [13] use a conventional image-processing method to extract 2D feature points. This method filters out the weak gradients and then uses the Hough transform to find out the edges of the target body. Then edges are combined in different shapes similar to the ones already stored in the memory from the 3D model. The pose is calculated using 3D–2D projection through efficient perspective-n-point (EPnP) [14] and Newton–Raphson method. This method can only detect 20% of the images and takes a long time to develop 3D–2D correspondence. Chen et al. [15] propose a learning-based convolution neural network (CNN) method for landmarks identification. faster-RCNN along with the HRNET network is used to localize the target and then use HRNET to extract the landmarks position. The pose is solved using EPnP and a simulated annealing scheme (SA-LMPE). The image resolution of 768 $\times$ 768 is adopted and presents a very low pose error. They have won Kelvins pose estimation challenge in 2020. This work seems to be good for simulation but high image resolution and the use of more floating-point operations make this method unsuitable for space-grade hardware. Piazza et al. [16] use CNN for key points detection. First, the target is localized using the Yolo network and then key points are detected using HRNET architecture. The 3D model points are projected onto the predicted 2D key points using EPnP and Levenberg–Marquardt (LMM) pose solvers. This work uses an image size of 416 $\times$ 416 and shows centimeter/degree-level accuracy on the spacecraft pose estimation dataset (SPEED). This method achieves less pose accuracy as compared to Chen [15] but also consumes less hardware resources due to the use of a single-stage detector for the bounded box. Wang et al. [17] replace the CNN with a transformer network for key features extraction. The network is based on a detection transformer (DETR) that was originally designed for object detection. This work proposes a bipartite matching loss function for key points that enables it to output the index of key points instead of a fixed index as in CNN. This work consumes a lot of hardware resources and has more than 200 million trainable parameters that make this architecture hard to use for space applications. There are a few other architectures related to transformers that are available in literature like a swin transformer [32], and vision transformer [33], etc. Swin transformer [32] is based on the hierarchical design that breaks the input image into small patches. Then use shift windows to develop a long-range relationship between adjacent pixels. The swin block is also introduced which has attention layers and feed-forward convolution layers. The other architecture is the vision transformer [33] which is probably the first work in vision based on transformers. In this architecture, the input image is distributed into small patches that are called tokens, and position is embedded in each token. Then multihead self-attention layer is used which assigns weight to the different pixels of the image to know about neighborhood awareness. However, transformers [32, 33] have more trainable parameters as compared to same-size CNN architecture.

The second approach is to estimate the pose using end–end CNN learning-based methods that do not require any separate pose solver and directly regress the key points to get the pose values. In this method, the prior information of the target 3D model is not required. Sharma et al. [18, 31] present the baseline solution on the SPEED dataset using the end–end pose estimation approach. The spacecraft localization network (SPN) is proposed to detect bounded box through branch 1 and then branch 2 and branch 3 use classification and regression to estimate the relative attitude. The position is determined with the help of constraint forced by the bounded box and the relative attitude. The SPN is specifically designed for the pose estimation of satellites so it has fewer parameters to train but it is compromised on accuracy as compared to previous commercial off-the-shelf networks. Garcia et al. [19] propose LSPnet end-end architecture for the pose estimation. It has the UNet network and the orientation CNN to determine position and orientation, respectively. Both the above approaches serve the purpose of estimating the pose; however, the combination of learning-based key points detection and geometrical pose solver is considered more accurate and has less trainable parameters as compared to the end-end network.

In summary, the existing state-of-the-art methods show progress toward solving the challenges that arise during the rendezvous operations; however, there are a few issues that still need consideration. First, the space environment is very harsh due to drastic changes in background, brightness, scale variation, contrast, and sun illumination that make it difficult to detect the boundaries of the body. Second, the order and index of 2D points are required to develop the 3D-2D correspondence otherwise the process will become iterative. Third, deep learning-based networks usually consume

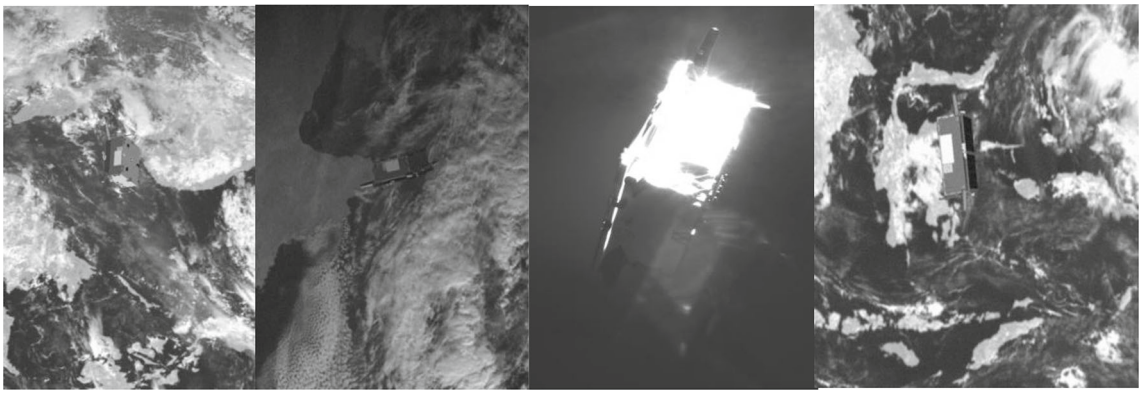


Fig. 1 Images from the SPEED dataset that shows a challenging space environment

more hardware resources that are not suitable for space-grade hardware. Fourth, the domain gap exists between real and synthetically generated space-borne datasets that affect the performance and accuracy of learning-based architectures [21, 23]. Figure 1 shows a few challenging images in which it is hard to identify the satellite [22].

To address these challenges, this paper presents the pose estimation pipeline that comprises of self-attention-based transformer architecture for key points detection along with mathematical modeling of the pose solver. Self-attention mechanism is useful for learning 2D neighborhood awareness and translation equivariance in key points detection. The generalization to global features enables the architecture to detect key points efficiently in the space environment and achieves comparable accuracy in comparison to state-of-the-art techniques by utilizing fewer hardware resources. Floating point operations (FLOPs) and learnable parameters are reduced significantly for key points detection. Moreover, learning-based architecture has the number of output channels equivalent to the number of key points that enables indexing with each predicted key point and linear mapping to the corresponding 3D point. The motivation is to improve the existing system with the use of a deep learning framework that can provide more accurate and robust pose estimation results for future technology development related to proximity operations.

The rest of the paper is organized as follows: Sect. 2 presents the perspective-n-point problem statement and Sect. 3 describe the details of the SPEED dataset that is used in this work for training and testing purpose. Section 4 elaborates on the technical design of relative pose estimation architecture including 3D model reconstruction, image-processing pipeline, and pose solvers. Section 5 provides information about evaluation metrics and Sect. 6 describes the training and testing results of the architecture. Section 7 compares the results of our architecture with state-of-the-art methods and the conclusion is provided in Sect. 8.

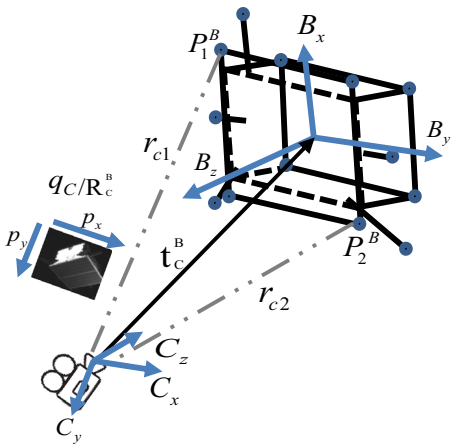


Fig. 2 Geometrical model of the PnP problem

2 Problem Statement

Suppose a rendezvous scenario has two spacecraft: one is the non-cooperative target satellite with reference frame B and the other one is a chaser satellite on which a camera is attached having reference frame C. The problem is to estimate the relative position and orientation of the target relative to the camera where t_C^B is a translational vector that defines the relative position from the origin of the camera to the center of the target and R_C^B is the rotation matrix that defines the rotation of the target to the camera. The problem statement is defined in Fig. 2. Let P_1^B, P_2^B, ...P_i^B are the 3D key points on the target body and ith 3D point P_i^B can be expressed in the camera frame as

$$r_{ci} = R_C^B P_i^B + t_C^B, \tag{1}$$

$$r_{ci} = \left[x_{ci},\ y_{ci}, z_{ci}\right]^T, \tag{2}$$

$$r_{ci} = \left[\frac{x_{ci}}{z_{ci}},\ \frac{y_{ci},}{z_{ci}},\ 1\right]^T, \tag{3}$$

where, r_{ci} are the points in the camera frame that represent the 3D points P_i^B on target body with respect to R_C^B and t_C^B. Assuming a pinhole camera,r_{ci} is expressed in the image plane $p_{img} = [p_x,\ p_y]^T$ by using perspective-n-point (PnP) Eq. (5) given that focal lengths of the camera are f_x,f_y and camera principal points are C_x and C_y:

$$p_{\text{img},i} = [p_{xi},\ p_{yi}]^T, \tag{4}$$

$$p_{\text{img},i} = \left[\frac{x_{ci}}{z_{ci}} f_x + C_x,\ \frac{y_{ci},}{z_{ci}} f_y + C_y,\ 1\right]^T, \tag{5}$$

where, i is the number of 3D points,p_x, p_y are aligned with C_x, C_y respectively and C_z is pointing in the direction of the target satellite. Equation (5) can be presented in homogeneous coordinates with camera intrinsic matrix K, camera extrinsic matrix M_{ext}, and 3D points P^B as

$$p_{\text{img},i} = K.M_{\text{ext}}.P_i^B, \tag{6}$$

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_C^B(1,1) & R_C^B(1,2) & R_C^B(1,3) & t_C^B(1) \\ R_C^B(2,1) & R_C^B(2,2) & R_C^B(2,3) & t_C^B(2) \\ R_C^B(3,1) & R_C^B(3,2) & R_C^B(3,3) & t_C^B(3) \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}. \tag{7}$$

6-DOF is required to define the relative position t_C^B and orientation R_C^B of the target satellite as defined in Eq. (7). Hence, a minimum of 6 points correspondence is required to uniquely determine the pose (R_C^B and t_C^B) solution.

3 Spacecraft Pose Estimation Dataset

For this work, the spacecraft pose estimation dataset (SPEED) [22] and SPEED+ [23] datasets are used to train and test our architecture. SPEED dataset is generated by SLAB from Stanford University in collaboration with the European Space Agency (ESA). The first version is named SPEED, which was released in 2019, and the 2nd version, which was released in 2022 was named SPEED+. SPEED dataset uses TANGO spacecraft from PRISMA mission and all the images in the dataset are in grayscale format. The distribution of images is shown in Table 1. The dataset only provides ground truth labels for training and real images so in this work test set is developed by selecting 3000 random images from a training set of SPEED images.

The range of distance between the target satellite and the camera is between 3 and 45 m and the relative attitude is uniformly distributed random rotations in SO (3) space. The synthetic images are generated using an OpenGL rendering simulator. In the simulator, the Tango spacecraft is rendered with plain black ground in almost half of the synthetic images, and the remaining half of the images are rendered with earth and clouds in the background. The real images are taken in the lab environment with a 1:1 mockup model of the Tango spacecraft. The intrinsic parameters of the camera (K) that is used for taking pictures are displayed in [22].

4 Methodology

The spacecraft pose estimation architecture mainly consists of the image-processing pipeline, pose solvers, and 3D model of the known target. The image-processing system takes input images from a monocular camera and outputs specific 2D feature points. It has deep learning-based subsystems for satellite localization and feature points detection. The pose solver uses these 2D feature points and a 3D model of the target spacecraft to get the pose values of the target with respect to the client satellite. The 3D model of the target satellite is unknown and it is reconstructed using the method of triangulation from 2D images. The top-level vision-based pose estimation architecture is shown in Fig. 3.

4.1 3D wireframe model reconstruction

The 3D wireframe model of the target spacecraft is constructed using a method of triangulation. In this method, the 14 images are randomly selected from the dataset in which the maximum number of visible features can be identified. Then 13 key features are manually annotated in each image that consists of four top and bottom corners of the satellite, three corners of the antennas, and two corners of the GPS receiver antennas. The 3D model points P^B are obtained by solving Eq. (7) using inverse direct linear transform (IDLT) [24]. This method is used because the camera intrinsic matrix (K) and the true pose of the selected images are provided. The camera extrinsic matrix M_{ext} is converted into the pose matrix $H_{\text{sat}}^{C_n}$ as shown:

$$H_{sat}^{C_n} = \begin{bmatrix} R_{11}^{C_n} & R_{12}^{C_n} & R_{13}^{C_n} & t_x^{C_n} \\ R_{21}^{C_n} & R_{22}^{C_n} & R_{23}^{C_n} & t_y^{C_n} \\ R_{31}^{C_n} & R_{32}^{C_n} & R_{33}^{C_n} & t_z^{C_n} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{8}$$

Table 1 SPEED dataset distribution

Image set	Train	Test	Real	Real_test
images	12,000	2998	05	300

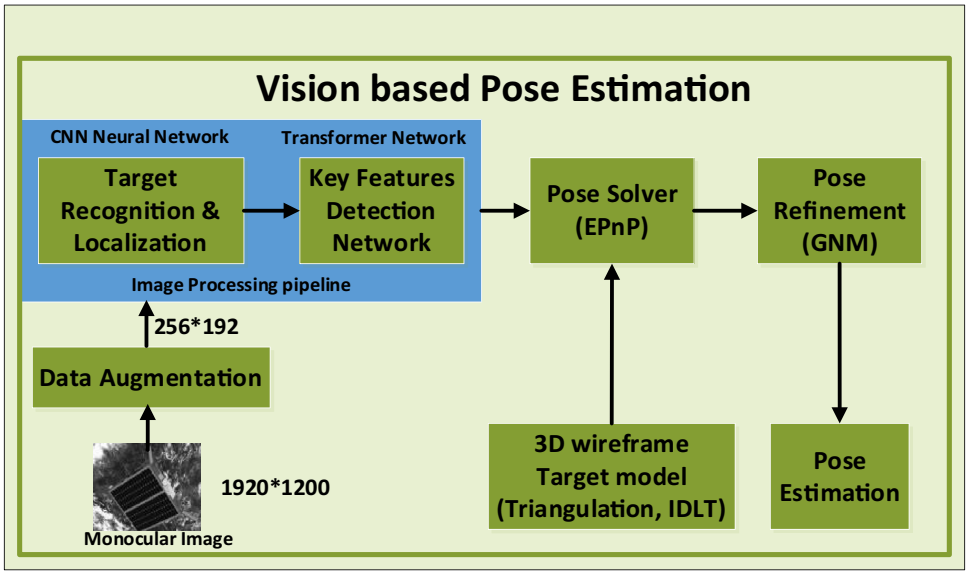


Fig. 3 Vision-based pose estimation architecture

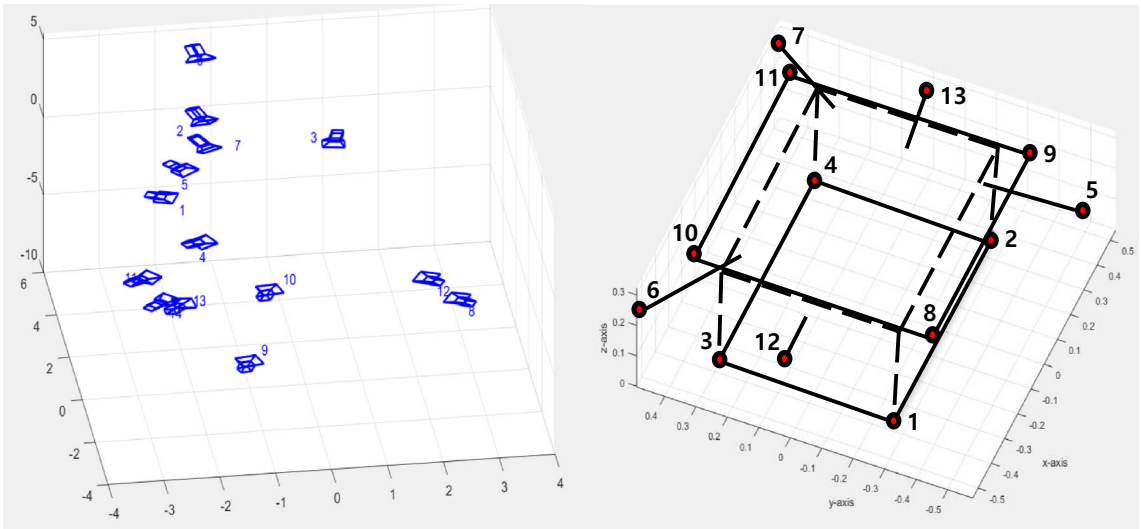


Fig. 4 **a** 14 cameras view to the target body; **b** a 3D reconstructed model of the target satellite

where $H_{\text{sat}}^{C_n}$ is the pose of the target satellite w.r.t. nth camera. The 3D reconstructed model along with the view of 14 cameras that are looking at the target from different viewing angles is shown in Fig. 4. These cameras are the 14 images that were selected earlier. In total, 26 equations are required to get thirteen 3D model points. Equation (7) is re-written in the format $Ax = b$ for n number of cameras to get true 3D model points P^B (X, Y, Z) as presented

$$\begin{bmatrix} R_{11}^{C_n} - x^{C_n} R_{31}^{C_n} & R_{12}^{C_n} - x^{C_n} R_{32}^{C_n} & R_{13}^{C_n} - x^{C_n} R_{33}^{C_n} \\ R_{21}^{C_n} - x^{C_n} R_{31}^{C_n} & R_{22}^{C_n} - x^{C_n} R_{32}^{C_n} & R_{23}^{C_n} - x^{C_n} R_{33}^{C_n} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} -t_x^{C_n} + x^{C_n} t_z^{C_n} \\ -t_y^{C_n} + y^{C_n} t_z^{C_n} \end{bmatrix}, \text{for } n = 14. \tag{9}$$

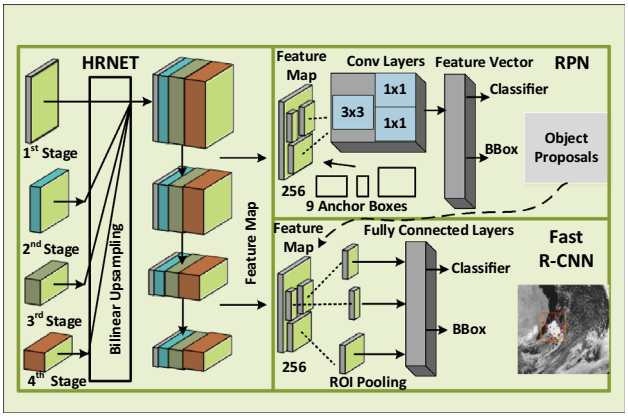


Fig. 5 Target localization architecture with a feature extractor, RPN, and object detector

4.2 Deep Learning-Based Image-Processing Pipeline

In the image-processing pipeline, the input image of size 1920 × 1200 is first pre-processed and different data augmentation techniques are applied e.g., image resizing, cropping, rotation, random brightness change, random contrast change, gaussian noise, and data normalization to reduce the image size to 256 × 192 pixels. The reduced image is passed to the CNN-based target recognition and localization network that separates the satellite from the image background and draws a bounded box around it. In the next step, the area under the bounded box is cropped, resized to 256 × 192 pixels, and passed to the key features detection network that is based on the transformer network. The features detection network tries to detect the maximum possible feature points among the 13 key points that are specified earlier.

4.2.1 Target Recognition and Localization

Target recognition and localization is an object detection (OD) task that draws a bounded box around the region of interest (ROI) in the image. In this work, the faster-RCNN [25] network is used for the detection task in which the original backbone network i.e., VGG is replaced by the HRNET-W18 [26] network to extract feature maps in the image. The HRNET is used because it is a high-resolution network that downsamples the image in a parallel manner in each layer instead of sequential downsampling as in Resnet architecture. Parallel down sampling preserves the spatial information of the image that is required to get better performance and accuracy. The architecture of the OD network is shown in Fig. 5. The backbone HRNET-w18 network has four parallel branches and each subsequent branch is low in resolution as compared to the previous branch. These four branches are connected through fuse layers to share data. The output of a multiresolution network is more precise even for the cases in which the target object is very small. It is unlike most CNN architectures that reduce the image resolution in each layer sequentially, extract the features, and then upsample the image resolution to get back the original image. For simplicity, only the last part of HRNET for feature map extraction is shown in Fig. 5 in which the output has four heads with different resolutions. All the heads are passed through bilinear up sampling and merged to output feature map. The merged heads are downsampled again to manage different sizes of feature maps.

First, the input image of size 256 × 192 is passed through the CNN backbone (HRNET-w18) network that extracts the feature maps and passes to two different stages; region proposal network (RPN), and object detector (FAST-RCNN). The region proposal network (RPN) takes the feature map as input from the HRNET and the kernel runs over all the feature maps in a sliding manner to output potential rectangular object proposals along with the confidence scores. The fixed set of 9 anchor boxes with different scales and aspect ratios are used for each pixel to determine the presence of an object. The RPN has convolution layers with a kernel size of 3 × 3 and two fully connected layers with a size of 1 × 1. The predictions of the RPN for each image pixel is (H * W) * number of anchor boxes * (2 + 4).

The FAST-RCNN (object detector) stage accepts object proposals from the RPN network and the feature map from the HRNET. The ROI pooling layer tries to fit region proposals to the corresponding feature map by dividing the proposals into a fixed number of windows and through maximum pooling. The output size of the ROI layer is (N, 7, 7, 256), where N are the proposals from the RPN stage. Then ROI output is fed to the fully connected layers (FCN) to perform classification and regression in two separate branches. The softmax classifier is used to identify the class of the feature map along with the confidence score. The regression branch outputs four coordinates of the bounded box. The loss function that is used for both classification and the bounded box is presented as:

$$L(c_j^p, b_j^p) = \frac{1}{n_{\text{class}}} \sum_j L_{\text{class}}(c_j^p, c_j^g) + \lambda \frac{1}{n_{\text{reg}}} \sum_j c_j^g L_{\text{reg}}(b_j^p, b_j^g), \tag{10}$$

where c_j^p, c_j^g are the predicted and ground truth classes and b_j^p, b_j^g are the predicted and ground truth bounded boxes respectively. L_{class} is the classification loss and L_{reg} is L_1 loss for bounded boxes. The network outputs multiple potential candidates for classes and bounded boxes so non-maximum suppression (NMS) with a threshold of 75% is used to select the ones with the highest confidence score and ignore others to get the final output.

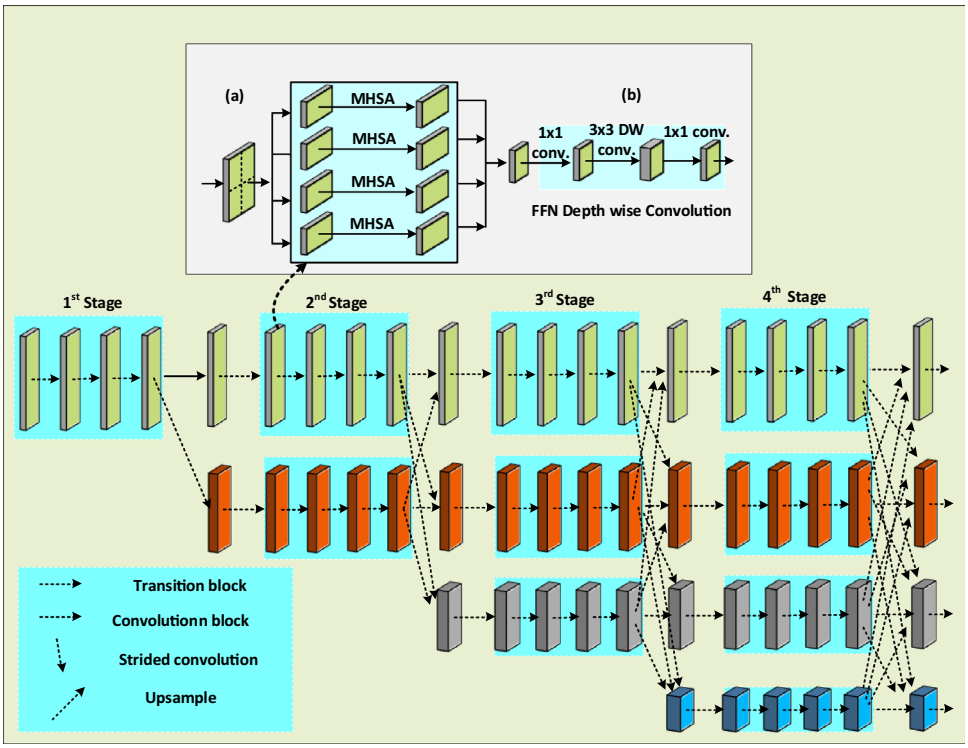


Fig. 6 HRFORMER architecture shows the four stages of the model. The transformer blocks are used in the areas highlighted with blue and the information exchange blocks are used at the end of each stage to share information with low-resolution layers. **a** Parallel self-attention windows; **b** FFN 3 × 3 depth-wise convolution block to share information with multiresolution layer

4.2.2 Key Features Detection Network

The key features detection network predicts the key points as heatmaps of size 64 × 48 in the input image of size 256 × 192. The CNN-based high-resolution network (HRNET-w32/48) [26] has shown great performance for terrestrial datasets in the applications of key points extraction and pose estimation. The network is also utilized by [15, 16] for the spaceborne SPEED dataset. However, CNN-based networks generalize to local features and develop a bias toward training images. In this work, the CNN network is replaced with the HRFORMER [27] transformer network. The transformers are initially developed for language models but later many architectures are proposed for vision applications as well like ViT, DETR, and BERT, etc. The transformers have shown better performance than CNN because they use a self-attention mechanism that makes the network aware of its 2D neighborhood based on the similarities in different parts of the image. Transformers not only learn about the local features but also learn about global features of the input and it is imperative for changing the space environment. So, if a network learns global features or has the capability of generalized learning then different types of inputs can be dynamically adjusted to predict key points that can improve accuracy. HRFORMER maintains the high resolution throughout the network using a multiresolution parallel design that helps the architecture process different resolution images (multiscale images). In addition, each branch starting from stage 2 has local self-attention windows to develop long-range dependencies. The architecture has a multiscale fusion module that is used to exchange information between different resolutions and as a result, HRFORMER has both local and global features. The HRFORMER [27] architecture is shown in Fig. 6.

The architecture has four stages with four parallel multiresolution layers and it is implemented using the top-down approach in which first the target object is isolated by drawing a bounded box around it then key points in the localized target are identified. The down sampling takes place with stride = 2 from high resolution to low-resolution layer at the time of connection with the fuse layer. The number of channels or the depth of the branch increases as the resolution in each subsequent branch decreases. The multi-scale fusion layers help in working on the multiresolution images as well. In the first stage, convolution is performed that has four residual units with a width of 64 and have convolution window of 3 × 3 to reduce the width of feature maps. The local self-attention windows are used from stage 2 to stage 4. The architecture in Fig. 6a shows the input feature map is broken down into parallel localized self-attention blocks in each multiresolution layer from stage 2 to stage 4 that is further attached to the feed-forward network (FFN) as shown in Fig. 6b. The self-attention is performed by dividing the input feature map $Y \in R^{N*D}$ into small non-overlapping windows and each

window is of size $K*K$. The multi-head self-attention [27] is performed in each small window by using Eqs. (11–13):

$$\text{multihead}(Y_p) = \text{concatenate}[\text{head}(Y_p)_1, \ldots.\text{head}(Y_p)_H] \in R^{(K^2)*D}, \tag{11}$$

$$\text{head}(Y_p)_h = \text{softmax}\left[\frac{\left(Y_p W_q^h\right)\left(Y_p W_k^h\right)^T}{\sqrt{D/H}}\right].(Y_p W_v^h) \in R^{K^2*(D/H)}, \tag{12}$$

$$\widehat{Y}_p = Y_p + \text{multihead}(Y_p W_o) \in R^{K^2*(D/H)}, \tag{13}$$

where $W_v^h \in R^{(D/H)*D}$, $W_q^h \in R^{(D/H)*D}$, $W_k^h \in R^{(D/H)*D}$, Yoo for $h \in \{1, 2, \ldots H\}$. D is a number of channels, N is input resolution, H represents the number of heads and $\widehat{Y}_p$ represents MHSA. All MHSA are combined to get $[\widehat{Y}_1, \widehat{Y}_2, \ldots.\widehat{Y}_p] \rightarrow Y^{\text{MHSA}}$.

The FFN uses 3×3 depth-wise convolutions and the information exchange block after each stage is not directly linked with self-attention. The relative position encoding is used to embed the position information of features in the self-attention window. The second, third, and fourth layers have 1, 4 and 3 exchange blocks, respectively with a 3×3 convolution window. The architecture has a 8 fuse/exchange blocks overall. The output of the HRFORMER has four heads in parallel with different resolutions that can be used for different applications. The memory and computation complexity of the architecture is reduced from quadratic to linear due to the use of localized parallel self-attention windows in each stage along with depth-wise convolution in FFN blocks.

The output has 13 channels in the form of heatmaps to get the required features. The loss function for the architecture is formulated as mean square error and presented in Eq. (14):

$$\text{loss}_{\text{MSE}}^n = \frac{1}{M}\sum_{i=1}^{M} v_i^{(n)}.(\hat{H}_i^n - H(p_i^n))^2. \tag{14}$$

The nth image is input to the model that has ground truth heatmap $\hat{H}_i^n$ with visibility $v_i^{(n)}$ of the ith heatmap. The Adam optimizer is used to update the weights. The regressed heatmap is then subtracted from the ground truth heatmap. The loss over the entire batch is calculated by taking the average over all the images N as following

$$\text{loss}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N} \text{loss}_{\text{MSE}}^i. \tag{15}$$

4.3 Pose Determination

The problem of predicting the relative pose of the target with respect to camera, given the point correspondences between them, is known as perspective-n-point (PnP). First, the PnP is solved by the use of a closed-form (analytical) linear pose solver that minimizes reprojection error by solving linear mapping between 3D model points and 2D image points in Eq. (7). These pose solvers are computationally fast but do not provide very accurate solutions. Second, the use of a non-linear iterative pose solver reduces the pose error using the least square fit. The iterative solver needs a good initial pose guess to converge at a global minimum in fewer iterations and it is more accurate than the earlier. However, these pose solvers work when the 3D–2D correspondence already exists. The pose value is obtained by solving Eq. (7) and a unique result is only possible when the number of 3D–2D correspondence points is six or greater.

4.3.1 Key Features Selection Criteria

The 3D–2D points correspondence is required before using the pose solver. By exploiting the learning-based key features network, the 2D predicted points are selected based on the confidence score of the key features output. The advantage of a learning-based keypoints network is that it predicts all the 13 points even those that are invisible due to satellite body orientation. However, only those key points are selected for pose solvers whose confidence score is greater than the threshold of 70%. The selected key points are output to the pose solvers along with their index so that these points have a linear mapping with the corresponding 3D model points. The key features selection follows the criteria: (1) key feature confidence score > 70% (2) number of key features ≥ 6.

4.3.2 Closed Form Relative Pose Estimation

First, the closed-form EPnP [14] pose solver is used to estimate the initial guess and it is suitable for both coplanar and non-coplanar 3D–2D key points. This method is considered to be state-of-the-art due to fast convergence to global minima and robustness against outliers. However, 3D–2D correspondence is already established and the pose solution turns out to be accurate. In this method, all the predicted 2D key points are represented as a weighted sum of four coplanar control points as:

$$r_{ci} = \sum_{j=1}^{4} \beta_{ij} c_j^C, \quad \text{for } i = 1, 2, \ldots.n, \tag{16}$$

where r_{ci} are the points in the camera frame that represent the 3D points on the target body with respect to R_C^B and t_C^B.

β_{ij} are the homogeneous coordinates and c_j^C are the 4 control points. Equation (16) is substituted into Eqs. (4) and (5) and acquires the following Eqs. for one corresponding point:

$$\sum_{j=1}^{4} \beta_{ij}(f_x c_{xj}^C + (C_x - p_{xi})c_{zj}^C) = 0, \text{ for } i = 1, 2, \ldots n, \tag{17}$$

$$\sum_{j=1}^{4} \beta_{ij}(f_y c_{yj}^C + (C_y - p_{yi})c_{zj}^C) = 0, \text{ for } i = 1, 2, \ldots n, \tag{18}$$

So, 2n equations are used to solve 12 unknown elements in Eq. (7) that are components of four control points with $n = 6$ correspondences at least. The pose value is computed by re-arranging the above $2n$ equations in the form $Ax = 0$ and solving for 12 unknown elements of R_C^B and t_C^B in x.

4.3.3 Pose Refinement

In the second phase, a non-linear iterative pose solver is adapted to further refine the pose values using the Gauss–Newton method (GNM) [31]. The pose values from EPnP are used as an initial guess for the pose refinement process. GNM is a first-order approximation technique in the Taylor series expansion. The pose error is computed by taking the Euclidean distance between the normalized 2D observed points and projected 3D to 2D points which are expressed as:

$$\text{Pose error} = e_P = \min_{R,t} \sum_{i=1}^{N} \left\| p_n^i - M_{\text{ext}} * P^{B(i)} \right\|_2, \text{ for } N \geq 6, \tag{19}$$

where, $P^{B(i)}$ are the 3D model points on the body of the target satellite, M_{ext} is comprised of R_C^B and t_C^B but converted to $x = \left[\theta^T\ t^T\right]^T$, p_n^i are the normalized 2D image points that are obtained by taking the product between the inverse of camera intrinsic matrix (K) and 2D predicted points as given by

$$p_{\text{norm}}^i = \text{inv}(K) * p^i. \tag{20}$$

GNM solves Eq. (16) by continuously updating the objective function as given below

$$\mathrm{d}x = (J^T J)^{-1} J^T \mathrm{d}y,$$

where $\mathrm{d}y = e_P$,

$$\mathrm{d}x = \left[\theta^T\ t^T\right]^T, \tag{21}$$

and J is the Jacobian matrix that consists of the partial derivatives as shown in Eq. (22). The values of partial derivatives are obtained from Eqs. (1) to (7):

$$J = \frac{\partial e_T}{\partial x} = \begin{bmatrix} \frac{\partial P_1^B}{\partial r_{c1}} \frac{\partial r_{c1}}{\partial t_C^B} & \frac{\partial P_1^B}{\partial r_{c1}} \frac{\partial r_{c1}}{\partial \theta_C^B} \\ \ldots\ldots & \ldots\ldots \\ \frac{\partial P_n^B}{\partial r_{cn}} \frac{\partial r_{cn}}{\partial t_C^B} & \frac{\partial P_n^B}{\partial r_{cn}} \frac{\partial r_{cn}}{\partial \theta_C^B} \end{bmatrix}, \tag{22}$$

$$\frac{\partial P_1^B}{\partial r_{c1}} = \begin{bmatrix} \frac{f_x}{z_c} & 0 & \frac{f_x x_c}{(z_c)^2} \\ 0 & \frac{f_y y_c}{z_c} & -\frac{f_y}{(z_c)^2} \end{bmatrix}, \tag{23}$$

$$\frac{\partial r_{c1}}{\partial t_C^B} = R_C^B . P^B = \frac{\partial R_C^B}{\partial \theta_C^B} P^B. \tag{24}$$

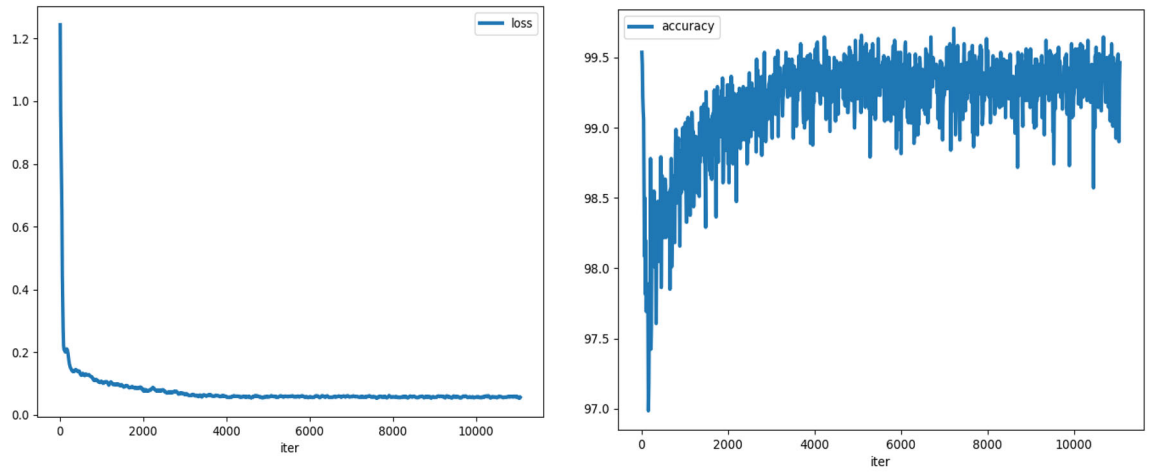
GNM algorithm initializes with the values $x = \left[\theta^T\ t^T\right]^T$ from EPnP pose solver and finds the initial pose error $\mathrm{d}y$. Then $\mathrm{d}x$ value is determined using Eq. (21) and x values are updated as $x = x + \mathrm{d}x$. The iterative process will continue until the pose error falls to 10^{-10} or reaches a set number of iterations.

5 Evaluation Metrics

In this section, the discussion about evaluation metrics is presented about each subsystem. The first subsystem spacecraft recognition and localization are evaluated based on Intersection over union (IOU) and average precision (AP) metrics. IOU predicts the overlapping area between the predicted box and the ground truth box and its range is from 0 to 1. AP is a metric that tells how many predictions are valid on the scale [0–1]. It is desired to have predicted values close to ground truth but it is hard to achieve. In this scenario, the average precision is calculated based on different threshold values of true positives e.g., AP50, AP75, AP50–95, etc. The threshold is based on IOU values. The second subsystem keypoints detection network is evaluated based on object keypoint similarity (OKS) and average precision (AP). OKS is calculated by comparing the Euclidean distance between ground truth key points and predicted key points in a per keypoint manner on the scale [0–1] as given by

$$\text{oks} = \sum_{n=1}^{m} v_n \exp\left(\frac{-d_n^2}{2 * s^2 k_n^2}\right), \tag{25}$$

here, d_n^2 is Euclidean distance between nth predicted and ground truth keypoint, s^2 is a scaling factor depends on the size of the object and s^2 is the area of the object under consideration, k_n^2 is the constant and adjusted based on the application that controls the fall off,v_n is a visibility index



(a) (b)

Fig. 7 **a** Training loss of target localization network; **b** training accuracy of target localization network

per keypoint and m is a total number of keypoints. The pose solver module is evaluated based on translation and rotational error and then both errors are combined to get the pose error. The translation error is calculated by taking the Euclidean distance between the predicted and ground truth translation vector using Eq. (26). The normalized translation error is calculated by dividing the absolute translation error by the Euclidean distance of the ground truth translation vector as presented in Eq. (27):

$$E_t = \left\| t^p_{C/B} - t^g_{C/B} \right\|, \tag{26}$$

$$e_t = \frac{E_t}{\left\| t^g_{C/B} \right\|}, \tag{27}$$

here, $t^p_{C/B}$ is the predicted position vector from the camera to the target body frame, $t^g_{C/B}$ is the ground truth position vector from the camera to the target body frame,E_t is the translation error, and e_t is the normalized translation error. The rotational error is measured both in terms of quaternions and Euler angles and is calculated as:

$$E_q = 2.\arccos(|q^p, q^g|), \tag{28}$$

$$E_{\theta x} = \left|\theta^p_x - \theta^g_x\right|,\ E_{\theta y} = \left|\theta^p_y - \theta^g_y\right|,\ E_{\theta z} = \left|\theta^p_z - \theta^g_z\right|, \tag{29}$$

where q^g, q^p are the ground truth and predicted quaternions,E_q is the rotational error and $\theta^g_{x,y,z}$, $\theta^p_{x,y,z}$ are the ground truth and predicted Euler angles, respectively. The pose error is the sum of translation and rotation error per image and it is calculated by using Eq. (30). Also, the pose error of the entire dataset is calculated by summation of all

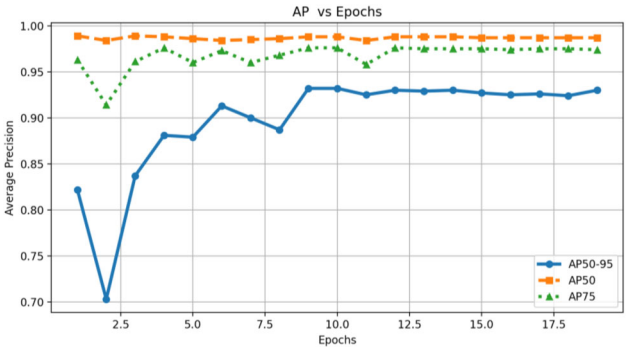


Fig. 8 Bounded box average precision vs epochs

pose errors and divided by the total number of images in the dataset. It is given by Eq. (31):

$$e_p = E_t + E_q,$$

$$e_T = \frac{1}{n}\sum_{p=1}^{n}(E^p_t + E^p_q),$$

where e_p is the pose error per image and e_T is the total pose error over the entire dataset.

6 Training and Results

The target localization network is trained and tested using high resolution network (HRNET-w18) as a backbone and faster-RCNN as a detection head. The training loss of the network is shown in Fig. 7a. The graph shows that the loss dropped significantly during the first 10 iterations because the pre-trained model was used. Then the loss is reduced

smoothly over the remaining iterations and becomes stable at 10,000 iterations. Contrarily, the training accuracy is increasing as loss is in decreasing trend. The training accuracy is shown in Fig. 7b. The sinusoidal type behavior is observed due to a small log time of 10 iterations.

The average precision of AP50, AP75, and AP50-95 on the test dataset is shown in Fig. 8. The graph shows the increase in average precision values over the epochs. The network has shown an average precision of almost 99% and 98% for the threshold of AP50 and AP75. However, AP50-95 is averaged to 93%, approximately. The network is also tested by computing mean and median IOU values. The IOU values depict that 95% of the predicted bounded boxes overlap with the ground truth bounded boxes both in mean and median IOU as shown in Table 2.

The key features detection network is trained and tested with (HRFORMER) network. Figure 9a shows that the key points loss converged to global minima almost after 9000 steps. There is a sudden drop in the start of training the transformer due to the use of the pre-trained model. The network takes a few iterations to adapt to the new dataset. The learning rate of the transformer is lr = 0.000125 and the batch size is 25 images. These are the maximum number of images in a batch size that can be used on the NVIDIA-TITAN XP GPU. The accuracy of the key points detection network on the training set is also monitored along with the loss values as shown in Fig. 9b. The initial accuracy rate is slow and gradually improves over the epochs. This slow accuracy rate at the start may be due to due to the self-attention mechanism of the transformer. After the training, the average precision and average recall values are computed for the test dataset to assess the performance of the network. The average precision and recall values are calculated for the same three threshold values as discussed above and plotted in Fig. 10a. The HRFORMER network attains an accuracy of 98% for AP/AR50 and AP/AR75. However, it has shown an accuracy of 94.2% and 95.87% for AP50:95 and AR50:95, respectively. The AP/AR50:95 is also plotted against the epochs in Fig. 10b to check the performance of the test dataset. The transformer network has attained an accuracy of 95.87% respectively over the period and then gets stable.

The accuracy and OKS values for CNN-based HRNET-w32 are also computed along with the values of HRFOMRER. The OKS values for both CNN and transformer are almost similar and close to 98% accuracy. The OKS value along with AP and AR values are shown in Table 3. The number of flops and training parameters based on the same input image size is also calculated for both CNN and transformer network. The transformer network uses almost half the flops as compared to CNN. Also, the number of parameters in the transformer is 4 times less in comparison to CNN. In a transformer, the reduction in flops and parameters is due to the use of local self-attention windows in parallel with depth-wise convolution in the feed-forward network at stages 2, 3 and 4 of the architecture. The resource utilization summary is shown in Table 4.

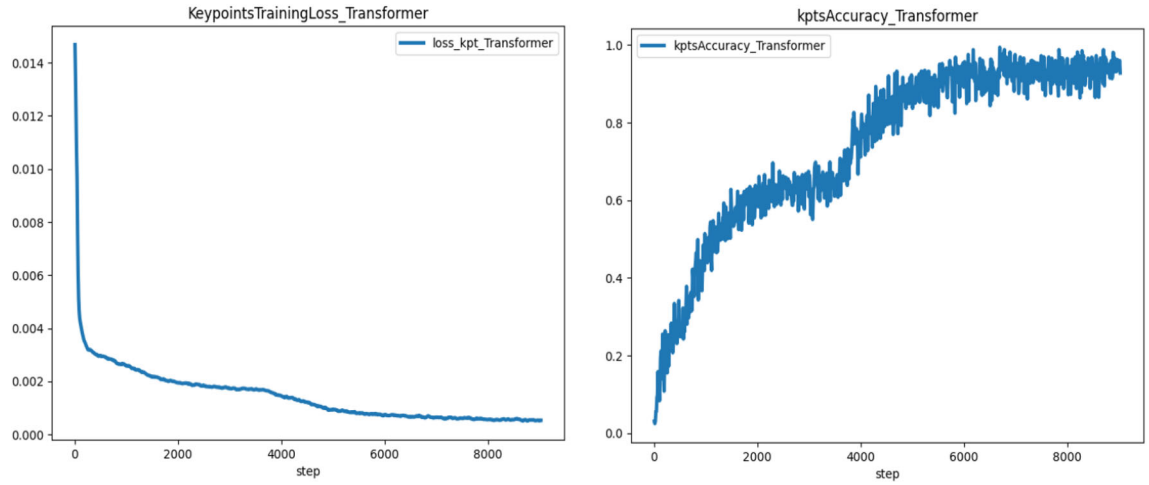


Fig. 9 **a** Training loss to detect key points; **b** keypoints training accuracy

Table 2 Bounded box precision, intersection over union

%	AP50-95	AP50	AP75	Mean IOU	Median IOU
CNN (HRNET-w18 + faster-RCNN)	93	98.8	97.9	95	96

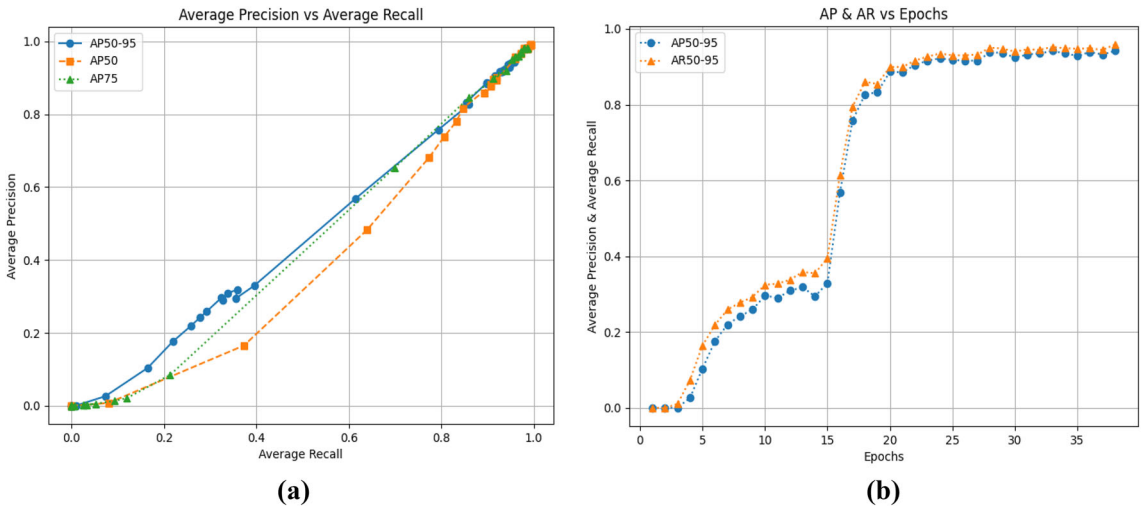


Fig. 10 **a** Average precision vs average recall; **b** average precision-recall vs epochs

Table 3 Precision-recall, OKS transformer

%	AP50-95	AP50	AP75	AR50-95	AR50	AR75	OKS
Transformer	94.20	98	98	95.87	98.67	98.67	98.47
CNN	93.61	98	97.99	94.7	98.6	98	98

Table 4 Resource utilization transformer

Resources	Transformer	CNN
Input image size	256 × 192	256 × 192
Flops	3.33G	7.68G
Parameters	7.73 M	28.53 M

The output images from the image-processing pipeline are shown in Fig. 11. HRFORMER has performed better than HRNET where key points are not visible. The images are of three types: in the first type, the background is plain black and the satellite is at a near distance (4–8 m) from the camera. The detected key points in these images have shown more OKS errors. Figure 11 shows that img000028 and img001718 are almost at a 4-m distance from the camera and a few predicted yellow key points are at some distance from the ground truth key points in red. In the second type, the background of the image is again plain black but the target is at a far distance from the camera that is within range of almost 14–25 m. In this range, almost all the key points are detected accurately as depicted in img001159 and img014902 of Fig. 11. In the third type, the background of the target satellite has earth and clouds for both near and far distances from the camera. The output behavior is almost similar to the black background. The cases in which the target's body is in the dark it is slightly difficult for the camera to accurately detect the key points i.e., the target in img007574 is under a dark area resulting in low accuracy of key points detection.

The last part of the architecture is the computation of pose values and pose errors. During the reconstruction of the 3D model, all 13 key points are indexed to know the location of each 3D point. Also, the output of the transformer model is labeled for all 13 2D key points. This indexing for both 3D and 2D points helps to quickly establish the correspondence between them without running the iterative process. The 13 predicted 2D keypoints are not available for all the images because sometimes a few 2D points cannot be detected due to contrast and light conditions or sometimes the target satellite is moving in such an orientation that few of the 2D keypoints will be invisible. So, the number of key points predicted by the HRFORMER has the confidence score that depicts how accurate is the specific key point. So, in our case, the threshold of 70% is set for confidence scores. The 2D key points that have a confidence score greater than 0.7 are kept for pose estimation along with their index and the remaining key points are discarded. If all the key points are fed to the pose estimation part then it will increase the pose error. At this stage, the index of 2D and 3D key points are matched and as a result, those 3D points are removed whose corresponding index is not detected in 2D key points. Then use camera matrix K, 3D model points, and 2D image points to minimize the error using the EPnP method. The EPnP algorithm outputs 3 Euler angles and 3 position values in the x, y, and z axis. These six values are then fed to the non-linear least

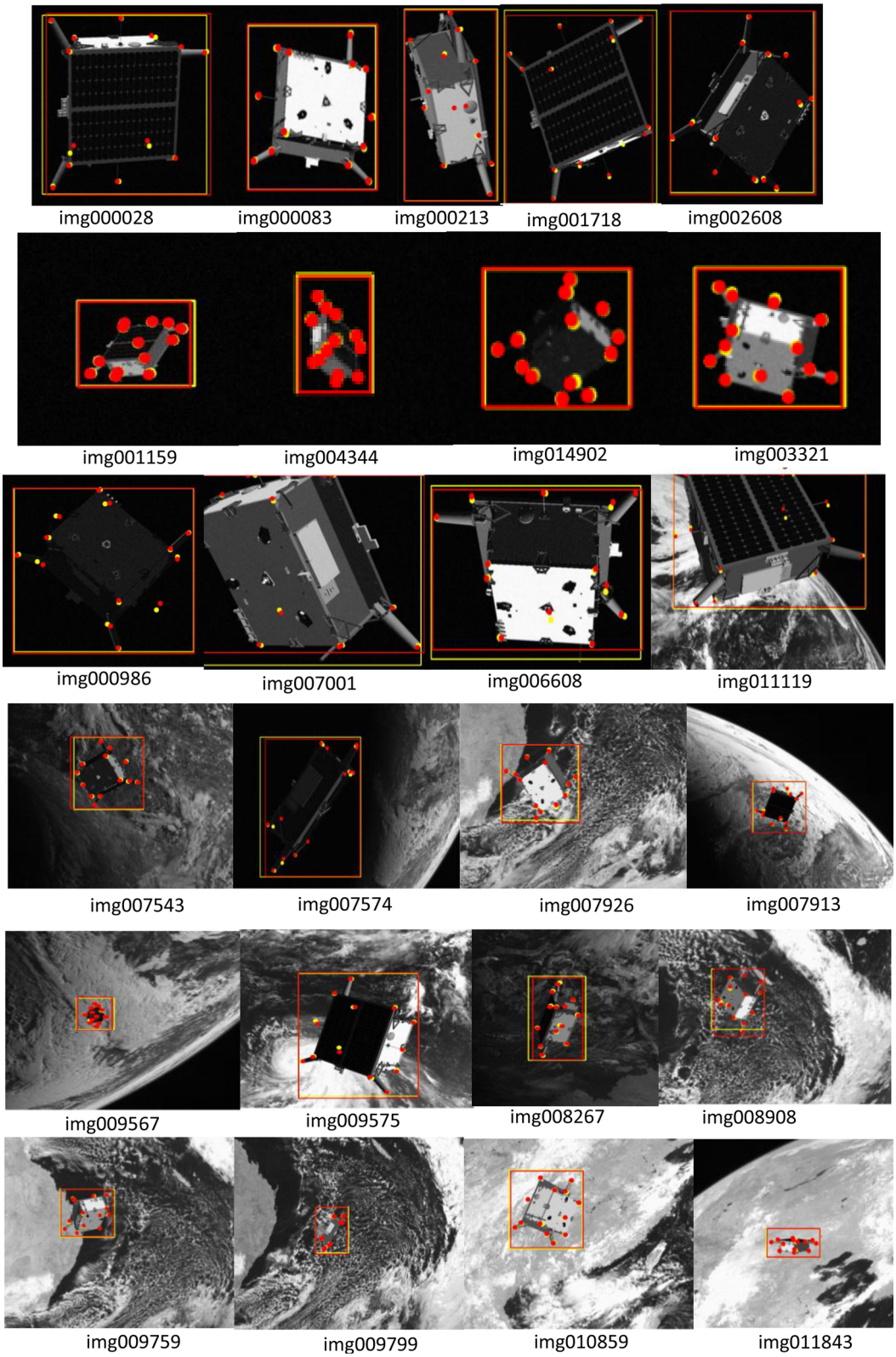


Fig. 11 Output images of key points detection network. The red color indicates ground truth values and yellow indicates predicted values

Table 5 Pose error metrics

%	Metrics	Unit	PnP + GNM	PnP
Translation error	Mean E_t	cm	6.48	10
	Median E_t	cm	3.78	6.62
	Norm. mean e_t	cm	0.84	1.24
	Mean $\left\|t^p_{C/B} - t^g_{C/B}\right\|$	cm	[0.54,0.51,6.32]	[0.78,0.75,9.53]
	Median $\left\|t^p_{C/B} - t^g_{C/B}\right\|$	cm	[0.39,0.40,3.61]	[0.61,0.64,6.55]
Rotation error	Mean E_q	°	1.52	2.22
	Mean $\left\|\theta^p_{xyz} - \theta^g_{xyz}\right\|$	°	[1.11,0.98,0.61]	[1.7,1.4,1]
	Median $\left\|\theta^p_{xyz} - \theta^g_{xyz}\right\|$	°	[0.89,0.78,0.43]	[1.37,1.07,0.75]
	Mean pose error e_T		0.0349	0.0525

square iterative Gauss–Newton method (GNM). The iterative process further refines the 6 values and ends up with the least pose error. The output of pose values using EPnP and a combination of EPnP plus GNM is shown in Table 5. The mean, median, and normalized values for translation and rotation are computed first then the mean pose error is calculated at the end. The maximum mean position error is along the z-axis and it may be because the camera is directly pointed to the target in this direction. The mean position error is 6.4 cm while the mean rotation error in terms of quaternions is 1.52 degrees. The mean rotation error in terms of Euler angles is maximum along the x-axis which is 1.11 degrees.

The error distribution of rotation and translation motion over the test dataset is shown in Fig. 12. The histogram plots show that error is mostly distributed around the mean value and only a few outliers are present. The standard deviation values are plotted that show the fluctuation of error regarding mean value over the entire dataset. The error distribution graph gives information about the frequency of each error when only the EPnP pose solver is used for 3D-2D reprojection. The reprojection error is further improved by adding a non-linear pose solver Gauss–Newton method (GNM) on top of the EPnP solver. The GNM iteratively reduces the error as shown by histograms in Fig. 13 and numerical values in Table 5. The reprojection error is almost reduced to half as compared to the EPnP solver. The final output images of our architecture along with key points and pose values are plotted in Fig. 14. The subset of images from the dataset are randomly chosen for different environments and their predicted translation and rotation values along with their errors are formulated in Table 6. It is depicted that if the target body is between the range of 4–5 m from the target then the prediction of key points is relatively less accurate in comparison when the distance is 6–30 m. The same pattern is observed for pose errors as well. The reason for having less accuracy in the near distance is maybe because the field of view of the camera is at its minimal value.

In conclusion, the feature points predicted by the key points detection network are projected onto the 3D model after 3D-2D linear correspondence between the features. The linear correspondence is only possible because the position of 2D key points is known due to the use of learning-based architecture. The reprojection error is reduced by using EPnP solver and then pose values are further refined by using the GNM method. The pose values, error metrics, and standard deviation for the dataset are formulated to validate the observations.

7 Comparison

The results of the learning-based bounded box and key points detection subsystems are compared in Table 7. Our architecture is thoroughly tested on standard evaluation metrics for precision, recall, IOU, and OKS. The results from related methods are not available for all evaluation metrics. The mean IOU and median IOU for our bounded box subsystem are 95 and 96% respectively which is almost similar to the work presented by Chen et al. [15] and Massimo et al. [16]. However, the size of the input image is variable for all architectures and it is observed that image resolution has a strong impact on learning-based detection architectures. Chen et al. [15] image size is 768 × 768, Massimo et al. [16] worked with 416 × 416 image resolution and our image size is 256 × 192. The high image resolution helps in getting better accuracy and high IOU values. In our case, the same IOU value is achieved using a smaller image resolution, and that resulted in less utilization of memory and FLOPs. Our IOU values are better than the work presented by SPN [22] and Stanford_lab [30].

The results of key points detection are also compared with the related works. Only Massimo et al. [16] discussed the precision of key points and his AP50:95 is 98.97%. The AP50:95 of our architecture is 94% which is less than the

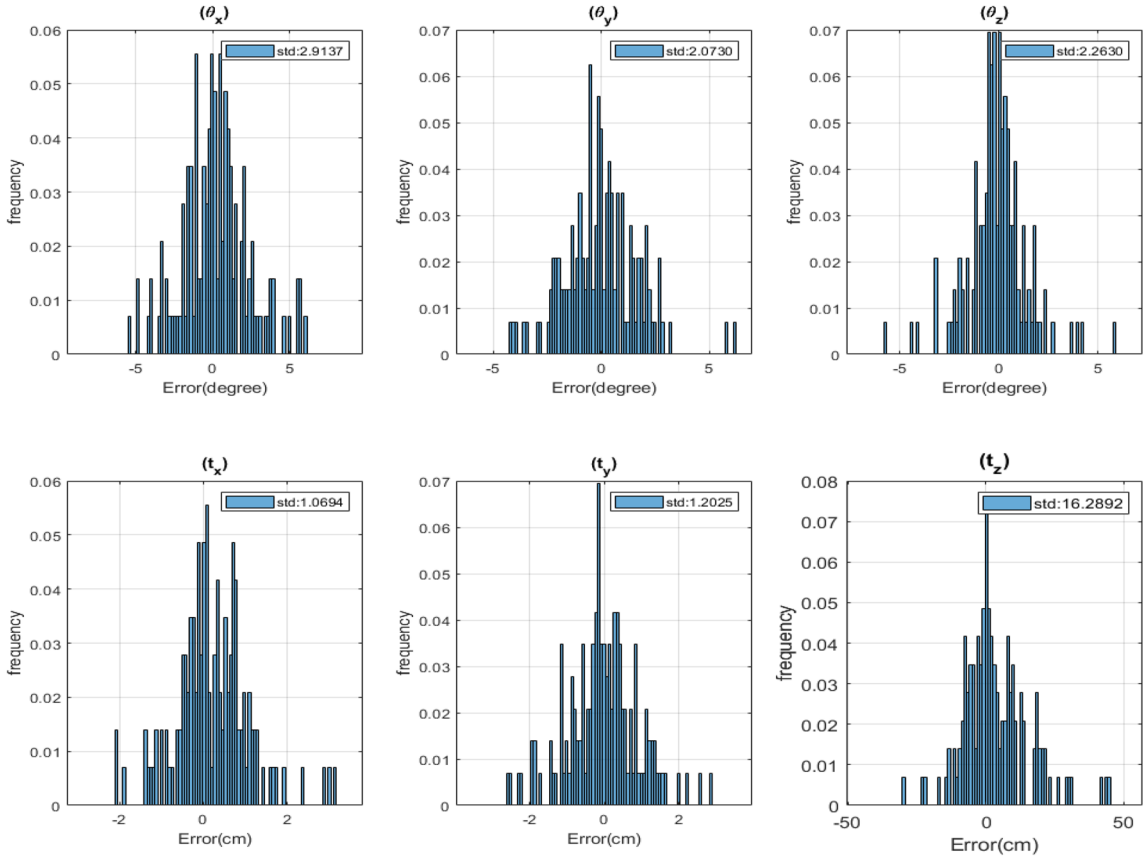


Fig. 12 Error distribution graphs of pose errors using EPnP method

Massimo but the mean OKS of our architecture is 98.5%. The OKS value tells how much the detected key points are close to the ground truth values and due to the high OKS value our pose error is less than Massimo et al. [16]. No other author has discussed the results of their accuracy. The resource utilization of the key points detection architectures is also compared. Our architecture is based on the transformer that uses localized parallel self-attention windows in the multiresolution branches at different stages of the network along with depth-wise convolution in feed forward network layers. The information exchange between layers is only possible through convolution exchange blocks that isolate the transformer from other layers and that is the reason it uses almost half Flops as compared to the CNN-based HRNET architecture used by [15] and [16]. Also, the trainable parameters are reduced by four in our architecture i.e., 28.53–7.68 M So, the same or better accuracy is achieved in our work by using a smaller number of parameters and less memory.

The final pose values are also compared in Table 8 with the available related works in the literature. Our work has performed better than [13, 16, 31] in translation and rotational errors. Our mean translation error is 6.48 cm which is less than [16]. Also, the mean translation value is calculated separately for each axis and it outperformed [16] and [31]. A similar pattern is followed in comparing mean rotation error results both in terms of quaternions and Euler angles. The results are also compared in terms of median values to give a better idea of the errors. Chen et al. [15] mean rotation and translation errors are less than our work because they were participants of the KELVIN POSE estimation challenge and their focus is to perform well in the challenge. So, they have used a higher image resolution of 768 × 768 and a simulated annealing pose solver to reduce the pose error. This high-resolution image and computationally complex pose solver is not feasible for space-grade applications. In comparison, our work is based on a smaller image resolution of 256 × 192 and uses a first-order GNM pose solver that is more feasible for space applications. Wang et al. [17] proposed a transformer architecture based on the DETR transformer. DETR was developed for object detection tasks and performed well on COCO datasets but it is again very hungry for resources. Wang et al. [17] work is good for simulation but it would not be feasible for space application. Our pose error is 0.0349 as compared to Wang's 0.0123 and Chen's 0.0094 but with more practical application for space missions.

The standard deviation of the translation and rotation error is computed and compared with the only available work of Massimo et al. [16]. Our work has significant improvement

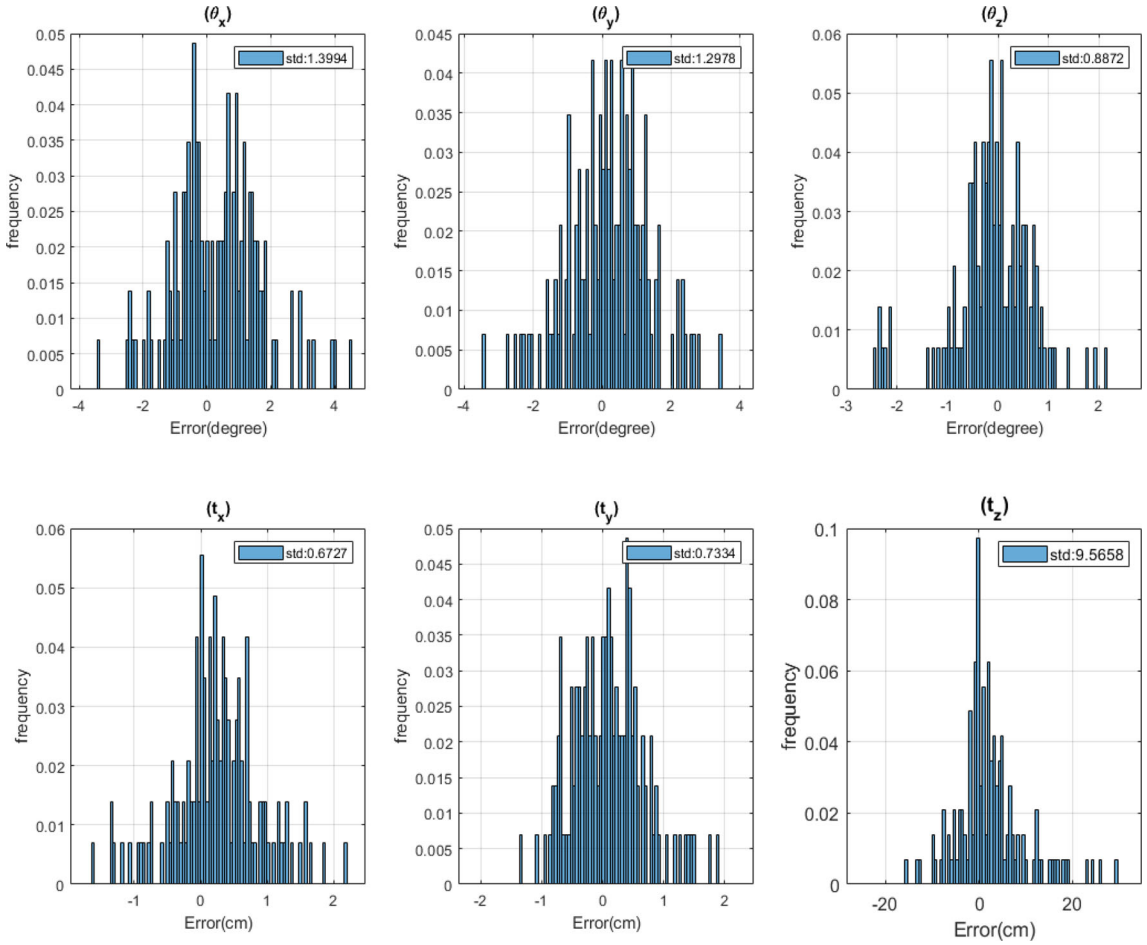


Fig. 13 Error distribution graphs of pose errors using EPnP + GNM method

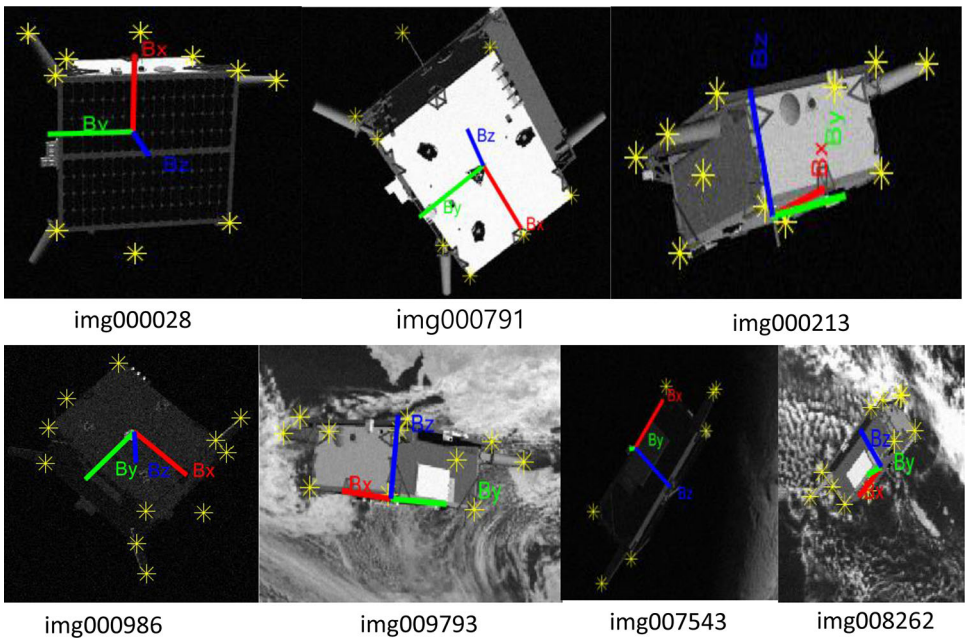


Fig. 14 The sample images from the test dataset show the output key points and axis directions. These are images after the final pose estimation. The pose values are shown in Table 6

Table 6 Pose values and pose errors of the images shown in Fig. 14

Images	Distance (m)	Distance error (m)	Attitude (degree)	Attitude error (degree)
img000028	7.15, − 3.55, 467.53	0.55, 0.19, 0.49	170, 16, 89	2.0, 1.17, 0.05
img000213	− 7.50, − 4.25, 428.30	0.40, 0.74, 7.01	− 81, − 56, − 68	0.80, 1.98, 0.72
img000791	− 5.55, − 4.08, 313.71	0.46, 0.55, 0.52	− 3.36, 30.27, − 54.5	0.43, 1.64, 0.25
img000986	7.2, − 39.4, 492.3	0.28, 0.24, 6.23	16.05, − 17.38, 40.50	2.25, 0.09, 0.29
img007543	− 20.9, 0.29, 391.3	0.89, 0.52, 9.13	93.6, − 3.29, 52.0	0.97, 1.01, 0.64
img008262	7.4, 19.3, 82.2	0.80, 0.12, 7.60	78.89, 22.80, 136.64	0.13, 0.47, 0.004
img009793	− 1.63, − 5.8, 524.0	0.91, 0.27, 4.6	90.1, − 50.51, 173.5	1.84, 1.03, 0.68

Table 7 Bounded box and key points detection evaluation metrics

	Metrics	Unit	Ours	Massimo et al. [16]	Stanford_lab [30]	Chen et al. [15]	SPN [22]
Bounded box	AP50	%	98.8	–	–	–	–
	AP75	%	97.9	–	–	–	–
	AP50–95	%	93	–	–	–	–
	Mean IOU	%	95	95.38	91.9	95.34	85.82
	Median IOU	%	95	96.50	93.6	96.34	89.08
Key points	AP50	%	98	99.74	–	–	–
	AP75	%	98	99.53	–	–	–
	AP50–95	%	94.2	98.97	–	–	–
	Mean OKS	%	98.5	–	–	–	–
	Median OKS	%	96	–	–	–	–
Resource utilization	Image size	pixel	256 × 192	416 × 416	–	768 × 768	224 × 224
	Flops	Giga	3.33	7.68	–	7.68	–
	Parameters	M	7.73	28.53	–	60	–

in comparison to [16]. The standard deviation for all six pose values in our case is much smaller as compared to [16] which shows our method of pose estimation is performing better on all types of images in the dataset. We have also compared our learning-based work with the conventional image-processing method [13]. The results show that 100% solution is provided by learning architectures in comparison to only 20% in conventional methods with very little pose error.

In our work, the image-processing part is based on the learning-based method that provides an inherited advantage of the one–one linear mapping in the pose solver block as shown in Fig. 15. Learning-based key points detection method provides both index and confidence scores of each key point. However, it is not possible to know the position and confidence score in a conventional method that may result in large iterative search space with no reliability of convergence on exact 3D points. The work proposed by Sharma et al. [13] uses a conventional key points detection method that can only provide 20% of the solution due to this issue.

8 Conclusion

The deep learning-based pose estimation pipeline for non-cooperative spacecraft is successfully implemented using a monocular vision sensor. The proposed work is based on the high-resolution transformer architecture for key points detection and according to my knowledge it is the first

Table 8 Comparison of pose error metrics with the related works available in the literature

Metrics	Unit	Our	Massimo et al. [16]	Wang et al. [17]	Chen et al. [15]	SPN [31]	Sharma et al. [13]
Mean E_t	cm	6.48	10.36	3.91	3.5	–	51
Median E_t	cm	3.78	3.58	2.19	1.5	–	–
Norm. mean e_t	cm	0.84	–	–	–	–	–
Mean $\left\|t^p_{C/B} - t^g_{C/B}\right\|$	cm	0.54, 0.51, 6.32	0.52, 0.56, 10.25	0.3, 0.3, 0.4	0.4, 0.4, 3.46	5.5, 4.6, 7.8	14, 6, 51
Median $\left\|t^p_{C/B} - t^g_{C/B}\right\|$	cm	0.39, 0.40, 3.61	0.24, 0.27, 3.50	0.18, 0.16, 2.1	0.31, 0.30, 1.34	2.4, 2.1, 49.6	–
Mean E_q	°	1.52	2.24	0.66	0.73	8.4	2.76
Mean $\left\|\theta^p_{xyz} - \theta^g_{xyz}\right\|$	°	1.11, 0.98, 0.61	1.57, 0.84, 1.72	–	–	–	– 0.6, 0.6, – 1.4
Median $\left\|\theta^p_{xyz} - \theta^g_{xyz}\right\|$	°	0.89, 0.78, 0.43	0.52, 0.33, 0.34	–	–	–	–
Mean e_T		0.0349	0.04627	0.0123	0.0094	0.0626	–
Image size	pixel	256 × 192	416 × 416	336 × 336	768 × 768	224 × 224	–
Solution available	%	100	100	100	100	100	20
σ_{E_T}	cm	0.7, 0.7, 9.6	1.62, 1.71, 30.44	–	–	–	–
σ_{E_θ}	°	1.4, 1.3, 0.9	8.92, 5.11, 10.82	–	–	–	–

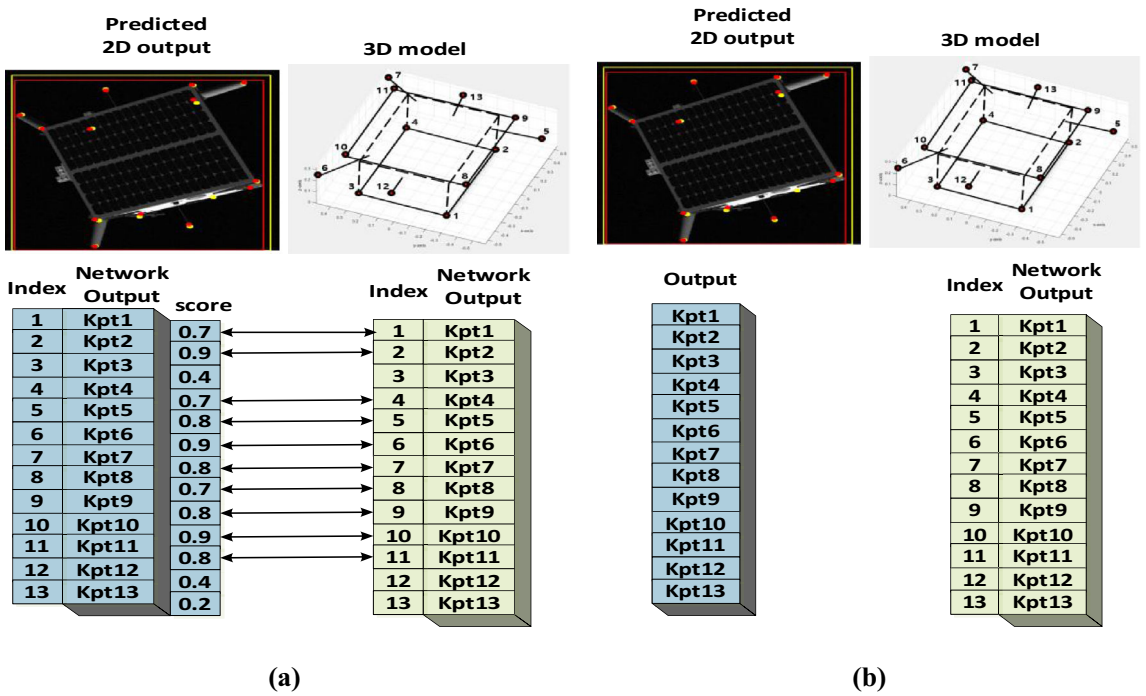


Fig. 15 3D-2D one–one linear mapping of key points **a** learning-based based keypoints network output **b** conventional method key points output

time implemented for the space application. The results have shown that it provides comparable accuracy by using less memory and computation resources on low-resolution images. The pose estimation results have also shown a significant improvement in terms of translation and rotation errors. There are few scenarios in the testing where a satellite seems invisible to the naked eye but it is easily detected by our architecture. The results achieved in this work will help in taking a step further to solve the challenges in this area of research. The use of learning-based architecture for image-processing work has played a significant role in achieving those results. It not only reduces the 3D–2D correspondence time but also significantly improves the accuracy and availability of the solution.

The proposed work is based on images from a single domain dataset however it is not possible to get real space environment images that can be used for training deep learning-based models. In this scenario, there is a requirement to train the model based on datasets that belong to different domains so that the model can learn domain invariant features. This problem is identified as a domain gap in which one dataset belongs to synthetic images or source domain and the other dataset belongs to real images or target domain. In this work, the transformer-based approach is preferred over the CNN to solve this issue to some extent but there is no significant change in results. However, in future work, the domain gap issue can be addressed through different approaches: (1) different data augmentation techniques can be applied to the source domain dataset for generalized model training (2) use of adversarial training either online or offline on source and target datasets so that the features from target domain are projected onto the source domain that will result in domain invariant-based learning (3) use of multimodal training in which model can perform multiple tasks at one time.

Acknowledgements This research was supported by the Space Challenge Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT, the Republic of Korea (2022M1A3B807318012).

Funding Open Access funding enabled and organized by KAIST.

References

1. Ventura J (2016) Autonomous proximity operations for noncooperative space target. Technische Universität München
2. Ming Y, Meng X, Fan C, Hui Yu (2021) Deep learning for monocular depth estimation: a review. Neurocomputing 438:14–33. https://doi.org/10.1016/j.neucom.2020.12.089
3. Xu J, Song B, Yang X, Nan X (2020) An improved deep keypoint detection network for space targets pose estimation. Remote Sens (Basel) 12:3857. https://doi.org/10.3390/rs12233857
4. Barad K (2020) Robust navigation framework for proximity operations around uncooperative spacecraft: a monocular vision-based navigation approach using deep learning. Delft University of Technology
5. Starek JA, AB, NIAD, P (2016) Spacecraft autonomy challenges for next generation space missions. In: Lecture notes in control and information sciences. Springer, Berlin, p 148
6. Telaar J, Ahrns I, Estable S, Rackl W, De Stefano M, Lampariello R, Gil-Fernandez J (2017) GNC architecture for the e. deorbit mission. In: 7th European conference for aeronautics and space sciences (EUCASS). ESA Publications Division, ESTEC, Noordwijk, pp 1–15
7. Pauly L, Rharbaoui W, Shneider C, Rathinam A, Gaudillière V, Aouada D (2023) A survey on deep learning-based monocular spacecraft pose estimation: current state, limitations and prospects. Acta Astronaut. https://doi.org/10.1016/j.actaastro.2023.08.001
8. Opromolla R, Fasano G, Rufino G, Grassi M (2017) A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations. Prog Aerosp Sci 93:53–72. https://doi.org/10.1016/j.paerosci.2017.07.001
9. Sharma S, D'Amico S (2016) Comparative assessment of techniques for initial pose estimation using monocular vision. Acta Astronaut 123:435–445. https://doi.org/10.1016/j.actaastro.2015.12.032
10. Pasqualetto Cassinis L, Fonod R, Gill E (2019) Review of the robustness and applicability of monocular pose estimation systems for relative navigation with an uncooperative spacecraft. Prog Aerosp Sci 110:100548. https://doi.org/10.1016/j.paerosci.2019.05.008
11. Song J, Rondao D, Aouf N (2022) Deep learning-based spacecraft relative navigation methods: a survey. Acta Astronaut 191:22–40. https://doi.org/10.1016/j.actaastro.2021.10.025
12. Han H, Kim H, Bang H (2022) Monocular pose estimation of an uncooperative spacecraft using convexity defect features. Sensors 22:8541. https://doi.org/10.3390/s22218541
13. Sharma S, Ventura J, D'Amico S (2018) Robust model-based monocular pose initialization for noncooperative spacecraft rendezvous. J Spacecr Rockets 55:1414–1429. https://doi.org/10.2514/1.A34124
14. Lepetit V, Moreno-Noguer F, Fua P (2009) EPnP: An accurate O(n) solution to the PnP problem. Int J Comput Vis 81:155–166. https://doi.org/10.1007/s11263-008-0152-6
15. Chen B, Cao J, Parra A, Chin TJ (2019) Satellite pose estimation with deep landmark regression and nonlinear pose refinement. In: Proceedings of the IEEE/CVF international conference on computer vision workshops
16. Piazza M, Maestrini M, Di Lizia P (2022) Monocular relative pose estimation pipeline for uncooperative resident space objects. J Aerosp Inf Syst 19:613–632. https://doi.org/10.2514/1.I011064/ASSET/IMAGES/LARGE/FIGURE25.JPEG
17. Wang Z, Zhang Z, Sun X, Li Z, Yu Q (2022) Revisiting monocular satellite pose estimation with transformer. IEEE Trans Aerosp Electron Syst 58:4279–4294. https://doi.org/10.1109/TAES.2022.3161605

18. Sharma S (2019) Pose estimation of uncooperative spacecraft using monocular vision and deep learning (order no. 28113348). Available from ProQuest Dissertations & Theses Global. (2467859008). https://www.proquest.com/dissertations-theses/pose-estimation-uncooperative-spacecraft-using/docview/2467859008/se-2
19. Garcia A, Musallam MA, Gaudilliere V, Ghorbel E, Al Ismaeil K, Perez M, Aouada D (2021) Lspnet: a 2d localization-oriented spacecraft pose estimation neural network. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2048–2056
20. 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 Astronaut 196:123–138. https://doi.org/10.1016/j.actaastro.2022.04.002
21. Jawaid M, Elms E, Latif Y, Chin TJ (2023) Towards bridging the space domain gap for satellite pose estimation using event sensing. In: 2023 IEEE international conference on robotics and automation (ICRA). IEEE, pp 11866–11873
22. Kisantal M, Sharma S, Park TH, Izzo D, Martens M, D'Amico S (2020) Satellite pose estimation challenge: dataset, competition design, and results. IEEE Trans Aerosp Electron Syst 56:4083–4098. https://doi.org/10.1109/TAES.2020.2989063
23. Park TH, Martens M, Lecuyer G, Izzo D, D'Amico S (2022) SPEED+: next-generation dataset for spacecraft pose estimation across domain gap. In: 2022 IEEE aerospace conference (AERO). IEEE, pp 1–15
24. William Hoff. http://inside.mines.edu/~whoff/course. In: Course CSCI 512
25. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems, p 28
26. Sun K, Xiao B, Liu D, Wang J (2019) Deep high-resolution representation learning for human pose estimation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5693–5703
27. Yuan Y, Fu R, Huang L, Lin W, Zhang C, Chen X, Wang J (2021) HRFormer: high-resolution transformer for dense prediction. In: Advances in neural information processing systems, NeurIPS 2021, arXiv:2110.09408v3. https://doi.org/10.48550/arXiv.2110.09408
28. Pauly L, Rharbaoui W, Shneider C, Rathinam A, Gaudillière V, Aouada D (2023) A survey on deep learning-based monocular spacecraft pose estimation: current state, limitations and prospects. Acta Astronaut 212:339–360. https://doi.org/10.1016/j.actaastro.2023.08.001
29. Park TH, Sharma S, D'Amico S (2019) Towards robust learning-based pose estimation of noncooperative spacecraft. AAS/AAIA astrodynamics specialist conference, Portland, Maine. https://doi.org/10.48550/arXiv.1909.00392
30. Park TH, Märtens M, Jawaid M, Wang Z, Chen B, Chin T-J, Izzo D, D'Amico S (2023) Satellite pose estimation competition 2021: results and analyses. Acta Astronaut 204:640–665. https://doi.org/10.1016/j.actaastro.2023.01.002
31. Sharma S, D'Amico S (2020) Neural network-based pose estimation for noncooperative spacecraft rendezvous. IEEE Trans Aerosp Electron Syst 56:4638–4658. https://doi.org/10.1109/TAES.2020.2999148
32. Liu Z, Ning J, Cao Y, Wei Y, Zhang Z, Lin S, Hu H (2022) Video swin transformer. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3202–3211. https://doi.org/10.48550/arXiv.2106.13230v1
33. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Polosukhin I (2017) Attention is all you need. In: Advances in neural information processing systems, p 30. https://doi.org/10.48550/arXiv.1706.03762