S.I.: AI FOR SPACE: THEORIES, MODELS AND APPLICATIONS

Object detection on space-based optical images leveraging machine learning techniques

Sebastian Samuele Rizzuto[1] · Riccardo Cipollone[1] · Andrea De Vittori[1] · Pierluigi Di Lizia[1] · Mauro Massari[1]

Received: 7 June 2024 / Accepted: 5 February 2025 / Published online: 6 March 2025

Abstract
The expansion of space exploration has led to a soar in the space debris population, increasing the risks of collisions. Addressing this challenge requires advanced space surveillance technologies. Traditional computer vision approaches are unsuitable for real-time applications due to their significant computational demands. Recent progress has been made in ground-based debris detection, thanks to the integration of CNNs. However, to overcome limitations imposed by the atmosphere and other external disturbances, a space-based framework is appealing for spotting fainter objects. This work presents a novel real-time object detection tool designed for space-based applications using machine learning techniques. The absence of labeled datasets for RSOs detection is one of the primary obstacles to AI training, particularly for space-based observations. To tackle this issue, a synthetic '.fits' image generator named SOIG has been developed using photon mapping techniques. The generator produces two types of images. In one instance, it takes into account the sensor's pointing, which follows the satellite's attitude. In the other scenario, the sensor's pointing is considered fixed. Following its training on synthetic images, the subsequent testing phase is conducted through semisynthetic images, which incorporate noise from an actual space-based image. Results demonstrate exceptional performance (mAP50-95 above 90%) for both fixed pointing and rotated and expanded survey pointing images. In the latter case, the tool utilizes sensor attitude information to enhance debris visibility. On the whole, this research wants to contribute to mitigating space collisions and increasing the understanding of machine learning's potential in space debris detection.

Keywords Space-based · Photon mapping · CNNs · Real time · Image generator

1 Introduction

The increasing number of space missions and satellite deployments has led to a major and pressing concern: the rise in space debris. Any human-made object that remains in orbit but is no longer functional is classified as space debris. Examples include fragments of multistage launch vehicles and wreckage from past collisions. These objects pose a growing threat to satellite operations and the long-term sustainability of space activities. In addition, they could initiate a cascade effect, known as the Kessler syndrome [1], a self-perpetuating cycle in which collisions generate more debris, increasing the likelihood of further collisions.

To solve this problem, space surveillance and tracking (SST) is essential. It helps control space pollution, maintain safe space operations, and actively monitor and track objects in orbit. SST involves a range of observations, using optical and radar networks. Optical observations are conducted either directly with sensors on board or on the ground.

Riccardo Cipollone, Andrea De Vittori, Pierluigi Di Lizia, and Mauro Massari have contributed equally to this work.

Accurate detection of space debris using ground-based sensors relies on clear sky conditions and an unobstructed field of view (FoV) at the observatory. Cloud cover, light and particle pollution, or physical barriers can hinder the ability to spot and track debris objects, limiting objects observability.

Ground-based instruments can detect objects with diameters of several centimeters in low earth orbit (LEO) and around two decimeters in geostationary earth orbit (GEO) [2]. To overcome these limitations, space-based techniques can be utilized.

In [3], H. Krag et al. explore, using passive space-based telescopes, the detection of smaller space debris objects. This approach offers advantages such as increased sensitivity and the ability to detect objects at higher altitudes where ground-based systems struggle. The study takes advantage of the passive reflectometry and occultation for orbit fitting (PROOF) tool to assess the influence of sensor orbit and viewing geometry on the types and quantities of objects in the telescope's FoV cone. Furthermore, the utilization of space-based telescopes offers several benefits, as highlighted in [4]:

- *Consecutive observations during 24 h*: space-based telescopes are not affected by weather conditions, atmospheric interference, or day/night cycles, allowing for continuous monitoring of space objects.
- *Complete longitudinal coverage of the geostationary earth orbit belt using a single sensor*: satellite catalogs can be created and maintained with the help of space-based telescopes, which can cover the entire GEO belt.
- *Tracking capability in all orbital regions*: space-based telescopes can track objects in various orbital regions, including LEO, medium earth orbit (MEO), geostationary transfer orbit (GTO), Molniya orbit, and near-earth objects (NEOs).
- *Proximity to LEO small debris for in situ measurement*: space-based telescopes positioned at higher altitudes have the advantage of being closer to LEO small debris, enabling in situ measurements and observations of these objects.
- *No geographical or political restrictions*: unlike ground-based assets, space-based telescopes are not limited by geographical or political constraints. They can be deployed to any location in space, providing unrestricted access to different orbital regions.
- *Scalable performance through the number of spacecraft*: the performance of space-based telescopes can be scaled by increasing the number of spacecraft deployed.

Accurate extraction of information from images often relies on computer vision algorithms. Object detection stands out as a crucial task, which can be done using non-neural or neural network-based techniques [5]. One method falling in the first category is edge detection, which leverages brightness discontinuities to identify object boundaries in images. Neural network-based object detection, on the other hand, uses deep learning methods to recognize objects in images and categorize them into groups, like cars, animals, or people [6].

Neural network-based object detection models can be classified into two categories: two-stage and one-stage detectors. Two-stage detectors break down the object detection task into multiple stages, following a 'coarse-to-fine' approach to progressively narrow down the potential locations of objects. In contrast, one-stage detectors aim to complete the entire process in a single step using a single network. For instance, faster region-based convolutional neural network (R-CNN) represents a prominent example of a two-stage detector, while single shot multibox detector (SSD) and you only look once (YOLO) stand out as popular one-stage detectors.

One-stage detectors like YOLO are faster, making them ideal for real-time analysis of telescope observations. However, they may struggle with tiny objects due to fixed anchor boxes. In contrast, two-stage detectors like Faster R-CNN excel in detecting small objects at the expense of a higher computational burden.

In the context of space object identification, traditional methods such as pixel summation and the Radon transform have been explored [7] [8]. While these techniques achieve good results, they are not suitable for real-time applications. Conversely, as explained in [9], applying edge detection methods like the Canny edge detector, followed by a probabilistic Hough transform, reduces computational cost but yields lower results than the Radon transform.

Another approach involves using thresholding operations to generate binary images, maintaining adequate results while reducing computational load. However, it still requires a classification process, which increases processing time [10] [11].

Other techniques emphasize the use of image sequences to enhance object detection. For instance, in [12], motion features extracted from multiple sequential images are used to identify candidate regions, which are subsequently classified with a dedicated network. A different approach in [13] employs track-before-detect methods, utilizing a particle filter to estimate the target state based on prior knowledge derived from image sequences. However, these methods rely on multiple images obtained through inertial pointing, which are not always available.

Regarding neural network-based methods, [14] and [15] introduce the incorporation of a prior Hough transform-based block within convolutional neural networks. This approach aims to reduce the amount of training data required by leveraging the information provided by the Hough transform. However, as previously discussed, while this technique can decrease the data dependency during training, it may also result in reduced detection performance.

Recent research, as cited in [16] and [17], turns to machine learning techniques as a means to surpass the limitations of traditional methods in identifying space objects from ground-based telescopes. While both studies follow a similar approach, they differ in the employed neural network architectures: an image segmentation model based on U-Net and an image detection model using YOLOv5. The results of these studies indicate that the image detection network is more suitable for real-time applications.

Another technique, proposed in [18], involves modifying the YOLO architecture to specifically address space object detection. While this approach presents a potential solution, it requires significant effort to efficiently adapt YOLO for this specialized application. Additionally, the method may face challenges in terms of generalizability, potentially limiting its effectiveness in diverse scenarios, such as detecting non-faint objects. Despite the positive outcomes achieved by these studies, there is still ample room to enhance the algorithms performance. One potential avenue for improvement lies in upgrading the data used to train the neural network. Collecting and curating such data sets is challenging, as space object observations are limited and often incomplete. Hence, it is imperative to prioritize enhancing the authenticity of synthetic training data. Additionally, it is important to train the neural network using a broader variety of objects observed under diverse conditions. Moreover, the detection of small debris remains a persistent challenge in this field. As previously mentioned, space-based telescopes are a viable option for tackling this issue. Given the real-time constraints typical of on-orbit sensors and the simplicity of the images, where the task is to identify a simple line, convolutional neural networks (CNNs) could offer significant benefits. In this context, the YOLOv8 architecture emerges as a promising solution.

The main objective of this research is to develop an effective and reliable debris detection system for space-based optical images.

The study encompasses three key components to achieve this goal. Firstly, a synthetic image generator is developed to create a comprehensive dataset suitable for training the convolutional neural network (CNN), addressing the significant challenge posed by the limited availability of space-based labeled datasets. Secondly, the focus shifts to training the YOLOv8-based CNN to accurately detect space debris objects within the optical images. Lastly, the paper aims to rigorously evaluate the performance of the trained neural network, using separate testing data.

The document is organized as follows: The core concepts of the employed reference frames, photon mapping, machine learning, and eventually a focus on the tools are presented in Chapter 2. Chapter 3 discusses the dataset development using space optical image generator (SOIG) for the creation of synthetic space-based optical images via photon mapping. It also comprises the creation of a different testing dataset as well as the post-processing of the images to aid the neural networks. In Chapter 4, the emphasis shifts to training and testing, using YOLOv8 architecture, for both survey and fixed pointing images; results considerations are provided at the end. Finally, conclusions and potential future developments are discussed in Chapter 5.

2 Materials and methods

This chapter provides an overview of the fundamental principles and tools utilized in formulating the approach to achieve the objectives of this work.

2.1 Reference frames

2.1.1 Equatorial reference frame

Measurements of right ascension and declination have the vernal equinox as the reference point. The vernal equinox's exact location at a specific instant must be defined, and typically, the J2000 frame is adopted:

- Right ascension, α, is measured positively to the east in the plane of the equator from the direction of the vernal equinox; its values range from 0° to 360°.
- Declination, δ, is measured northward from the equator to the object's location and has values between 90° and -90°.

2.1.2 Sensor and image reference frame

The sensor reference frame is defined according to the sensor (see Fig. 1a). In this configuration, the x-axis is aligned with the positive direction of the right ascension, and the y-axis is aligned with the positive direction of the declination. In a programming environment, the image reference frame is usually centered on the top-left corner, corresponding to the pixel (0,0) (see Fig. 1b).

2.2 Photon mapping

Photon mapping, an advanced technique for realistic image synthesis, has seen significant development in computer graphics in recent years. The process involves two main passes: photon tracing and image rendering. During the first pass, photons are traced from the light sources into the scene, carrying flux information, and are cached in a data structure called the photon map. In the second pass, the image is rendered by evaluating each pixel's intensity using the stored photon map information and detector properties. Before discussing photon mapping key passes, it is pertinent to acknowledge that this work draws inspiration from a paper centered around

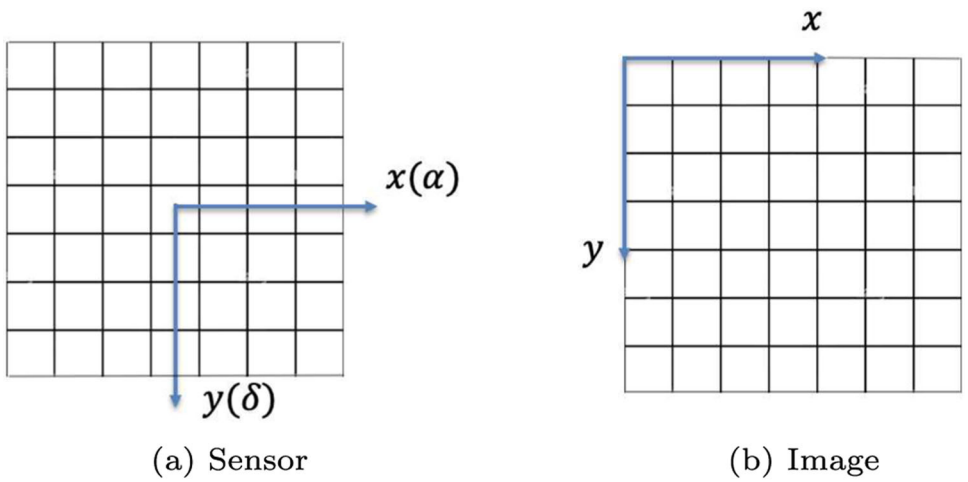


Fig. 1 Sensor and image reference frames.

a ground-based telescope [19]. However, in this study, the telescope is considered to be directly mounted on the satellite itself.

2.2.1 Observation geometry

The observation geometry defines the spatial relationships among the moving target, field stars, and sensor. The relationships among these elements evolve due to factors such as the target's orbital motion and the sensor's movement. The analysis of observational geometry is of utmost importance as it establishes the link between celestial coordinates (α, δ) and image coordinates (x, y). In this study, this transformation is carried out as follows:

$$\begin{aligned} x &= (\alpha - A)\frac{d_x}{r} + d_x \\ y &= (\delta - D)\frac{d_y}{r} + d_y \end{aligned} \tag{1}$$

where α and δ represent the right ascension and declination of the stars and the moving target, A and D are the right ascension and declination of the sensor pointing, r is half of the sensor's FoV, and dx and dy correspond to half of the image size along the x and y coordinates, respectively. During their movement, a sensor is inevitably influenced by external factors such as gravity and thermal loads, which can cause deformations. To model the sensor pointing, a 2D Brownian motion in the right ascension and declination directions is adopted. Consequently, the following equations are utilized:

$$\begin{aligned} A(t) &= A_0(t) + \sigma_A B_A(t) \\ D(t) &= D_0(t) + \sigma_D B_D(t) \end{aligned} \tag{2}$$

where $(A_0(t), D0(t))$ denotes the sensor's pointing direction in the absence of Brownian motion, and $B_A(t)$ and $B_D(t)$ are 1D standard Brownian motion in the right ascension and declination directions, respectively. σ_A and σ_D are the corresponding scaling parameters.

2.2.2 Photon tracing

Photon tracing involves emitting discrete photons from light sources and tracing their paths through the scene. The primary goal is to populate the photon maps used in the rendering. Light sources emit photons that travel through space, interacting with various media—some are absorbed, while others survive. The focus is solely on how the surviving photons are distributed on the image plane. The expected (or average) number of photons arriving at the image plane is given by:

$$N_{\mathrm{pi}} = t_{\mathrm{exp}} 10^{-0.4(m_i + ZP)} \tag{3}$$

where ZP is the zero-point, m_i is the magnitude, t_{exp} is the exposure time, and -0.4 is a fundamental component of the Pogson magnitude system, a logarithmic scale used to quantify stellar brightness. Various parameters, such as the telescope collecting area, the fractional spectral bandwidth of the filter, and source flux density, are commonly used in the literature to estimate the number of photons. However, obtaining these parameters may not always be straightforward. For device-specific simulations, Eq. 3 offers a more convenient approach. When considering the fluctuation in the number of photons from the same source in different images, N_{pi} can be replaced with a random number drawn from a Poisson distribution with mean N_{pi}. This stochastic approach helps account for the inherent variability in photon counts and allows for a more realistic representation of the imaging process.

2.2.3 Image rendering

Image rendering evaluates the value of each pixel using the information stored in the photon map and the parameters of the detector. This process includes the conversion from photons to analog-to-digital units (ADUs). It accounts for various detector effects such as bias and flat-field effects, and the presence of manufacturing defective pixels. Photons from the same source may fall on different pixels, and a single pixel may receive photons from different sources. Additionally, some photons from a source may fall outside the charge-coupled device (CCD) plane. The discrete shape of the CCD pixels allows for recording the photon's approximate position. For optical sensors with a small FoV, the sky background can be considered approximately uniform by adding a constant value E_{back} in photon-electrons to all pixel values. To introduce noise in the background, each E_{back} value should be replaced with a random number of events drawn from a Poisson or Gaussian distribution with mean E_{back}. Alternatively, the background E_{back} can be estimated from a real observed image. The total number of photon-electrons in a pixel $E = E_{source} + E_{back}$ provides a pure photon-electron image without contamination from the readout process, which can be beneficial for certain image processing tasks. However, to include this additional disturbance on a realistic CCD, the entire digitization process can be realized, for a single pixel, by:

$$\mathrm{ADU} = \frac{E}{\mathrm{Gain}}\mathrm{Flat} + \mathrm{Bias} \tag{4}$$

These parameters can be specified arbitrarily or measured using a real device.

2.3 Convolutional neural networks

A CNN is a powerful deep learning algorithm designed to process input images, assigning importance (learnable weights and biases) to different features or objects within the images. CNNs excel at reducing images into a more streamlined format, allowing efficient processing and analysis without sacrificing critical and useful data required for accurate predictions. This capability not only enables faster learning but also makes it possible to leverage massive datasets that would otherwise demand significant computational time. Within the realm of CNNs, YOLOv8 represents the latest state-of-the-art model in the YOLOseries, catering to tasks like object detection, image classification, and instance segmentation. Developed by Ultralytics, the creators of the influential YOLOv5 model, YOLOv8 introduces a range of architectural and developer experience enhancements, setting it apart from its predecessors.

2.3.1 Performance analysis

For object detection tasks, precision and recall are concise and valuable metrics for understanding a network's performance. They are defined as follows:

- Precision: The ability of a model to identify only the relevant objects.
- Recall: The ability of a model to find all the relevant cases.

They are calculated based on the intersection over union (IoU) value for a given threshold. This parameter is computed for each detected object and represents the overlap between the predicted bounding box and the ground truth bounding box. The IoU threshold commonly used is 0.5, which leads to the following classifications:

- IoU > 0.5: Classified as true positive (TP).
- IoU < 0.5: Classified as false positive (FP).
- Missed detections: Classified as false negative (FN).

Adopting the formal definitions of TP, FP, and FN, the precision and recall for a given class across the test set are calculated as follows:

$$\begin{aligned} \text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}} \\ \text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}} \end{aligned} \tag{5}$$

The precision–recall curve is commonly plotted to visualize how recall changes concerning a given precision and vice versa. The average precision (AP) is then calculated by taking the area under the precision–recall curve, and the mean average precision (mAP) is the average of the AP calculated for all the classes.

2.4 Tools

2.4.1 TLE2KERNEL

The two-line element (TLE) format, a widely used text-based standard for storing orbital data, encodes essential information in two lines of 69 characters each. For this application, the NASA SPICE (spacecraft planet instrument C-matrix events) library was utilized to efficiently extract the target's orbital state over time by leveraging SPICE's standard SPK (spacecraft and planet Kernel) files—binary files that store detailed orbital data. To streamline this process, the Department of Aerospace Science and Technology has developed an internal pipeline that processes the TLEs of resident space objects (RSOs), and by specifying a desired time window, the pipeline generates the corresponding SPICE SPK files.

2.4.2 SPIMT

The simulating photometric images of moving targets with photon mapping (SPIMT), originally developed by Dujunju [19], serve as the fundamental backbone for this work. SPIMT employs the photon mapping approach to effectively simulate images. Initially designed for Earth-based telescope simulations, the versatile nature of SPIMT's codes makes it well suited for space-based implementations.

3 Dataset generation (SOIG)

Neural networks, in this work YOLOv8, demand a substantial number of images, typically in the thousands, for effective training. However, the availability of space-based real images is limited. Consequently, substantial efforts are directed toward the development of a Python application dubbed SOIG. This tool is meticulously crafted to generate '.fits' images that faithfully emulate space-based real-world scenarios.

3.1 Debris and observing satellite

In this study, the observing satellite of choice is the SENTINEL 3A. The rationale behind selecting this satellite stems from its orbit, which aligns well with the hypothetical space observation mission. To enhance the diversity of the dataset and effectively consider various crowded orbits, a meticulous selection process is undertaken, resulting in the inclusion of four distinct debris objects (Table 1).

In the context of simulating images, it is pertinent to take into account the concept of effective area. Under the assumption that the radar cross section (RCS) can be equated to the effective area, a modeling approach is

Table 1 Debris orbits characteristics

NORAD CAT ID	Period	Inclination	Apogee	Perigee
37476	121.27 min	98.96°	2619 km	856 km
40386	272.14 min	50.72°	14702 km	364 km
40649	375.48 min	52.78°	20490 km	1235 km
40989	1418.22 min	4.43°	35540 km	35332 km

adopted for all debris. This modeling is founded on a normal distribution, where the mean RCS is fixed at 0.956 m^2, and the standard deviation is set at 2.291 m^2. The sampled area is taken with the absolute value.

Determining the positions of the observing satellite and debris objects, a critical step involves generating SPK kernels. This process is efficiently executed by employing the TLE2KERNEL codes.

To build a diverse scenario dataset, observation windows lasting a few seconds are randomly selected within a broader timeframe. Commencing at UTC 2022-01-01T00:00:00.00 and extending to UTC 2023-01-01T06:00:00.00, this strategic approach ensures the inclusion of diverse moments throughout the designated period.

3.2 SOIG structure

The workflow of the SOIG application comprises a sequence of intricate stages (Fig. 2), culminating in the generation of synthetic 1024 x 1024 images (one without noise) accompanied by their corresponding labels. This intricate process is steered by several key inputs, including:

- Selection and loading of the SPK file of one of the four pre-identified debris objects.
- A dynamic observation period: In this study, it is chosen at random from a range of 2 to 8 s.
- Specification of the initial time, confined within a year-long timeframe as outlined earlier.
- Establishment of the FoV radius, with a consistent value of 0.1° chosen for this work.
- Definition of the maximum magnitude discernible by the sensor.
- Introduction of a Boolean variable named 'attitude,' which, when enabled, offers the capability to take into account the observing satellite's attitude for survey pointing.

The key points of this flowchart will be illustrated below.

3.2.1 Stage 1: Fixed vs survey pointing

When the 'attitude' input variable is set to True, a comprehensive approach is adopted to account for the observing satellite's attitude. In this context, the attitude refers specifically to Nadir Pointing. By closely tracking changes in the satellite's right ascension and declination, the sensor precisely mirrors these fluctuations (Fig. 3a). This approach is referred to as survey pointing. Conversely, when the Boolean variable is set to False, the sensor's pointing remains fixed at the specific right ascension and declination coordinates (Fig. 3b).

Furthermore, the variation in attitude significantly impacts the image generation process. For instance, the choice of the stars' FoV radius to be uploaded is greatly influenced. In the context of fixed pointing, this radius is set to 0.15° to ensure comprehensive coverage of the images, as the image's FoV radius is 0.1°. Conversely, when considering survey pointing, the FoV radius of the stars to be uploaded encompasses the maximum variations in right ascension or declination due to the attitude changes. An extra value is added to guarantee complete coverage of the image.

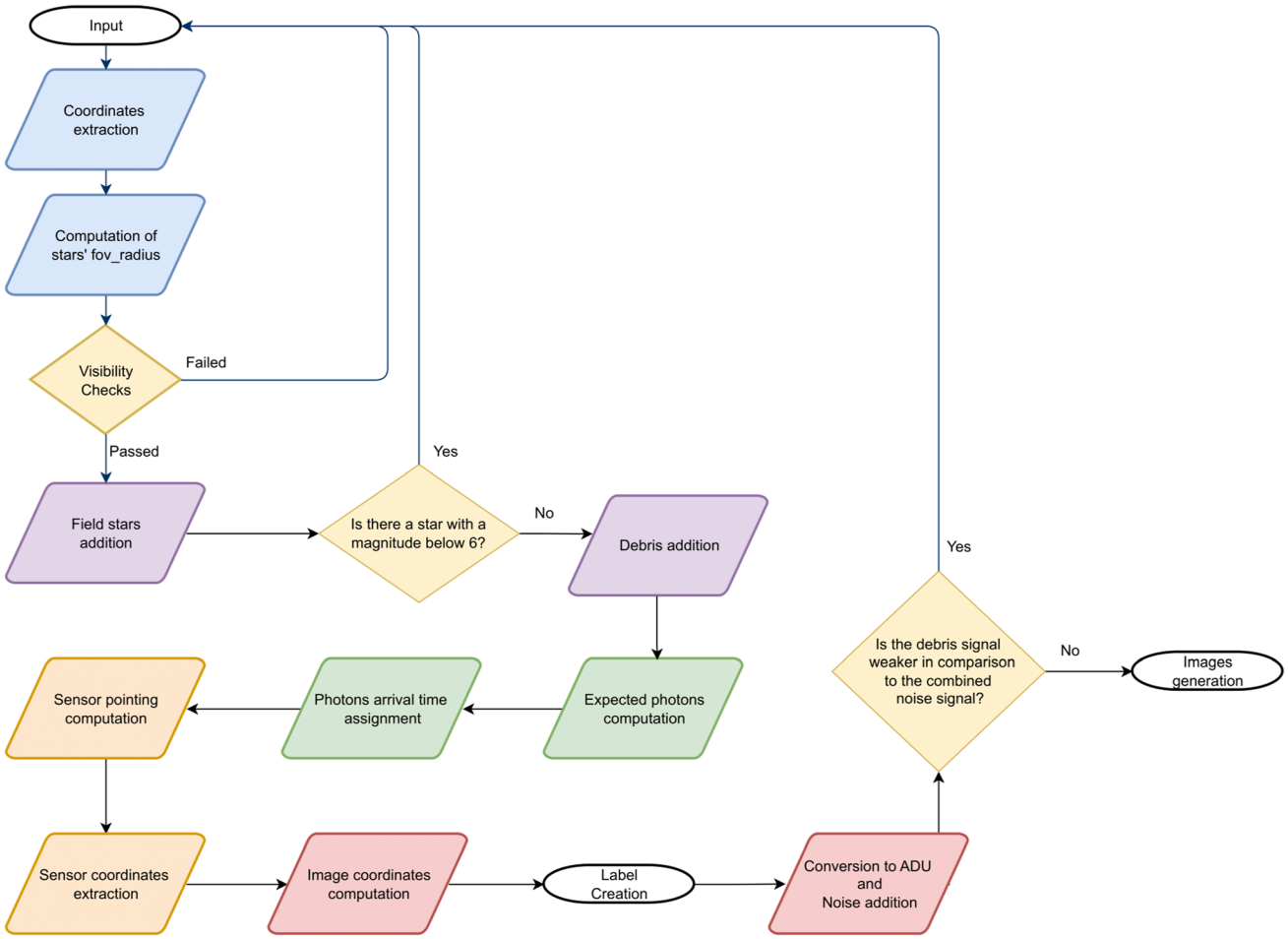
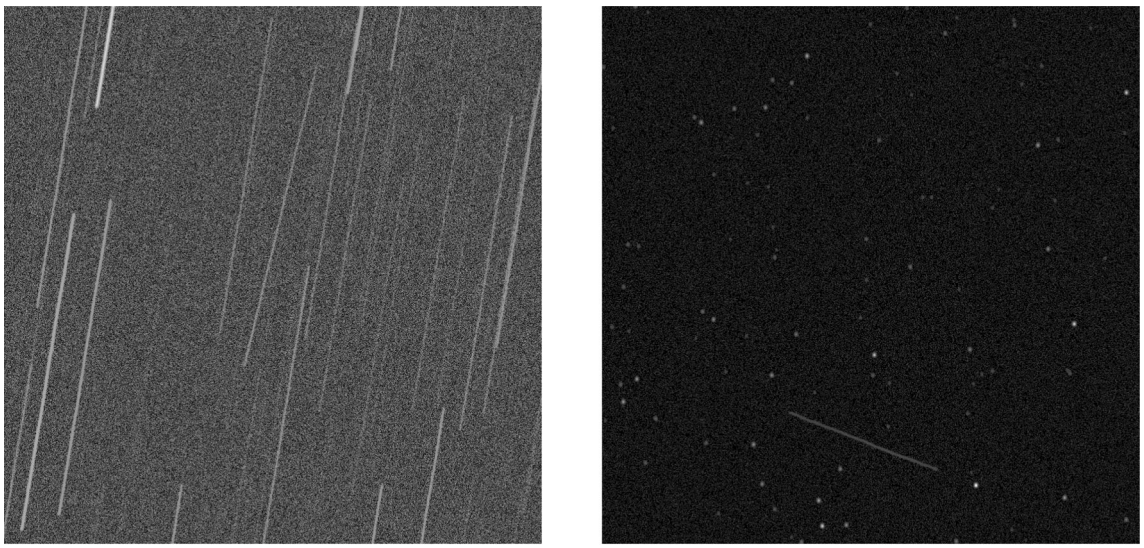


Fig. 2 SOIG structure



(a) Survey (NORAD: 40989) (b) Fixed (NORAD: 40649)

Fig. 3 Comparison between survey and fixed pointing: images generated by SOIG.

3.2.2 Stage 2: Visibility checks

The visibility of debris objects is not guaranteed at all times, as they can be obscured by the intense light emitted by the Sun or Earth and Moon's albedo. To ensure the utmost realism in the image generation process, meticulous consideration is given to the angular separation between the debris pointing direction and the respective positions of the Sun, Earth, and Moon. A minimum separation angle of more than 105° is enforced for the Earth, and a minimum separation angle of 20° and 15° is upheld concerning the Sun and Moon, respectively. Additionally, the sensors can detect objects only when their magnitude is below a specified threshold. For this study, the threshold is set at 16.5, to replicate real-world sensor limitations as discussed in [20]. To maintain the realism of the generated images, a verification of the debris magnitude is incorporated. This check ensures that images are not generated with debris of higher magnitudes. The computation of the debris magnitude is carried out using the following equation:

$$mv = -26.7 - 2.5\log(A_c \rho F(\phi)) + 5.0\log(r_0) \tag{6}$$

where A_c stands for the object's cross-sectional area, ρ represents its reflection coefficient, r_o indicates its distance from the observer (in the same units as A_c), ϕ stands for the object's phase angle, and F is a function dependent on the object's shape and orientation. The value of -26.7 corresponds to the sun's apparent visual magnitude. The phase angle (ϕ) represents the angle between the observer and the Sun, measured from the target's perspective.

The value derived from the equation serves as a reference for rescaling an authentic light curve trend extracted randomly from the database managed by the MiniMegaTORTORA (MMT-9) observatory. Since the chosen debris are not present in the database, a similar debris with regard to orbital regime is selected for each case. This rescaling process significantly enhances the realism of the simulation.

For a diffuse sphere, the function $F(\phi)$ is expressed as:

$$F(\phi) = \frac{2}{3\pi^2}[(\pi - \phi)\cos\phi + \sin\phi] \tag{7}$$

The reflection coefficient ρ is fixed at 1 in this paper, and the cross-sectional area for each image is randomly sampled from a Gaussian distribution, as explained in Chapter 3.1. Debris attitude considerations are omitted.

Only when all the specified checks are passed, does the attention turn toward the initial pointing. In the context of dataset generation, the initial pointing is deliberately selected to be random around the debris. This deliberate choice serves two main purposes: firstly, to prevent the generation of empty images, which hold no utility for the neural network's training phase, and secondly, to accelerate the process of image generation.

3.2.3 Stage 3: Adding field stars and debris to the scene

Following these preparatory stages, the process of image generation is set in motion. A first step entails the integration of all stars situated within the 'FoV radius of stars,' while simultaneously considering the predetermined initial pointing. This integration process involves querying the specified region to retrieve the pertinent information from a designated catalog, utilizing the VizieR web services. For this study, two distinct catalogs are chosen: the initial Gaia source list (IGSL) and the Gaia data early release 3 (Gaia EDR3).

Subsequently, these two selected catalogs are subjected to cross-matching. The process of cross-matching between the catalogs results in the extraction of several critical parameters. These parameters encompass the barycentric right ascension of the source in the International Celestial Reference System (ICRS) at the epoch Ep=2016.0, the barycentric declination of the source in ICRS at the same epoch, the photometric magnitude

(magG), and the distinctive source identifier (ID) linked to each matched source. Given that the reference epoch is set at 2016, it becomes necessary to implement a correction to account for the proper motion:

$$\begin{aligned} RA &= \mathrm{ra} + \frac{\mathrm{pmra} \cdot (t_c - t_{\mathrm{epoch}})}{\mathrm{conversion\ factor} \cdot \cos(dec)} \\ DEC &= \mathrm{dec} + \frac{\mathrm{pmdec} \cdot (t_c - t_{\mathrm{epoch}})}{\mathrm{conversion\ factor}} \end{aligned} \tag{8}$$

The equation takes into account several parameters, including the initial right ascension (ra), the proper motion in right ascension (pmra), the current time (t_c), specifically, the initial time plus half of the observation period, and the reference epoch (t_{epoch}) set to 2016. It also considers a conversion factor that transforms milliarcseconds per year (mas/year) to degrees per year, along with the initial declination (dec) and the proper motion in declination (pmdec).

So, a comprehensive list of stars is readily accessible. This compilation's refinement is bolstered by the meticulous incorporation of a maximum magnitude threshold, tailored to align with the specifications of the employed sensor. Moreover, an enhancement is the introduction of a minimum magnitude criterion, established at a value of 6. This threshold assumes a strategic role, filtering out celestial objects emitting higher luminosity. Importantly, if even a single star registers a magnitude below this established threshold, the generation of the image will be interrupted. These measures assume a pivotal role in preventing the overshadowing influence exerted by these brighter entities. Without such precautions, there could be a risk of diminishing the clarity and discernibility of their fainter counterparts.

Having successfully integrated all available stars, the focus now turns to incorporating space debris. Debris, which exhibits dynamic movement over time, is discretized within the exposure time, resulting in 2000 distinct points. Each of these points is treated as a celestial entity and added to the list as a star. Their magnitudes are calculated meticulously using Eq. 6 followed by subsequent rescaling of the light curves.

Additionally, precise right ascension and declination values are extracted from the SPK kernel, accompanied by corresponding temporal information for each point.

3.2.4 Stage 4: Adding photons to the scene

The first step of the photon tracing is to compute the expected number of photons arriving at the image plane. Recognizing that the computational efficacy of photon mapping experiences degradation when handling an extensive volume of photons, it becomes necessary to calculate the appropriate exposure duration for stars. This calculation is crucial since stars, particularly when operating in a survey mode, are not observable throughout the entirety of the observation period. Conversely, for both the debris points and stars within fixed pointing, the exposure time encompasses the entirety of the observation period. The number of photons is then computed using Eq. 3, with a zero-point (ZP) of 2.74, based on data from the ACS Zeropoints Calculator provided by the Space Telescope Science Institute.

Concerning the debris, the variable N_{pi} generated for each discretization point is proportionally divided by the number of points.

Ultimately, N_{pi} is substituted with a random count of events drawn from a Poisson distribution, where the mean of this distribution remains N_{pi}.

To mitigate computational overhead and prevent memory overload, an approximation is introduced. When the count of surviving photons for a single source exceeds 100,000, every five photons are merged into a single entity. The exact value is then recalculated and assigned to the corresponding pixel during the rendering phase. However, this approach does come with a minor drawback. The generation of the point spread function (PSF) is executed with a reduced count of photons, leading to a decrease in their scattering characteristics.

In the following phase, the assignment of arrival times to the surviving photons takes place. For stars in fixed pointing scenarios, this assignment is carried out via a uniform distribution that spans from 0 to the duration of the observation period. On the other hand, for stars in survey pointing, the arrival times are uniformly distributed within the range of exposure duration. However, in the context of debris, the temporal information corresponding to each discretized point inherently serves as the arrival time itself.

3.2.5 Stage 5: Photons tracing

The subsequent stage involves the tracing of photons in alignment with the sensor's orientation. To begin, let us examine the concept of sensor pointing. Specifically, within the framework of a survey, the process of computing the sensor pointing is elucidated by the steps outlined in Algorithm 1. Here, exposure_value represents the observation duration, while *ra_v* denotes the vector indicating the sensor's right ascension pointing, reflecting changes in the satellite's right ascension. Moreover, *ra_tar* embodies the vector that accurately portrays the pointing, taking into account the arrival time of each photon *t_ij*. Analogously, *dec_v* denotes the vector indicating the declination of the sensor pointing, and *dec_tar* symbolizes the vector representing the pointing, considering the arrival time of each photon.

Algorithm 1 Survey pointing

1: $ra\_tck = \text{interpolate.splrep}(\text{np.linspace}(0, \text{exposure\_value} + 0.1, 1000), ra\_v, s = 0)$
2: $ra\_tar = \text{interpolate.splev}(t\_ij, ra\_tck, \text{der} = 0)$
3: $dec\_tck = \text{interpolate.splrep}(\text{np.linspace}(0, \text{exposure\_value} + 0.1, 1000), dec\_v, s = 0)$
4: $dec\_tar = \text{interpolate.splev}(t\_ij, dec\_tck, \text{der} = 0)$

Conversely, for the fixed pointing scenario, the variables *ra_tar* and *dec_tar* remain constant throughout the entire exposure duration.

For both fixed and survey pointing approaches, an additional parameter is introduced to encompass the effects of external factors on the sensor's behavior. This parameter is characterized by a Brownian motion model, as described by Eq. 2. In this particular study, it is noteworthy that the values assigned to the parameters σ_A and σ_D are both equal to 0.0002. The next step consists of mapping celestial coordinates into the sensor coordinates:

$$\begin{aligned} x_s &= (\alpha - ra\_tar)\frac{\text{dx}}{\text{fov\_radius}} \\ y_s &= (\delta - dec\_tar)\frac{\text{dy}}{\text{fov\_radius}} \end{aligned} \tag{9}$$

In this context, each element of the vector representing the right ascension (α) and declination (δ) of stars and debris is adjusted by subtracting the corresponding sensor pointing value at the same time instance. In the scope of this study, the image dimensions considered are 1024 x 1024, leading to dx and dy values both being set to 512.

An additional Gaussian-distributed component (mean 0, standard deviation 1 arcsecond) is applied to the right ascension and declination of stars and debris to simulate the effects of the point spread function (PSF).

3.2.6 Stage 6: Image rendering

Once the sensor's coordinates have been computed, the subsequent stage involves assessing the value of individual pixels. The initial phase involves transitioning from sensor coordinates to image coordinates. This transition is accomplished by adding the values of dx and dy to x_s and y_s, respectively:

$$x = x_s + dx \; y = y_s + dy \tag{10}$$

Subsequently, the rendering phase is executed in the following manner:

Algorithm 2 Image rendering

```
for x, y in zip(x,y) do
    if -0.5 < x < x_size and -0.5 < y < y_size then
        IMG_PHOTON[int(y), int(x)] = img_photon[int(y), int(x)] + 1
    end if
end for
```

In this context, IMG_PHOTON represents a matrix with dimensions aligned to the image, where each element corresponds to a pixel. For sources subject to the approximation during photons addition within the scene, the value +1 is exchanged for +5. Among the more demanding tasks in object detection is undoubtedly the formulation of labels, a process entailing the pinpointing of regions within an image that encompasses the sought-after elements. This data holds paramount significance during the training phase, enabling the network to internalize the traits of the elements it is intended to recognize.

The YOLO labels format is:

[class ID] [object center in X] [object center in Y] [width in X] [width in Y]

Figure 4 provides an illustrative example of label placement on images.

The rendering phase encompasses the conversion to ADU, while also addressing multiple detector influences including background noise, bias, and flat-field effects. To derive the analog-to-digital units for each pixel from the IMG_PHOTON matrix, division by the gain is essential. In this study, the gain is calculated using data from an actual CCD sensor (TELEDYNE CCD47-20), resulting in a value of 1.6. In scenarios where optical sensors

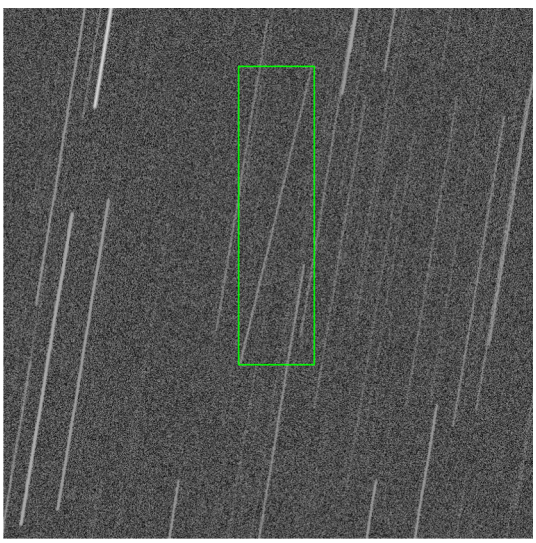


Fig. 4 Label representation (indicated in green): image generated by SOIG (NORAD: 40989).

have constrained FoVs, it is common to approximate the sky background as uniformly distributed. In this study, the background noise matrix is generated by applying a uniform distribution to each pixel. Specifically, this distribution ranges from 190 to 210 ADUs for an exposure time of 5 s. It is worth noting that these values change linearly with different exposure times. Conversely, the flat-field effect is not accounted for, and the bias effect is modeled as a uniform distribution across each pixel spanning from 2 to 3 ADUs. To calculate the ADU for each pixel, the subsequent equation is utilized:

$$IMG\_ADU(x,y) = \frac{\text{img\_photon}\ (x,y)}{\text{gain}} + \text{background}\ (x,y) + \text{bias}\ (x,y) \tag{11}$$

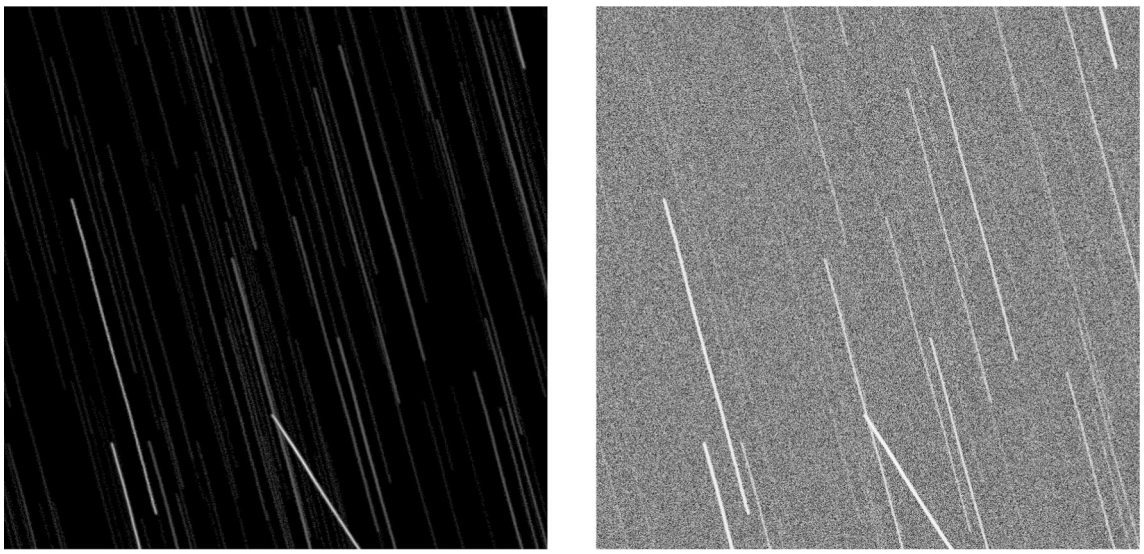
Before initiating the image generation, an additional verification step is incorporated. This step revolves around evaluating whether the debris signal is notably weaker in comparison with the combined noise signal. If the ratio of the debris signal to the overall noise signal drops below a predetermined threshold, the image generation is stopped to prevent creating images where debris is obscured by excessive noise. The specific threshold value is established at 0.035 for images with survey pointing and at 0.15 for images with fixed pointing. Finally, two distinct images are generated: one without noise. The noise-free image is produced by leveraging the IMG_PHOTON matrix (Fig. 5a). Conversely, the second image is generated using the IMG_ADU matrix (Fig. 5b).

3.3 Image processing

3.3.1 Fits to Png conversion

The SOIG-generated images initially come in '.fits' format, which must be converted to '.png' format to meet YOLOv8's requirements. This conversion process aims to enhance the visibility of moving objects and subsequently reduces the images to 8-bit color depth. The foundational code for this process, 'FITStoPNG.py,' was originally developed by Jason Calvi [17] and serves as the fundamental framework, with adaptations made within this work. The transformations include:

1. Square root compression (for fixed pointing images only)



(a) Noise-free (b) Noisy

Fig. 5 Comparison between fixed and survey pointing: images generated by SOIG (NORAD: 40649)

2. Logarithmic correction
3. CLAHE (contrast-limited adaptive histogram equalization)
4. Performs-global-contrast (PGC)

An example illustrating how these transformations enhance the visibility of the image is depicted in Fig. 6.

3.3.2 Square root compression

The square root compression involves the application of the square root to every pixel in the images, resulting in a compression of higher pixel values. In the context of this study, this compression technique is exclusively employed on fixed pointing images, which are more susceptible to the influence of elevated pixel values.

3.3.3 Inverse logarithmic correction

With the inverse logarithmic correction, all pixels are elevated to higher values, consequently widening the gap between higher and lower value pixels, thereby enhancing the overall contrast. The degree of enhancement is determined by the precise value of gain within the inverse logarithmic correction, carefully chosen for each image to avoid pixel saturation.

3.3.4 CLAHE

CLAHE is an algorithm designed to enhance local contrast by analyzing histograms within various tile regions of an image. This approach allows for the enhancement of details, even in areas that may be inherently darker or lighter than others.

3.3.5 Performs gamma correction

The PGC transformation, alternatively referred to as the power law transform, operates on the input image at a pixel level, following the equation:

$$O = I^{\gamma} \tag{12}$$

(a) Image without visibility enhancements

(b) Image further enhanced with Square Root Compression and Logarithmic Correction

(c) Image further enhanced with CLAHE and Gamma Correction

Fig. 6 Visibility enhancement process: images generated by SOIG (NORAD: 40989)

During this process, each pixel undergoes rescaling to ensure that its values are confined within the range of 0 to 1. Notably, the chosen value for γ in this study is set at 0.7.

3.3.6 Image rotation

Considering the satellite's built-in optical sensor, the inherent information it possesses emerges as a valuable asset that can significantly enhance the performance of neural networks in object detection. Specifically, for survey pointing images, the variations in both the sensor's right ascension and declination pointing yield crucial insights. By discerning and analyzing these variations, it becomes feasible to extrapolate the angle formed between the luminous trails left by stars and the x-axis orientation of the captured image:

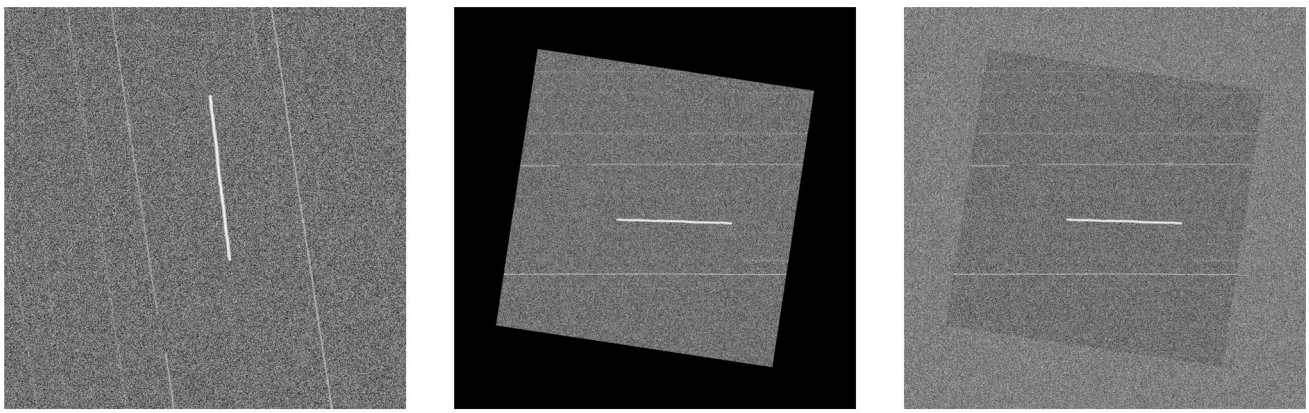
$$\text{angle} = \arctan\left(\frac{\text{declination\_var}}{\text{right\_ascension\_var}}\right) \tag{13}$$

where declination_var and right_ascension_var denote variations in pointing, specifically in declination and right ascension, respectively. As a result, the image can be effectively rotated by this computed angle, aligning the stars horizontally and thereby augmenting the clarity of the debris (Fig. 7). However, to prevent loss of information, the rotated images should be expanded. This presents a challenge regarding their suitability for neural network training, as the process introduces black regions that impede usability. To address this limitation blank regions are filled with a noise pattern, wherein the noise's mean and standard deviation match those of the original images (refer to Fig. 7c). This expanding procedure effectively increases the image dimensions from 1024 x 1024 to 1472 x 1472 pixels.

Following this, a critical step involves adjusting the corresponding labels to align with the rotated and expanded image. This label replacement procedure is seamlessly integrated within the generator, a strategic decision due to its requirement of a comprehensive understanding of debris positions.

3.4 Semisynthetic images generation

To thoroughly assess the trained neural network, particularly due to the unavailability of authentic space-based images, a separate set of images is crafted for both fixed and survey pointing scenarios. These new images are also generated through the utilization of the SOIG tool, albeit with specific variations. Notably, a primary distinction lies in the choice of the observing satellite. In this context, the SKYMED4 satellite is adopted, introducing a new facet to the experimental process.



(a) Initial image (b) Rotated and expanded image (c) Image after noise filling

Fig. 7 Image rotation process: images generated by SOIG (NORAD: 37476)

Another notable differentiation lies in the background noise of the image derived from an actual space-based image, coming from the Canadian Astronomy Data Center and sourced from NEOSSAT (Fig. 8), adding a layer of authenticity to the generated images.

However, the noise matrix from NEOSSAT images, based on a 48-second observation, must be normalized by this duration and then scaled to the observation period of the semisynthetic image. Finally, let us examine a comparison between the synthetic and semisynthetic images (Fig. 9) to better understand and appreciate the distinctions between the two. In Fig. 9b, it is evident that the noise exhibits a non-uniform pattern, with a black band within the image. This phenomenon arises due to the adaptation of the real image to conform to the dimensions of the synthetic images, specifically 1024×1024.

4 Neural network implementation and analysis

4.1 Data preparation and detection analysis

In both fixed and survey pointing scenarios, a different dataset consisting of 4000 noisy images along with their corresponding labels is employed. To streamline processing, these datasets are organized into three separate sub-folders: one designated for training, another for validation, and a third for testing. This division maintains a distribution ratio of 70% for training, 20% for validation, and 10% for testing.

To ensure efficient implementation, the input files are further grouped into two additional sub-folders: one for 'images' and another for 'labels.' The precise arrangement of images and their corresponding labels within these folders hold significant importance, as YOLOv8 automatically pairs these elements based on the folder structure.

Lastly, to facilitate this organizational framework, a YAML file is generated. This file outlines directory paths and establishes class names for bounding boxes, streamlining the process further.

4.2 Fixed pointing

4.2.1 Training and validation

Training a neural network typically demands significant hardware resources, particularly as the model's complexity and the size of the training dataset increase. The decision to use the YOLOv8 light version in this context

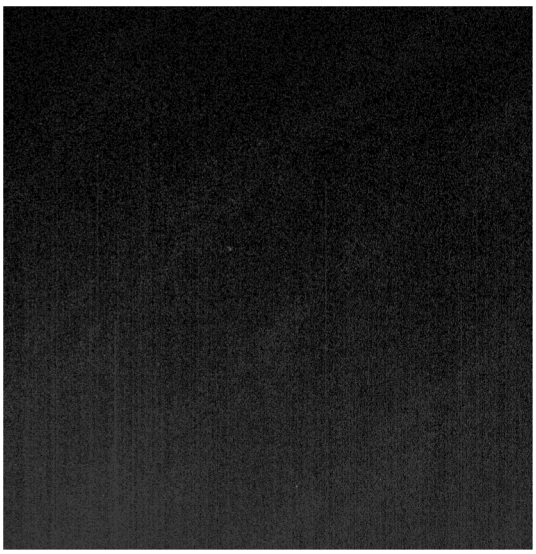
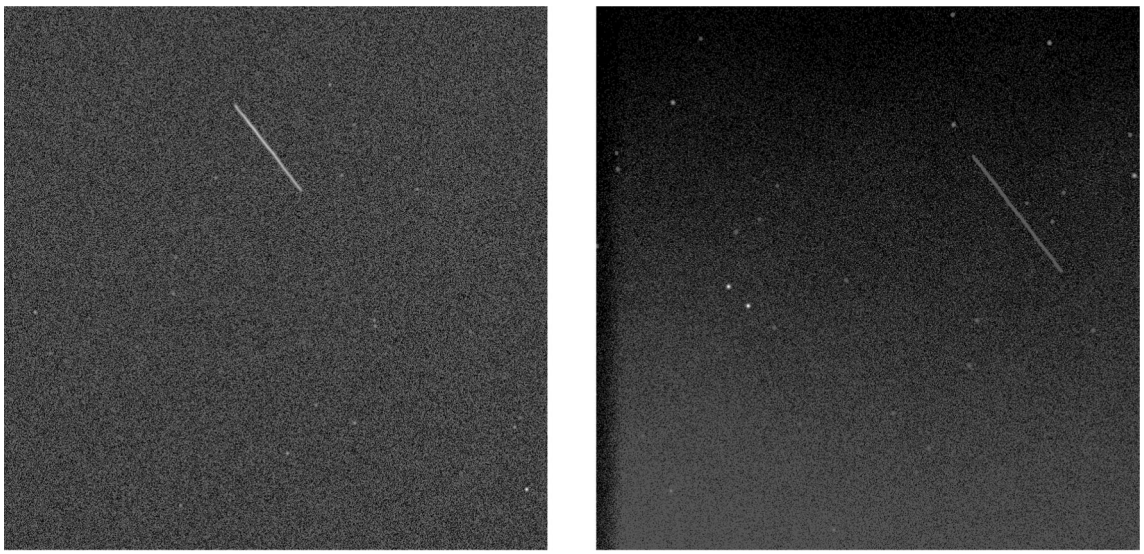


Fig. 8 Real image taken on board the NEOSSAT satellite



(a) Synthetic (b) Semi-Synthetic

Fig. 9 Comparison between synthetic and semisynthetic images: images generated by SOIG (NORAD: 40649)

is strategic, considering the limitations of graphics processing unit (GPU) memory. The training command incorporates the following parameters:

- model: Specifies the utilized model, in this case, it is set to 'yolov8n.pt.'
- data: Defines the path to the '.yaml' configuration file, crafted in the preceding step.
- epochs: Determines the number of training epochs, set to 200 for this particular scenario.
- batch: Sets the batch size as 16 images per batch.
- imgsz: Establishes the dimensions of input images.
- optimizer: Selects the optimizer to employ, with options including stochastic gradient descent (SGD), Adam, Adamax, AdamW, NAdam, RAdam, RMSProp, and auto. Here, it is set to 'auto.'
- val: Determines whether validation occurs during training, configured as True.

In this study, two distinct training approaches utilizing the same dataset are employed. The first involves using the original 1024 x 1024 images, while the second expands the images to 1088 x 1088 dimensions. This expansion process emulates a form of padding without modifications to the original YOLO code. During this process, the empty parts of the images are filled with noise modeled as a Gaussian distribution. The mean and standard deviation of this noise match those of the original image. By considering the optimal weights, the validation results presented in Table 2 are obtained.

Considering the validation set, the graphs in Fig. 10 illustrate precision over recall, where the curve's area underneath signifies the average precision (AP) for distinct classes (in this case, a single class). The ideal trajectory of this curve starts from a point where precision is 1 and recall is 0 at the top-left, extending toward a point where precision is 0 and recall is 1 at the bottom-right.

In conclusion, both curves follow the desired ideal behavior, indicating the successful completion of the training phase.

Table 2 Validation results

	Precision	Recall	mAP50	mAP50-95
Original	0.996	0.99	0.995	0.934
Padded	0.996	0.993	0.994	0.932

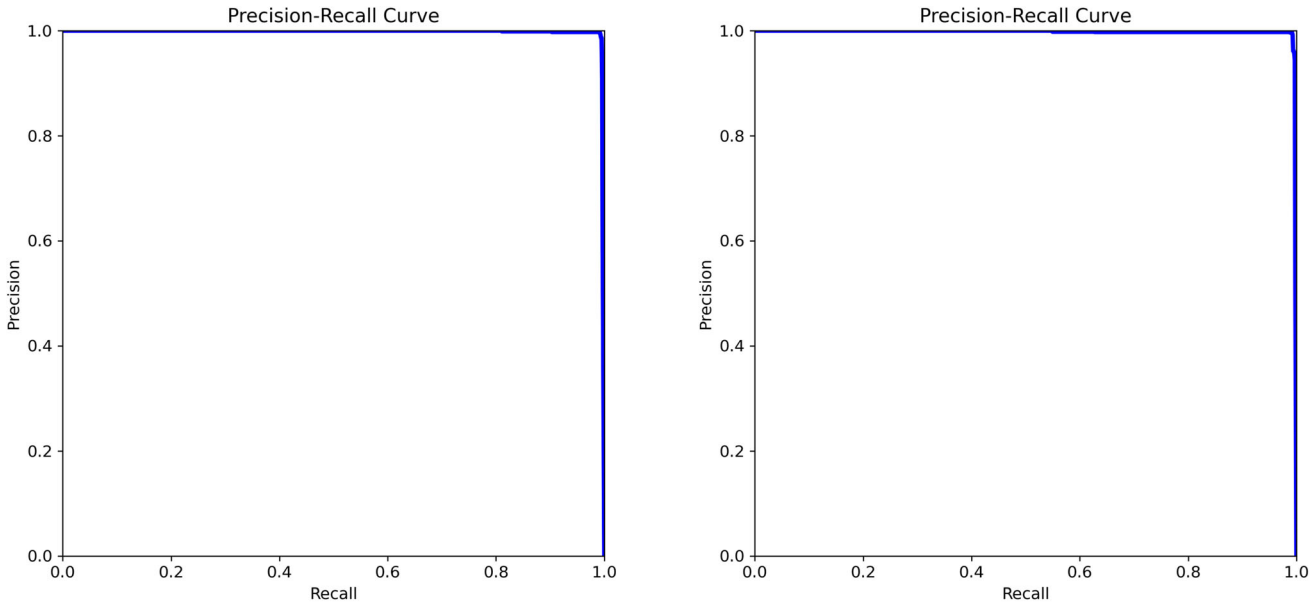


Fig. 10 Precision over recall

Table 3 Testing results

	Precision	Recall	mAP50	mAP50-95
Original (synthetic)	1	0.988	0.993	0.933
Padded (synthetic)	1	0.99	0.995	0.941
Original (semisynthetic)	0.995	0.985	0.992	0.945
Padded (semisynthetic)	1	0.988	0.993	0.946

4.2.2 Testing

The testing phase processes the remaining 10% of the dataset. Testing can be done directly on YOLO. To calculate the mAP, precision, and recall values of all classes the procedure requires input images with their labels so that the network can detect them and compare the input bounding boxes, with the generated ones. These results are attained by configuring YOLO with an IoU threshold of 0.5 and a confidence level of 0.5. Notably, these settings are chosen while utilizing the best weights obtained from the preceding training phase. Furthermore, to ensure a comprehensive assessment of the model's reliability and efficacy, an additional round of testing entails 400 semisynthetic images. The results from both synthetic and semisynthetic tests are displayed in Table 3.

In Fig. 11, clear examples of true-positive predictions are showcased. Particularly, in the context of padded images, it becomes apparent how effectively the bounding box encompasses the debris object better than the original one. This demonstrates how a form of padding can enhance the neural network's performance.

Despite slight performance losses in some cases, the overall results confirm the neural network's reliability and generalization capabilities in the fixed pointing scenario.

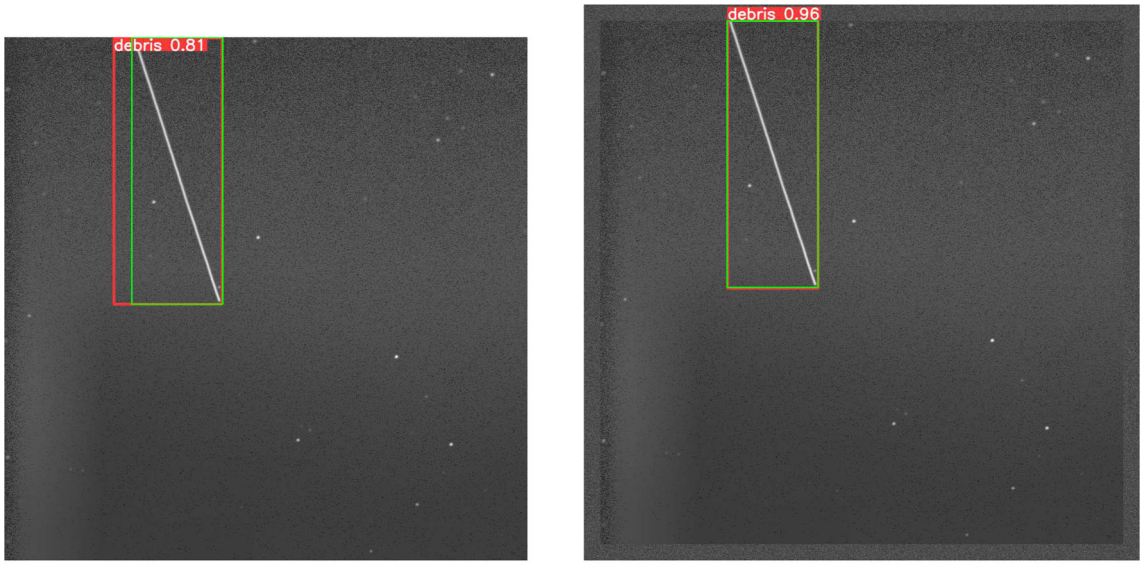


Fig. 11 True positive: correct labels are highlighted in green, while labels predicted by YOLO are marked in red (NORAD: 37476).

4.3 Survey pointing

4.3.1 Training and validation

Similar to the methodology adopted in the fixed pointing scenario, two distinct training approaches utilizing the same dataset are employed. The first training approach involves using the original 1024 x 1024 images. In contrast, the second approach involves taking the same images but rotating and expanding them to 1472 x 1472 dimensions. This rotation and expansion leverage satellite attitude information to enhance the neural network's performance.

The training inputs remain consistent with those employed in the fixed pointing scenario, with the only modification occurring when training involves rotated and enlarged images. In this case, the batch size is set to 10, due to limitations on the GPU.

Utilizing the optimal weights, the validation results are presented in Table 4.

Notably, each performance parameter exhibits improvements in the rotated and expanded image case, a fact that becomes even more evident in the subsequent graphical representations (Fig. 12), representing the precision over recall curves.

4.4 Testing

Similarly, in the survey pointing scenario, two distinct testing phases are conducted, comprising 400 synthetic and semisynthetic images. The corresponding results are summarized in Table 5.

In conclusion, a noticeable decline in performance is observed when handling non-rotated and enlarged images. Throughout the development of this work, attempts were made to enlarge the images without applying

Table 4 Validation results

	Precision	Recall	mAP50	mAP50-95
Original	0.922	0.895	0.948	0.887
Rotated and Enlarged	0.986	0.949	0.98	0.917

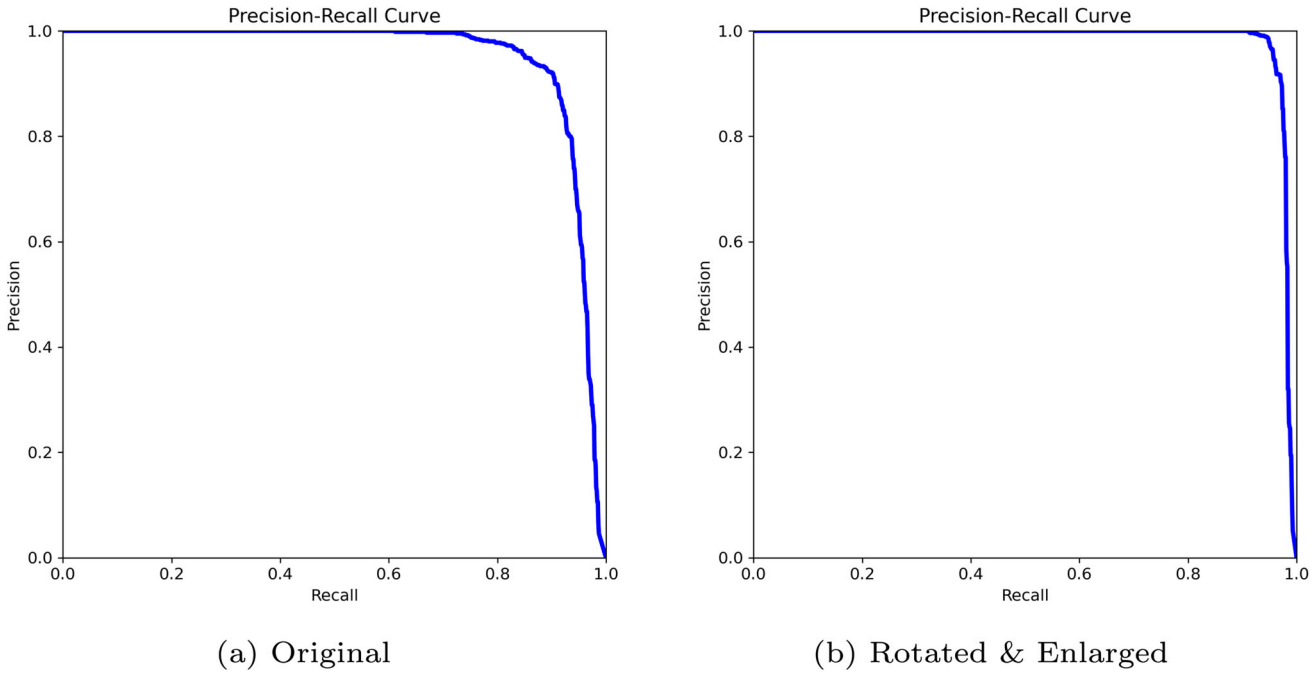


Fig. 12 Precision over recall.

Table 5 Testing results

	Precision	Recall	mAP50	mAP50-95
Original (synthetic)	0.921	0.899	0.94	0.907
Rotated and Enlarged (synthetic)	0.989	0.929	0.978	0.934
Original (semisynthetic)	0.847	0.856	0.894	0.853
Rotated and Enlarged (semisynthetic)	0.987	0.938	0.972	0.95

any rotation; however, the results remained consistent with those of the original case and thus are not presented. Figure 13 confirms that false negatives are frequently given for the original photographs rather than the rotated and enlarged ones.

4.5 Results considerations

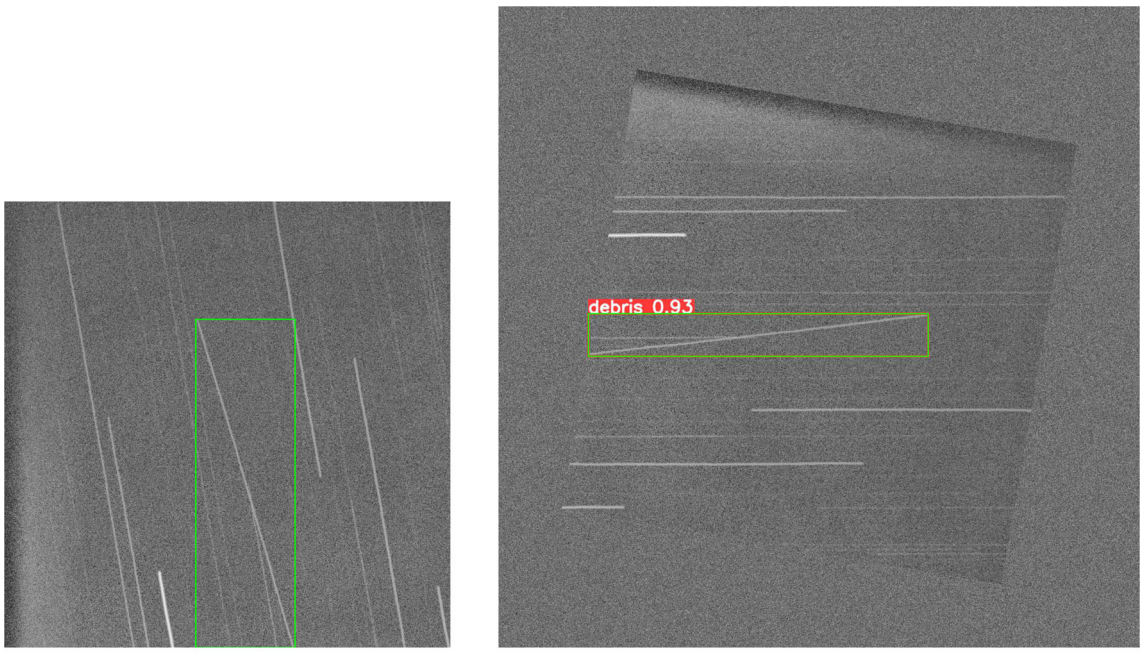
The performance of the neural network consistently showcased its superiority when dealing with fixed pointing images, particularly in comparison with its performance with survey images across a range of tests. Within the survey pointing scenario, the neural network excels the most when presented with rotated and enlarged images, displaying notably better performance compared to its outcomes with original images.

The objective of the work is to present a strategy that balances speed and accuracy effectively in tracklet identification. This approach is designed to facilitate the rapid scheduling of follow-up observations.

The temporal aspects of the algorithm's execution can be dissected into two key processes:

1. Conversion of images, potentially involving rotation and expansion, from .fits format to .png images.
2. Detection phase.

Efforts to minimize the processing time for the first process involve the implementation of a fast and efficient observation processing algorithm. However, achieving this efficiency is primarily the responsibility of the programmer, who must devise an algorithm that minimizes downtime and identifies the most suitable image transformations. In the context of the second, their enhancement is inherently constrained by hardware



(a) Original (b) Rotated & Enlarged

Fig. 13 False negative: correct labels are highlighted in green, while labels predicted by YOLO are marked in red (NORAD: 40386).

Table 6 Processes time required

	Conversion	Detection
Original	0.29s	30ms
Padded	0.30s	30ms
Rotated and Enlarged	0.42s	30 ms

capabilities. The quantification of timing for the detection phases is conducted using an Nvidia GeForce RTX 2060 6GB. Meanwhile, the image conversion process is executed on a local computer equipped with a 13th Gen Intel(R) Core(TM) i7-1370P. Table 6 presents the time values for both original, padded, and rotated and enlarged scenarios.

A significant portion of the time consumption is attributed to the image conversion process. However, the conversion time is notably shorter for original images, and it can likely be attributed to the fact that the image size is reduced in the case of original images, leading to faster saving times for the images in the '.png' format.

The detection time demonstrates remarkable efficiency, although results may vary with different GPUs. The reported time of 30 ms encompasses the entire process conducted on YOLO, including preprocessing, inference, and post-processing stages. Notably, the timing performances remain consistent across both fixed and survey pointing scenarios, as well as between original, padded, and rotated and enlarged images.

In conclusion, the algorithm demonstrates its suitability for real-time applications, given its consistent results across various scenarios.

5 Conclusion

Space situational awareness remains a pivotal focus within the space industry, and its significance will only grow in the years to come. Central to these endeavors is the detection and cataloging of space debris, which remains at the core of every operational undertaking.

Over recent years, ground-based techniques for object detection have experienced remarkable advancement, largely due to the integration of CNNs. However, there is a compelling need to shift attention toward a space-based approach to overcome the constraints inherent in terrestrial methodologies. This work leads to the development of a suitably authentic SOIG generator, accompanied by a suite of neural networks designed for detecting debris in fixed pointing images and survey pointing rotated and expanded images. These networks exhibit a robust proficiency in RSO detection, maintaining a balance between accuracy and efficiency, all while delivering suitable inference times for real-time applications. However, there is room for improvement and several potential enhancements warrant consideration:

- Incorporating real spaceborne sensor images during both training and testing phases could significantly enhance the neural network's performance and validate its real-world applicability.
- Improving the authenticity of synthetic images can enhance the realism of the dataset, thereby facilitating more effective training for real-world debris detection.
- Optimizing YOLO training commands and architecture could further improve performance parameters.

In addition to the previously mentioned improvements, future research directions may involve further optimizing the neural network by harnessing sensor attitude information more effectively or exploring alternative networks such as FAST R-CNN or SSD, which may enhance performance in the survey pointing scenario. Furthermore, the next steps in research could encompass precise astrometric reduction for each debris, culminating in the estimation of the centroid's angular coordinates.

Funding Open access funding provided by Politecnico di Milano within the CRUI-CARE Agreement.

Data availability The processed datasets are not publicly accessible, but they can be made available upon request under reasonable conditions.

Declarations

Conflict of interest The authors have no conflict of interest to declare that are relevant to the content of this article.

References

1. ESA Space Debris Office (2023) ESA's Annual Space Environment Report. https://www.sdo.esoc.esa.int/environment_report/Space_Environment_Report_latest.pdf
2. Flohrer T, Peltonen J, Kramer A, Eronen T, Kuusela J, Riihonen E, Schildknecht T, Stöveken E, Valtonen E, Wokke F, Flury W (2005) Space-Based Optical Observations of Space Debris. In: Paper presented at the 4th European Conference on Space Debris

3. Krag H, Kahl M, Bendisch J, Klinkrad H, Schildknecht T (2001) Space based optical observation of small debris objects. In: Paper presented at the 3rd European Conference on Space Debris
4. Utzmann J, Wagner A (2016) SBSS Demonstrator : A Space-Based Telescope for Space Surveillance and Tracking. NATO Public Release
5. O'Mahony N, Campbell S, Carvalho A, Harapanahalli S, Velasco Hernandez G, Krpalkova L, Riordan L, Walsh J (2019) Deep Learning vs. Traditional Computer Vision. In: Paper presented at the 2019 Computer Vision Conference
6. Zou Z, Chen K, Shi Z, Guo Y, Ye J (2023) Object detection in 20 years: a survey. Proceed IEEE 111(3):257–276. https://doi.org/10.1109/JPROC.2023.3238524
7. Milani A, Villani A, Stiavelli M (1996) Discovery of very small asteroids by automated trail detection. Earth Moon Planet 72:257–262. https://doi.org/10.1007/BF00117526
8. Nir G, Zackay B, Ofek E (2018) Optimal and efficient streak detection in astronomical images. The Astronom J. https://doi.org/10.3847/1538-3881/aaddff
9. Martín FIS, Perez CA, Tapia JE, Virani S, Holzinger MJ (2020) Automatic space object detection on all-sky images from a synoptic survey synthetic telescope array. Acta Astronautica 65(1):337–350. https://doi.org/10.1016/j.asr.2019.09.037
10. Virtanen J, Poikonen J, Säntti T, Komulainen T, Torppa J, Granvik M, Muinonen K, Pentikäinen H, Martikainen J, Jyri Näränen JL, Flohrer T (2016) Streak detection and analysis pipeline for space-debris optical images. Adv Space Res 57(8):1607–1623. https://doi.org/10.1016/j.asr.2015.09.024
11. Diprima F, Santoni F, Piergentili F, Fortunato V, Abbattista C, Amoruso L (2018) Efficient and automatic image reduction framework for space debris detection based on gpu technology. Acta Astronautica 145:332–341. https://doi.org/10.1016/j.actaastro.2018.02.009
12. Xi J, Xiang Y, Ersoy OK, Cong M, Wei X, Gu J (2020) Space debris detection using feature learning of candidate regions in optical image sequences. IEEE Access 8:150864–150877. https://doi.org/10.1109/ACCESS.2020.3016761
13. Uetsuhara M, Ikoma N (2014) Faint Debris Detection by Particle Based Track-Before-Detect Method. In: Paper presented at the Advanced Maui Optical and Space Surveillance Technologies Conference
14. Rachith E, Hellmich S, Irureta-Goyena BY (2023) A Novel Machine Learning Based Algorithm for Efficient Streak Detection. In: Paper presented at the 2nd International Orbital Debris Conference
15. Lin Y, Pintea SL, Gemert JC (2020) Deep hough-transform line priors. CoRR https://doi.org/10.48550/arXiv.2007.09493
16. De Vittori A, Cipollone R, Di Lizia P, Massari M (2022) Real-time space object tracklet extraction from telescope survey images with machine learning. Astrodynamics 6(3):205–218. https://doi.org/10.1007/s42064-022-0134-4
17. Calvi J, De Vittori A, Cipollone R, Di Lizia P, Panico A (2021) Machine Learning Techniques for Detection and Tracking of Space Objects in Optical Telescope Images. In: Paper presented at the Aerospace Europe Conference, Warsaw, Poland
18. Jiang Y, Tang Y, Ying C (2023) Finding a needle in a haystack: faint and small space object detection in 16-bit astronomical images using a deep learning-based approach. Electronics. https://doi.org/10.3390/electronics12234820
19. Du J, Hu S, Chen X, Cao H, Jiang Y (2021) Simulating photometric images of moving targets with photon-mapping. The Astronom Soc Pacific 133(1028):104501. https://doi.org/10.1088/1538-3873/ac240f
20. Ackermann MR, Cox DD, Kiziah RR, Zimmer PC, McGraw JT, Cox DD (2015) A systematic examination of ground-based and space-based approaches to optical detection and tracking of artificial satellites. In: Paper presented at 31st Space Symposium, Technical Track, Colorado Springs, Colorado, United States of America

Authors and Affiliations

Sebastian Samuele Rizzuto[1] · Riccardo Cipollone[1] · Andrea De Vittori[1] · Pierluigi Di Lizia[1] · Mauro Massari[1]

✉ Sebastian Samuele Rizzuto
sebastiansamuele.rizzuto@polimi.it

Riccardo Cipollone
riccardo.cipollone@polimi.it

Andrea De Vittori
andrea.devittori@polimi.it

Pierluigi Di Lizia
pierluigi.dilizia@polimi.it

Mauro Massari
mauro.massari@polimi.it

[1] Department of Aerospace Science and Technology, Politecnico Di Milano, Via Giuseppe La Masa 32, Milan 20156, Lombardy, Italy