Arabian Journal for Science and Engineering (2023) 48:10589–10604
https://doi.org/10.1007/s13369-023-07669-0

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

The Image Processing System for Ultra-Fast Moving Space Debris Objects

Stanislav Krajčovič[1] · Jiří Šilha[2] · Matej Zigo[2] · Roman Ďurikovič[1]

Received: 5 August 2022 / Accepted: 25 January 2023 / Published online: 27 February 2023

Abstract
The ever-increasing number of space debris objects on the Earth's orbit presents a danger to existing functional satellites and human infrastructure. These objects need to be tracked to be documented and catalogued. The paper addresses the development of an image processing and data reduction pipeline to process space debris tracking observations from an optical passive sensor. The pipeline starts from the raw, un-calibrated camera frames and ends with the formation of "tracklets", i.e. consecutive series of celestial positions of the objects of interest and possibly an identification of these objects based on a reference catalogue. The paper is, on one hand, improving existing software modules, and, on the other hand, adding a series of new modules to the pipeline. The validation of the system's results in both astrometry and photometry and proves that it is one of the few capable of observing and processing low-Earth orbit objects.

Keywords Image processing · Correlation · Space debris · LEO

1 Introduction

Space debris presents risk to currently on-going missions regardless of its size. For example, International Space Station (ISS) needs to perform evasive manoeuvres to dodge space debris few times a year. In order to know where and when does the ISS need to dodge incoming fragments, they need to be properly tracked and catalogued. One such catalogue is the mentioned US Space Surveillance Network (USSSN). A service, SPACETRACK [1], represents a worldwide network of sensors which contribute to the USSSN. Similarly to USSSN, other entities around the globe have their own programmes to track space debris and objects of interest on the orbit such as the European Space Agency's (ESA's) Space Situational Awareness Programme [2].

Of course, not only ISS is at risk. Every currently functioning device, whether it is a weather station, GPS satellite or something else, is susceptible to potential collision with a space debris object, small or large.

It is undeniable that observing and tracking movements and behaviour of space debris objects is essential. This work focuses on data acquired by passive optical sensors which generate star field images.

Astrometry is a set of procedures, algorithms, and techniques involving measurements of positions and movements of celestial bodies.

Photometry is similar to astrometry, but it involves measurements of intensity of light reflected or radiated by celestial bodies.

In order to track a celestial object potentially dangerous to working satellites or ongoing missions, both astrometry and photometry need to be employed. Astrometry is to determine its orbit, and therefore its positions at any given time; photometry to determine its rotation period, or shape.

✉ Stanislav Krajčovič
stanislav.krajcovic@uniba.sk

Jiří Šilha
jiri.silha@fmph.uniba.sk

Matej Zigo
matej.zigo@fmph.uniba.sk

Roman Ďurikovič
durikovic@fmph.uniba.sk

1 Department of Applied Informatics, Comenius University, Mlynská dolina F1, 842 48 Bratislava, Slovakia

2 Department of Astronomy, Physics of the Earth and Meteorology, Comenius University, Mlynská dolina F1, 842 48 Bratislava, Slovakia

1.1 Existing Solutions for Image Processing

There are many institutions concerning themselves with space debris and related problems. This section lists mainly software solutions and observational networks currently active in the space debris field. The first three subsections: Source Extractor, WCSTools, and Astrometry.net, are software tools for partial problems of astronomical image processing. After that, OWL-net is an observational network of satellites, and APEX II is an extensive software system for the reduction of space debris images.

1.1.1 Source Extractor

Source Extractor (abbreviated by its authors as SExtractor [3]) is software commonly used for astrometric purposes. SExtractor allows a user to extract the background of an astronomical image. Most importantly, though, it is used to extract light sources and their positions in frame-relative coordinates. The software does not limit its use to space debris. It also allows to segment stars and galaxies and distinguishes between the two using a neural network. For decades, it has been considered a standard in image segmentation of astronomical images.

1.1.2 WCSTools

World Coordinate System (WCS) is a convention that allows for transformation between frame-relative coordinates (pixels, on X- and Y-axes) and celestial coordinates (most commonly in Equatorial Coordinate System, or EQS with right ascension α, declination δ). WCSTools [4] is, therefore, a software used to relate frame-relative coordinates to sky coordinates.

1.1.3 Astrometry.net

Astrometry.net (A.NET), as the name implies, is used for astrometric reduction, i.e. translation between frame-relative coordinates into celestial coordinates (in this case, EQS). A.NET does not require a first guess to correctly identify the portion of the sky present in an astronomical image. Instead, it uses a complex algorithm based on *quads*-tuples of four stars. A.NET uses index files containing quads of stars spanning the whole sky to correlate between quads in user-provided images. The complex explanation can be found in [5]. A.NET is extensively used in the astronomical community partially because it provides a command-line interface for ease of use.

1.1.4 OWL-net

OWL-net is a Korean network of satellites with the main purpose of tracking Korean satellites on geosynchronous orbits (GEO) and low-Earth orbits (LEO). After the acquisition of images, the processing system starts with the removal of additive and multiplicative errors with bias, dark, and flat-field corrections. Then, SExtractor (see Sect. 1.1.1) is used to segment light sources (either points or streaks) from the background in each image, and WCSTools (see Sect. 1.1.2) is used to transform between frame-relative coordinates and celestial coordinates by using WCS. Finally, the end product—real positions of satellites in time—is created by data reduction algorithms described in the article [6].

1.1.5 APEX II

APEX II is a data reduction system focused on astronomical images used mainly with telescopes in the International Scientific Optical Network (ISON). The pipeline's results are presented in the context of GEO objects. There are several steps before a result is produced: bias, dark, and flat-field correction to clean input images. Sky background flattening, image filtering, and enhancements belong to the second set of corrections possible on any given astronomical image. Global thresholding producing logical masks is used to reduce noise for segmentation. A threshold is given to segment objects with intensities above the pre-determined value. Deblending allows for the separation of overlapping objects and then follows isophotal analysis, Point Spread Function (PSF) fitting, astrometric catalogue matching, differential astrometry, differential photometry, and report generation [7].

1.2 Image Processing and Data Reduction Algorithms

This section describes the most important and commonly used algorithms for processing astronomical images/frames of space debris. Algorithms are introduced in the order in which a processing pipeline could be designed.

1.2.1 Correction of Additive and Multiplicative Errors

Image cleaning can be achieved by using general, common, astronomical algorithms. These are dark, bias, and flat-field corrections, the same as used in pipelines in Sect. 1.1. In addition, they correct additive and multiplicative errors.

Additive errors are caused by dark and bias currents in the camera. Acquiring a bias and dark frame (or image) and subtracting them from a frame corrects these errors. A bias frame is taken at the shortest (usually zero) exposure time to capture only the bias currents. A dark frame is taken at a specific exposure time to capture dark currents, collected on the pixels creating a dark current offset. The dark current already contains the bias signal.

Multiplicative errors are caused by different quantum efficiency of pixels, illumination differences, dust halos, or all at once. They are corrected by creating a flat field frame captured on an evenly illuminated field. For example, dusk or dawn near the zenith.

To increase the robustness of bias, dark, and flat-field images, it is also possible to create *master* frames, summing several bias frames, dark frames, and flat field frames together, respectively. For example, for the master bias frame M_b:

$$M_b = \sum_{i=1}^{N} B_i \,, \tag{1}$$

where N is the number of bias frames, B_i. The process is identical for master dark and master flat-field frames.

1.2.2 Background Noise Estimation

Background noise estimation and extraction are beneficial for several reasons, mainly because they improve segmentation and photometric reduction accuracy.

Initial background estimation can be done by using the median value of the image. Current software libraries providing common operations, including median calculation, are optimised and recommended. However, sorting 1024×1024 values (an example resolution of an image) and picking the middle value is computationally expensive if not done correctly. Furthermore, this approach has major disadvantages stemming from its simplicity—the median is an integer value and does not calculate the standard deviation. Improvement of this approach might be calculating a median of sub-images created from the image and using the lowest median of them as the background value [8].

More commonly used method, also in other science branches than astronomy, is sigma clipping. First, input image is shrunk to about 10% of its size by a spline filter, to minimise aliasing effect. The scaled image is then smoothed by a median filter and expanded to its original size. Map produced by this process is then base for sigma clipping algorithm:

$$\Delta B_{i+1}(x, y) = \begin{cases} \Delta B_i(x, y) & |\Delta B_i(x, y) - \overline{\Delta B_i}| < 3\sigma_i, \\ \overline{\Delta B_i} & |\Delta B_i(x, y) - \overline{\Delta B_i}| \geq 3\sigma_i \end{cases} \tag{2}$$

where $\overline{\Delta B_i}$ is mean deviation and σ_i is standard deviation of the background map correction $B_i(x, y)$ at i-th iteration and $\Delta B(x, y) = I(x, y) - B_0(x, y)$ where $I(x, y)$ is the intensity at a pixel given by (x, y). The final background map is calculated as:

$$B(x, y) = B_0(x, y) + \Delta B_n(x, y)\,, \tag{3}$$

where n is the number of iterations.

The main disadvantage of this method is that it can remove faint objects.

1.2.3 Segmentation of Light Sources

In order to segment the image and extract positions of celestial objects represented as points or streaks, it is vitally important to determine which pixels belong to the background and which to a light source.

Tree algorithm for light source recognition begins with an initial pixel presumed to belong to a light source. Then, a search for borders of the light source is performed along a tree-like path, testing the 4- or 8-pixels connectivity algorithm. A light source has been segmented if a 4- or 8-connected path from one pixel above a pre-determined threshold connected to another pixel can be found.

Other methods include blob detection, which uses Laplacian of Gaussian [9], Difference of Gaussian, and Determinant of Hessian [10]. All of these methods can be found implemented in common software libraries. The disadvantages of these are that they do not detect streak-like light sources, only point-like ones.

Similarly to blob detection, contour finding is a marching squares algorithm used to find contours in images. Contours can have any shape, which is very important for astronomical images. An example of contour matching used for problems related to astronomy is ASTRiDE [11].

Subsequently, each detected object needs to undergo centroiding technique, i.e. a technique in which the centroid of the object and its total intensity are calculated. Centroiding is vastly different between point-like and streak-like sources.

For point-like sources, it is assumed shape of the light source is either defined by a PSF function or the centre of the light source is in centre of gravity defined by light source's pixels' intensity values. PSF can be usually described by a symmetric Gaussian distribution [12]:

$$g(i, j) = \frac{A}{2\pi\sigma_x\sigma_y} \exp\left(-\frac{(i - x_m)^2}{2\sigma_x^2} - \frac{(j - y_n)^2}{2\sigma_y^2} \right) \tag{4}$$

where i and j are the integer coordinates of the pixel with coordinates of (i, j), and $g(i, j)$ is the intensity of the pixel measured from the image. A is the height of the PSF, σ_x and σ_y are the widths of the Gaussian PSF through which Full Width Half Maximum (FWHM) is determined. Finally, x_m and y_n define the position of the centre of the frame light source in the frame coordinates in decimal numbers.

If the centroid of the light source is deviated by $\Delta x = x_m - m$ and $\Delta y = y_n - n$ where (m, n) are coordinates of the centroid pixel, Eq. 4 changes accordingly:

$$g(i,j) = \frac{A}{2\pi\sigma_x\sigma_y} \times \\ \times \exp\left(-\frac{(i-m-\Delta x)^2}{2\sigma_x^2} - \frac{(j-n-\Delta y)^2}{2\sigma_y^2}\right). \tag{5}$$

The centre of gravity (COG) can be calculated the following way:

$$(x_b, y_b) = \left(\frac{\sum_{ij} I_{ij}x_{ij}}{\sum_{ij} I_{ij}}, \frac{\sum_{ij} I_{ij}y_{ij}}{\sum_{ij} I_{ij}}\right), \tag{6}$$

where (x_b, y_b) are the coordinates of the centroid; I_{ij} is the intensity of the i, j pixel with coordinates (x_{ij}, y_{ij}). A disadvantage of this fast algorithm is that it is highly sensitive to background noise.

For streak-like sources, it is assumed that the profile is elongated. The width of the profile Y is defined by seeing conditions at the observing site, while the length of the profile X is defined by the angular velocity of the light source's object. Both of these parameters are usually known before the processing.

Authors of [13] argue that moving 2D Gaussian method [14] has stable and accurate results. PSF convolution trail function is:

$$f_T(x', y') = b(x', y') + \frac{\Phi}{L}\frac{1}{\sqrt{2\pi\sigma^2}} \\ \times \int_{-L/2}^{+L/2} \exp\left[-\frac{1}{2\sigma^2}\{(x'-l)^2 + (y')^2\}\right] dl, \tag{7}$$

where (x', y') are coordinates of a specific point in the sensor reference frame, L is the length of the frame, Φ is the total photometric flux in the trail, and $b(x', y')$ is the background flux at a specific point.

Fitting of the trail is done by the following function:

$$f_G(x, y) = b(x, y) + \frac{\Phi}{2\pi\sigma_x\sigma_y} \\ \times \exp\left[-\left(\frac{(x-x_0)^2}{2\sigma_x^2} + \frac{(y-y_0)^2}{2\sigma_y^2}\right)\right], \tag{8}$$

where $b(x, y)$ is the background flux at a specific point. Fitting trail by the above function estimates centroid position, second-order moments, background, and total flux.

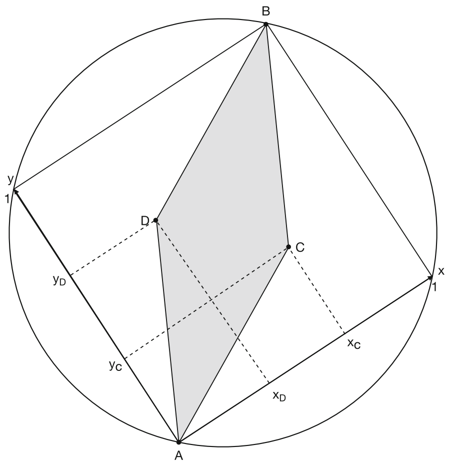


Fig. 1 An example of a quad structure used in Astrometry.net. Points A, B, C, and D are stars and they together create a unique shape

1.2.4 Astrometric Reduction

After segmenting the image and extracting light sources, the next step is translating between frame-relative coordinates and celestial coordinates.

The most relevant algorithm is that of Astrometry.net mentioned in 1.1.3 and shortly described in this section.

A.NET firstly constructs a hypothesis about the possible location of the astronomical image. The hypothesis is defined as a continuous four-dimensional space on the celestial sphere defined by the camera pointing (two parameters), orientation, and size of the field of view (FOV).

A geometric hash code is computed when given a set of stars in the image. A quad consists of 4 stars. A quad can be seen in Fig. 1.

Stars A and B are picked as the two most distant stars in this quad, defining a local coordinate system. Stars C and D must always be inside a circle defined by a diameter equal to the distance between A and B.

As mentioned in Sect. 1.1.3, the whole sky is reduced into index files that contain quads like the one in Fig. 1. For each quad calculated in an unknown frame, A.NET produces a hash code subsequently compared with quads inside indexed files. The closest the match, the higher chance the algorithm has identified the stars.

Due to the number of stars in the sky, each comparison creates many hypotheses. These are then filtered to remove false matches by applying the Bayesian decision-making algorithm, which is influenced by three factors:

1. The relative abilities of the models to explain the observations,
2. The relative proportions of true and false alignments it is expected to see,

3. and the relative costs of the outcomes resulting from the decision.

Transformation between frame-relative two-dimensional Cartesian coordinates (x, y) and equatorial coordinates (α, δ) requires plate constants to be calculated. However, there are three coordinate systems in total:

- the frame-relative two-dimensional coordinates (x, y),
- standard two-dimensional coordinates (X, Y),
- equatorial coordinates the right ascension and declination (α, δ).

Transformation from (α, δ) to (X, Y) is defined by central projection and spherical trigonometry. Required transformations for right ascension α are as follows:

$$\alpha = \alpha_0 + \tan^{-1}\left\{\frac{-X}{\cos\delta_0 - Y\cos\delta_0}\right\}, \tag{9}$$

and equation for δ is as follows:

$$\delta = \sin^{-1}\left\{\frac{\sin\delta_0 + Y\cos\delta_0}{\sqrt{1 + X^2 + Y^2}}\right\}. \tag{10}$$

In both equations, (α_0, δ_0) are coordinates of the point in the sky which the axis of the camera is pointing at.

Transforming frame-relative two-dimensional coordinates into standard two-dimensional coordinates has well-known equations as well:

$$X = \sum_{p,q} A_{pq} x^p y^q;\ p \geq 0;\ q \geq 0;\ p + q \leq A_{\text{order}}, \tag{11}$$

$$Y = \sum_{p,q} B_{pq} x^p y^q;\ p \geq 0;\ q \geq 0;\ p + q \leq B_{\text{order}}, \tag{12}$$

where A_{pq} and B_{pq} are polynomial coefficients for polynomial terms $x^p y^q$ and coefficients A_{order} and B_{order} define the polynomial order. The number of plate constants depends on polynomial order, so for first-order polynomial one needs six plate coefficients $(A_{10}, A_{01}, A_{00}, B_{10}, B_{01}, B_{00})$ to solve:

$$X = A_{10}x + A_{01}y + A_{00}, \tag{13}$$

$$X = B_{10}x + B_{01}y + B_{00}. \tag{14}$$

In order to calculate plate coefficients of a reference star for which its frame-relative two-dimensional coordinates (x, y) and its celestial coordinates (α, δ) from a star catalogue are known, one needs to use Eqs. 11 and 12.

A minimum number of reference stars needed for least-squares adjustment (required to solve Eqs. 11 and 12) is defined by the total number of plate constants to be solved. For linear polynomial, at least three stars are required, i.e. 6 plate coefficients, each star has two.

1.2.5 Star Removal: Masking

The main purpose of stars is to perform the astrometric reduction. After the process (as detailed in Sect. 1.2.4), it is recommended to remove the stars. There are two main reasons for this: correlating space debris objects with each other to produce complex structures describing space debris' trajectory can be computationally expensive; therefore, the fewer light sources there are in the image, the faster the calculation is; stars can introduce false positives when a light source of a space debris object is replaced by a star which fits into its trajectory better.

Thus, creating a mask that would cover pixels belonging to the stars (or even hot pixels, i.e. defective bright pixels) removes the stars from consideration in further processing steps, improving the pipeline's performance and reliability. A mask is generated by using an astrometric catalogue, the image field's centre position, its orientation and scale, and used tracking velocity, direction, and exposure time. After the stars are segmented by using algorithms in Sect. 1.2.3 and identified by a catalogue, their pixels are either set to zero or the previously calculated local background value. This approach also requires second use of background estimation, Sect. 1.2.2, to refine the background value or the background map.

Another approach involves abandoning images at earlier steps in the pipeline and only using extracted data in a text format.

1.2.6 Tracklet Building

Tracklet is loosely defined as a data structure containing observations of a celestial object in time. For example, assume there are eight images in a tracking series of a random space debris object. There is exactly one observation of the said object in each image. The images also contain timestamps of the exact time they were taken. If one were to construct a tracklet, they would need to put all eight observations into one data structure.

Tracklet construction has three important assumptions:

1. Apparent motion of the frame object in the image series is linear,
2. Apparent motion of the object of interest is performed on a great circle during acquisition,
3. Object of interest is on geocentric (circular or elliptical) orbit.

Apparent linear motion in the image series depends on the telescope's observational strategy. Even though space debris objects have circular or elliptical geocentric orbits, relatively short observations capture only a portion of their orbit, which appears to be linear. It is possible to perform orbit determination or orbit refining from data acquired this way, simplifying the problem.

The apparent linear motion in the image frame can be approximated by using a simple linear regression fit. If the object's position in an image is (x_i, y_i), where i is the i-th frame, and the motion is linear, it is true for each y_i:

$$y_i = \gamma + x_i\beta + \epsilon_i \,, \tag{15}$$

where γ is the intercept, β is the slope of the line, and ϵ_i is a random error. The goal of the simple linear regression fit is to determine these three parameters. Bearing in mind errors introduced in observation (additional and multiplicative errors described in Sect. 1.2.1), ϵ_i is used to evaluate the quality of the fit and therefore of the tracklet itself.

Cartesian coordinates still might possess errors that cannot be corrected by previously mentioned procedures in Sect. 1.2.1. For example, slight, unexpected movements of the camera might place the object in a different portion of the next image in the series. Angular measurements, i.e. celestial coordinates, do not have these errors. In order to be able to use celestial coordinates instead of Cartesian coordinates in a simple linear regression fit, the camera must have a small FOV, or the series must be in a short span of time. If both of these criteria are fulfilled, Eq. 15 might be used as follows:

$$\alpha_i = \gamma + \delta_i\beta + \epsilon_i \,, \tag{16}$$

where α_i and δ_i are equatorial coordinates in the i-th frame.

Limitations of this approach reside in the large FOV of a camera, long tracking time, when the object is close to its culmination point or when the centre of the FOV is close to the North or the South pole. Alternatives for simple linear regression fit in these cases are least-square fitting of the circle—geometric fit in parametric form and minimisation of the algebraic and geometric distances [15].

1.2.7 Object Correlation/Identification

This step aims to correlate the obtained tracklet with ephemerides of objects in available databases.

It is assumed that tracklet constructed in previous step consists from N number of observations and each observation point has at least three parameters:

- right ascension α_i,
- declination δ_i,
- observation time t_i,

where $i \in \langle 1, 2, \dots N\rangle$ and $t_i < t_{i+1}$.

By using the first and the last observation in the series, apparent angular velocity of the object in equatorial coordinates can be calculated by using the following formula:

$$\omega = \frac{1}{\Delta t}\cos^{-1}(\sin\delta_1 \sin\delta_N + \cos\delta_1 \cos\delta_N \cos(\alpha_1 - \alpha_N)), \tag{17}$$

where $\Delta t = t_N - t_1$.

Position angle, the angle between an object's apparent path and the North direction, can be calculated as:

$$\mathrm{PA} = \mathrm{atan2}\Big((\alpha_N - \alpha_1), (\delta_N - \delta_1)\Big)\,. \tag{18}$$

In this context, *atan2* is understood as a method in programming languages dealing with quadrants of the tan^{-1} function.

Correlation itself consists of comparing tracklet parameters for specific positions (α_i, δ_i), and ephemeris parameters for specific positions $(\alpha_i^{'}, \delta_i^{'})$. Specifically, angular distance for the same reference epoch, for example for the first observation time t_1:

$$\Delta = \cos^{-1}(\sin\delta_1\ \sin\delta_1^{'} + \cos\delta_1 \cos\delta_1^{'} \cos(\alpha_1 - \alpha_1^{'})). \tag{19}$$

Then, difference between position angles:

$$\Delta\mathrm{PA} = (\mathrm{PA} - \mathrm{PA}^{'}), \tag{20}$$

and finally, difference in apparent angular velocities:

$$\Delta\omega = \omega - \omega^{'}. \tag{21}$$

Apparent angular velocity ω for a tracklet and apparent angular velocity for an ephemeris $\omega^{'}$ are calculated according to Eq. 17.

The smaller the values of these three parameters, the higher the chance the object in the ephemeris is the object defined by the tracklet. Increasing robustness of the algorithm can be achieved by introducing thresholds for all of these parameters. However, they change between object types—apparent velocities of GEO objects are smaller by order of two than apparent velocities of LEO objects.

1.3 Instrumentation

Comenius University in Bratislava acquired a 70-cm telescope in 2016 [16]. The telescope is a Newtonian mount with an equatorial fork and a cooled Finger Lakes Instruments (FLI) charge-coupled device (CCD) camera. The camera's resolution is 1024 × 1024 pixels, with a pixel size of 24 μm. Focal ratio of the camera is 28.5 × 28.5 arc-minutes; pixel size is 1.67 arc-seconds/pixel. For more telescope properties information, see [17].

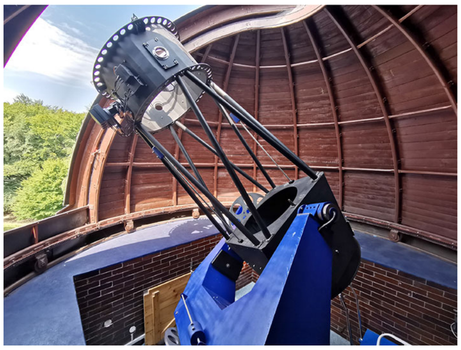


Fig. 2 AGO70 telescope seated in its cupola at AGO. Credit: Stanislav Griguš

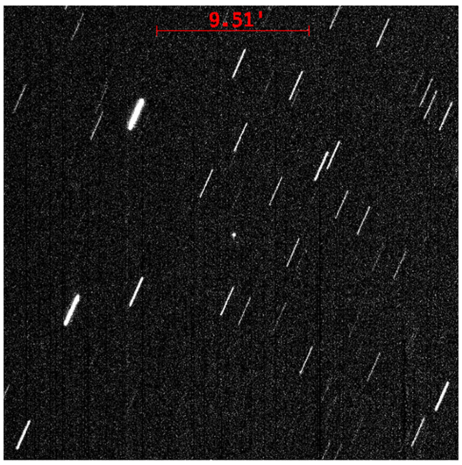


Fig. 3 Example frame taken by AGO70. Space debris object is a COSMOS 2500 (GLONASS) satellite, a point in the middle of the frame. Streaks in the background are stars. Image displayed by AstroImageJ [18]

The telescope is seated in its cupola at the Astronomical and Geophysical Observatory in Modra, Slovakia (AGO). Because of the primary mirror's diameter of 70 cm, it was named AGO70. Its main purpose is to be used in scientific research and academic applications (by astronomy students), for observations of space debris and NEA [16] (Fig. 2).

AGO70's hardware subsystems include:

1. CCD camera,
2. Colour filter wheel,
3. Mount motors and encoders,
4. Cupola motors and encoders.

All of the above is controlled by an observer using a Low-Level Telescope Control (LLTC) software, which is also responsible for creating frames and assigning them proper meta-data headers.

The Low-Level Telescope Control unifies the subsystems' interfaces and provides a concise summary of the system parameters, such as motor encoders' positions.

The camera interface connects to the FLI provided library, which lists all the required functions to control the camera itself. The filter wheel is of the same make, and the same library controls it. On the other hand, the Comenius University custom-made mount and cupola motors and encoders and their communication protocols follow packet structure. The packets' sizes range from 1 to 9 bytes, and each packet sent from the control computer and received by the controller chip of the motor and encoder structure prompts a properly defined response. This way, it is easily verifiable whether a packet had correctly been received or not.

Figure 3 shows an example of a frame/image created by AGO70. The image contains light sources represented by white point-like sources and streak-like sources. Depending on the observation strategy, space debris objects might be both points or streaks, same for stars. The frames generally also contain additive and multiplicative noise on black background. Additive errors are caused by the hardware, while imperfect observational conditions cause multiplicative errors. The observed object, in this case, is a GLONASS satellite, COSMOS 2500.

The produced images are in the format commonly used in the astronomical community, Flexible Image Transport System (FITS) [19]. In short, FITS files use the common structure of a header and body—or more than one of each—to store information. In the case of AGO70's images, there is one header containing meta-data about the frame itself and one body with a two-dimensional 16-byte header containing values ranging from 0 to 65,535, representing the intensity of a given pixel. LLTC creates the frames themselves by reading out data from the CCD camera and encapsulating them into the FITS format with custom headers. CFITSIO library [20] is used for this purpose.

Observational strategies at AGO70 usually create multiple frames per observation of one object, called a series. Each series has the same number of frames. The latest observational strategy tailored for best data extraction typically has *16* frames. Observational strategies are not only crucial for image series' properties but also for the character of each frame. Additionally, the nature of the followed object is essential as well. If it is a faint object, longer camera exposure is required to properly capture it, making stars appear as streaks. On the contrary, if the telescope is not currently tracking any objects and is moving in a *sidereal* mode (i.e. movement with the stars), a space debris object moving through the frame with relatively high exposure would be depicted as a streak, while the stars would be points.

In short, frames created by AGO70 might have tracked objects take the form of both points and streaks, same for stars, while the case of both the space debris objects and stars being streaks is the rarest. It is imperative for a potential software pipeline that would process these images to handle such cases.

2 Proposed Image Processing System

This section will describe the design and main functionalities of elements that perform tasks for the Image Processing System (IPS). New modules were designed, developed, and implemented. The modules are related to the classical CCD image reduction steps including bias, dark current, and flat field calibrations, and to the also classical photometric reduction of astronomical observations using photometric reference star catalogues. The improvements of existing modules described here concern technical issues mainly related to performance and parallel processing. New modules for the discrimination of moving objects, the building of tracklets, and the identification of objects in reference catalogues are presented.

2.1 Design

IPS most closely resembles the microservices architecture: it is divided into several parts, each with its exclusive tasks. Additionally, the tasks assigned to each part are as atomic as possible so that the parts can be as lean as possible.

The main philosophy of the IPS is that each Image Processing Element (IPE) is executable both in sequence with other IPEs as well as on its own. For example, the IPE responsible for the segmentation of images should be able to accept an image coming from a previous IPE (for example, one that would remove noise) and a raw, IPS-unrelated image of space debris.

In total, there exist 9 IPEs at the moment:

1. Image Reduction (IPE-IR),
2. Background Extraction (IPE-BE),
3. Objects Search and Centroiding (IPE-SC),
4. Astrometric Reduction (IPE-AR),
5. Masking (IPE-MR),
6. Tracklet Building (IPE-TB),
7. Post-processing (IPE-PC),
8. Objects Identification (IPE-OI),
9. Data Conversion (IPE-DC).

Figure 4 depicts the IPS and the flow of data. Dashed lines mark optional IPEs, which can be skipped if their purpose is unnecessary due to good data quality. Full lines mark mandatory IPEs. They need to be executed each time IPS runs, no matter the data quality. Furthermore, each line marks the flow of data from one IPE to another.

In the beginning, raw astronomical images can enter either IPE-IR, IPE-BE, or IPE-SC. They can also flow from IPE-IR to IPE-BE to IPE-SC; IPE-IR can be skipped, or IPE-BE can be skipped. IPE-SC is the first mandatory IPE, followed by IPE-AR, IPE-MR, and IPE-TB, all mandatory as well. Reasons as to why they are required will be duly explained later. Both IPE-PC and IPE-OI are optional, while IPE-DC is mandatory.

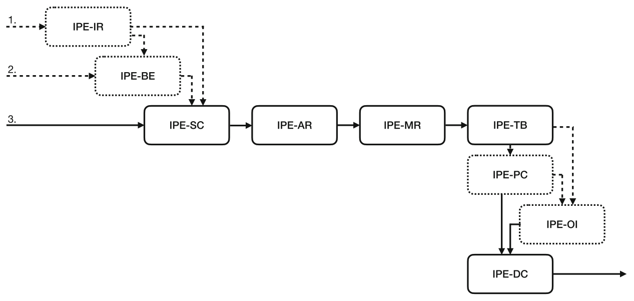


Fig. 4 Diagram of the latest IPS containing all the IPEs. Dashed lines mark optional IPEs, while full lines mark mandatory subsystems

The philosophy of which IPE is required and which is not follows the principles of modularity and fast processing times. Obligatory IPEs transform or create completely new data from existing, while optional IPEs change or correct them.

Each subsystem will be explained in its own following section.

2.2 Image Reduction

IPE-IR is the subsystem whose main goal is to correct additive and multiplicative errors, as described in Sect. 1.2.1. It also uses algorithms used to correct these errors, also described in the same section.

IPE-IR provides the user, or the pipeline, functionality to create master dark images and master flat-field images. There are several requirements for these images, which need to be fulfilled beforehand. First, each image must have the same exposure time; second, each must be taken at the same CCD temperature. Finally, each image must have the same dimensions. IPE-IR checks for these basic requirements, and if they are fulfilled, it creates the master frame. Master images can be created either by using the mean or median of the pixels. If the mean is chosen, IPE-IR takes a pixel at the same position in all the calibration frames used to produce a master frame, putting their average at that position in the newly created master frame. The process is analogical to the median. Master flat-field images need to be normalised beforehand.

Master frame by using mean is produced as:

$$M_{\text{avg}} = \begin{pmatrix} \text{avg}(\sum_{i=1}^{N} \text{pix}_{i,1,1}) & \cdots & \text{avg}(\sum_{i=1}^{N} \text{pix}_{i,1,y}) \\ \vdots & \ddots & \vdots \\ \text{avg}(\sum_{i=1}^{N} \text{pix}_{i,x,1}) & \cdots & \text{avg}(\sum_{i=1}^{N} \text{pix}_{i,x,y}) \end{pmatrix}, \quad (22)$$

where N is the number of calibration frames.

And master frame produced by median:

$$M_{\text{med}} = \begin{pmatrix} \text{med}(\text{pix}_{i,1,1}) & \cdots & \text{med}(\text{pix}_{i,1,y}) \\ \vdots & \ddots & \vdots \\ \text{med}(\text{pix}_{i,x,1}) & \cdots & \text{med}(\text{pix}_{i,x,y}) \end{pmatrix}, \quad (23)$$

where i denotes an image in the calibration series.

Additionally, IPE-IR also contains the capability to subtract, divide, and sum FITS frames. This is required not only for master frame creation but also for cleaning raw astronomical images of space debris (also called light frames, as opposed to calibration frames). As mentioned in Sect. 1.2.1, an image is cleaned from additive and multiplicative errors by subtracting or dividing it by calibration/master frames. IPE-IR has, therefore, two modes—creation of master frames and cleaning of raw frames. The first case requires a directory containing FITS calibration images as an input. The second use requires a directory of FITS raw light frames, a path to a master frame, and a parameter instructing IPE-IR which correction needs to be done.

Figure 5 depicts examples of master dark and master flat-field images created by IPE-IR. The calibration frames were taken by AGO70. Note the prominent halo in the master flat-field frame created by a dust particle.

Master bias frames are not created because dark currents corrected by master dark frames also contain the bias signal.

The output of this IPE is either a master frame, or light frames cleaned of additive, or multiplicative, or both types of errors.

2.3 Background Extraction

The product of the previous IPE (a cleaned raw frame), or a completely raw light frame, can be passed as the input of IPE-BE. It is responsible for estimating and extracting the background of an image.

Description of this problem along with some of the algorithms which are used to solve it is listed in Sect. 1.2.2.

IPS implementation of IPE-BE uses a subsequent (iterative) sigma clipping algorithm to create a background map. An example of such a map can be seen in Fig. 6.

IPE-BE allows the user to select how many iterations of sigma clipping are to be applied. Usually, the parameter is set to 5 iterations. The kernel used to convolve the image before sigma clipping is 15 × 15 Gaussian. The size of the kernel can also be changed.

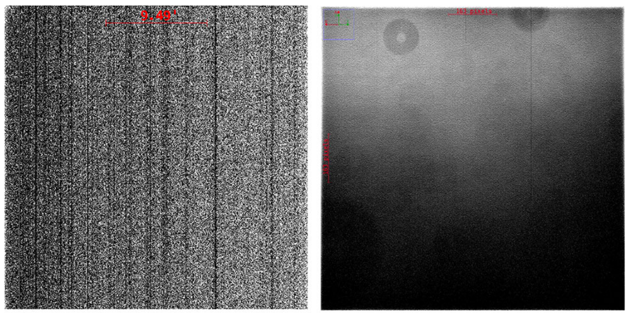

(a) Master dark frame (b) Master flat-field frame

Fig. 5 Example master dark frame and master flat-field frame generated by the IPE-IR. The calibration images from which the master was created were taken by AGO70

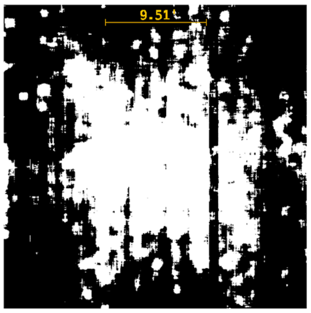


Fig. 6 Example of an extracted background from a raw image

"Pre-processing" of the frame before the sigma clipping contains resizing the image to 10% of its size, returning it to its original size, applying a median filter with 15 × 15 kernel size, and finally, convolved by the Gaussian filter.

IPE-BE only creates the background map. It does not provide the functionality to subtract the map from the image. Users can do that manually by using many tools, such as AstroImageJ. IPS handles it by using the subtract function of IPE-IR.

The background map is saved as a FITS frame on the disk.

2.4 Objects Search and Centroiding

Segmentation (as discussed in Sect. 1.2.3) in IPS is arguably one of the most important IPEs. Similarly to algorithms mentioned in Sect. 1.2.3, IPE-SC's main goal is to find and "extract" light sources in frames. The focus of previous IPEs was to clean the image as much as possible before it was segmented. IPE-IR removed noise that could be falsely classified as objects, and IPE-BE removed background to help identify faint light sources. If both of these IPEs were skipped, the results of IPE-SC would be objectively worse. During the routine use of IPS, it was found that IPE-BE's benefits out-

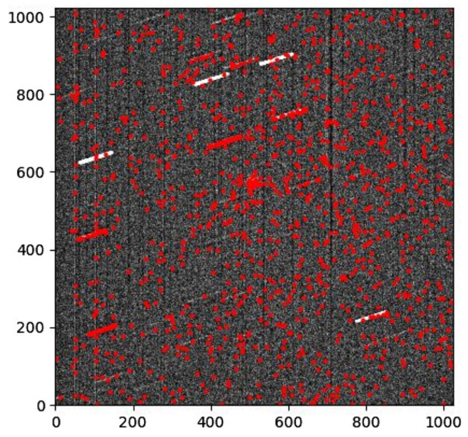


Fig. 7 Example of contour matching. Algorithm is set to a sensitive setting in order to find even faint contours. Each red point in the image represents the centre of a contour

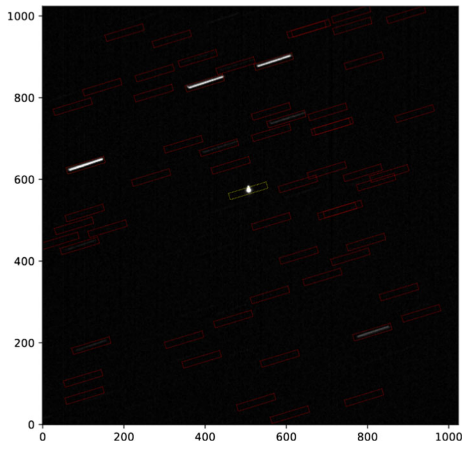


Fig. 8 Apertures fitted to objects are red rectangles. Note that the same aperture is also fitted to point light sources

weigh its costs and its execution in IPS is recommended each time.

The first step in IPE-SC is to find all contours which exceed a set threshold. These are potential light sources. See Fig. 7 for visualisation.

Each red point in the image represents a potential object. Note the long, bright streaks in the upper part of the image—there are only a few red points on them, signalling that the contour algorithm is sensitive enough to find multiple contours even on such bright objects. The contour matching algorithm is from the *scikit-image* package of the SciPy library, and they are quantified using ASTRiDE library [11]. The most important property of the contours is their centre, defined by pixel coordinates on both axes X and Y.

In order to properly extract valuable information about each light source, IPE-SC needs to use *apertures*. In the context of IPS, apertures are either squares or rectangles (depending on the user's setting). For example, in the context of Fig. 7, apertures are rectangles because it needs to capture streaks. Therefore, an aperture is defined by height h, weight w, and angle θ.

The next step of IPE-SC is putting an aperture on each possible object, i.e. now marked, just like the red points in Fig. 7. In the case of a sensitive contour matching algorithm, performance slightly suffers because the next step is the touch-down part of the IPE-SC when every aperture is iteratively moved in the direction of the most intensive pixels. It is almost guaranteed that after the end of this process, many apertures overlap. IPE-SC keeps only one. The touch-down algorithm is considered a centroiding algorithm. Finding the COG of an object is discussed in Sect. 1.2.3 (Fig. 8).

Properties extracted by IPE-SC are as follows:

- **cent.x**—position of the object's centroid on the X-axis,
- **cent.y**—position of the object's centroid on the Y-axis,
- **snr**—SNR of the object,
- **iter**—number of iterations of the touch-down algorithm,
- **sum**—total intensity of the object,
- **mean**—mean,
- **var**—variance,
- **std**—standard deviation of the pixels representing the object,
- **skew**—skewness of the object's profile,
- **kurt**—kurtosis of the object's profile,
- **bckg**—aperture's background value.

IPE-SC has the option to put these properties into a TSV or a JSON format. Each frame has its own text file. TSVs have the properties described above in columns, while each row represents one object from the frame. JSON is in a standard-valid array with JSON objects for each segmented object and properties as JSON properties.

There is no need to pass images further at this point in the pipeline. From now until the last IPE, IPS passes data via text files, either TSVs or JSONs, with some exceptions.

2.5 Astrometric Reduction

The problem of astrometric reduction is not a trivial one (see Sect. 1.2.4). While it is completely possible to create custom formulas to translate between frame-relative coordinates and celestial coordinates, it is more time and cost-efficient to use a robust, existing solution.

IPE-AR is nothing more than a thin wrapper on the CLI version of the A.NET procedures, namely the *solve-field* file. The wrapper accepts the output of IPE-SC as input, taking x and y values of each segmented light source. A.NET authors recommend sorting the light sources according to their magnitude/intensity. Brighter objects are more easily identified and should be at the top. IPE-AR sorts the entries, puts them into the recommended binary file, and passes that file into the *solve-field* procedure. There is also possible to input some additional information to the *solve-field* procedure to improve processing time—such as pixel size or initial guess. See [5] for all possible parameters.

After A.NET successfully finishes the reduction, it produces several files containing the sought information. The most important for IPE-AR is the binary *output.wcs*, which contains celestial coordinates in the same order as were the frame-relative coordinates. The form of the file is two columns of many rows. First column stands for right ascension (α, RA); second stands for declination (δ, DEC). IPE-AR pairs the frame-relative coordinates with the celestial coordinates and enriches the text file produced by IPE-SC.

If the text file was TSV format, the RA/DEC coordinates are appended as two last columns. If the text file was JSON format, they are added according to the JSON standard to each object they belong to.

An additional output of IPE-AR is the astrometric error of the reduction displayed as console output. Properties describing the astrometric error are:

- astrometric root-mean-square (RMS) [arc-seconds],
- astrometric RA RMS [arc-seconds],
- astrometric DEC RMS [arc-seconds].

2.6 Masking

IPE-MR does not produce masks as they are described in Sect. 1.2.5. IPE-SC removes the need for images to flow between each IPE, as it produces text files with all the required information from each frame in the series. Nevertheless, IPE-MR does the same as masking solutions introduced in Sect. 1.2.5.

IPE-MR algorithm needs to be appropriately changed to reflect that it works with text files. At this point, frame-relative coordinates in the text files are secondary, the important ones being the celestial coordinates RA and DEC.

By nature of celestial coordinates, stars between frames in series have the same celestial coordinates. If there is a star A in frame 1 with coordinates of 240° RA and 30° DEC, and a star B in frame 5 with coordinates of 240° RA and 30° DEC, it is with 100% certainty that both of these objects are one star. However, the problem of correlating objects with each other and finding the location of a star throughout the series can be computationally expensive. Additionally, minor deviations (for the AGO70 camera in arc-seconds) between locations also do not mean that the considered objects belong to separate stars. A threshold must be introduced (also to mitigate decimal representation in computers).

The algorithm is as follows:

1. Low-intensity objects are removed first (with ADU below 100).
2. Objects are sorted by RA, to effectively remove this axis from consideration while iterating over the objects and considering if they belong to the same celestial object or not.
3. Then, an object at index i and object at index $i + 1$ are compared.
4. If their angular distance (see Sect. 1.2.7) is less than the pre-determined threshold (6 arc-seconds in the case of AGO70), both objects are marked as the same object.
5. Then, the object at position $i + 2$ is considered. If its position is also similar enough to the object at position i, it is considered to belong to the tuple constructed in the previous step. If its position is too far, position i changes to $i + 2$, and this object is compared to the next one.
6. Repeat steps 3–5 until there are no more objects left.

Objects found by the algorithm described above are *removed* from the text file that is passed between the IPEs at this point. This is because they are considered stars and are redundant at this point. Moreover, they would increase computation time and could introduce false positives in IPE-TB in the next step.

The output of this IPE is the input file with fewer objects than at the beginning.

Percentage of the stars removed and additional information about IPE-MR can also be found in [21,22].

2.7 Tracklet Building

Tracklet building is the process of correlating objects from separate frames in the series with each other. Discussion about the problem of tracklets can be found in Sect. 1.2.6.

Due to the instrumentation of observations coming into the IPS (discussed in Sect. 1.3—specifically the relatively small FOV of the camera), IPE-TB needs to be able to handle only objects which adhere to linear motion. While simplifying the problem is welcome, IPE-TB can also handle objects on curved trajectories.

The process of how objects are correlated and a tracklet of the space debris observations begins with pairing every object from the first data file in the series with every object from the second data file in the series. This process creates tuples: $(o_{i,1}, o_{i,2})$, where i is the index number of the object in the frame. IPE-TB correlates objects according to their position and other parameters described later. If position is considered, the definition of objects in the tuples can be rewritten as: $(o_i(\alpha, \delta, 1), o_j(\alpha, \delta, 2))$, where j is the index number of the object in the data file, same as i. Every object's pairing ensures a guaranteed tuple that contains two observations of the sought space debris object.

The rest of the algorithm consists of finding the remaining observations. The first parameter important in finding the next observation of space debris is distance from an imaginary line $l_{i,1,j,2}$ such that $o_i(\alpha, \delta, 1) \wedge o_j(\alpha, \delta, 2) \in l_{i,1,j,2}$. Distance from the line can be flexible and is determined by a threshold D_l which can be changed freely. If the object fulfils the following equation, it is kept and considered to be, possibly, valid:

$$D_l < \frac{|a\alpha_0 + b\delta_0 + c|}{\sqrt{a^2 + b^2}}, \tag{24}$$

which is the standard Euclidean distance formula for a line given by:

$$ax_0 + by_0 + c = 0. \tag{25}$$

As discussed previously, celestial coordinates and frame-relative coordinates are interchangeable in observations performed by AGO70.

For objects which fulfil Eq. 24, there are few other conditions to be met.

IPS calculates baseline values for each tuple ($o_i(\alpha, \delta, 1)$, $o_j(\alpha, \delta, 2)$): baseline apparent angular velocity ω_{o_1,o_2} and baseline position angle PA_{o_1,o_2}. For observations belonging to the same object, it is safe to assume that apparent angular velocity and position angle do not change. Therefore, to find the next observation that would belong to the tuple, its apparent angular velocity ω_{o_{n-1},o_n} and position angle $\mathrm{PA}_{o_{n-1},o_n}$, where n is the number of an image in the series, must be below a pre-determined threshold. The threshold changes according to the type of observed object.

Considering 30 light sources in the first and the second data file in the series, the number of tuples is 900. Out of those, only two contain observations of real objects. It is with 100% certainty that some of the noise tuples will have assigned more objects to them.

After processing the last data file from the series and considering the previous example with 900 tuples, IPE-TB contains 900 tracklets. The vast majority of them have a length of 2. Some are longer than that, but the real object's tracklet should be the longest. However, if the parameters are loose enough, there is also a possibility that more than one object from a data file was considered valid. Sorting them according to their difference from the baseline values seems the best approach. Those closest to the baseline values are on the top of the list. Figure 9 displays this "two-dimensional tracklet".

Only the most probable objects in the tracklet are used to produce a tracklet file which is the output of this IPE. The rest of them are discarded.

Selecting these parameters during the design process of IPE-TB proved to be useful even though the main goal

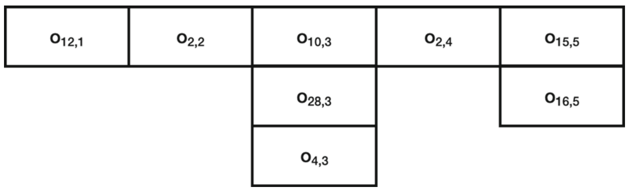


Fig. 9 An example of a tracklet. Objects in the first row are those which were classified as having the highest probability to belong to the initial tuple

was to segment straight lines. It is also possible to find curved objects along hundreds of observations by setting high enough thresholds.

The output of this IPE is a text file with no standardised format. It contains a head and a body part. While the head contains metadata about the series and the tracklet, data contain objects from the tracklet—their position in both coordinate systems, timestamp, magnitude, and others. For more information about IPE-TB, see [21,22].

2.8 Post-processing

Post-processing is responsible for correcting the data with known coefficients if applicable. Within IPS, two different corrections, epoch bias and annual aberration, were applied. Epoch bias is the bias in the time tag determined by comparing observed data with ground-truth positions obtained for calibration objects (see Sect. 3.1). Epoch bias can be stable over a longer period of time and is different for a given sensor [23]. For example, for AGO70, it has been identified during observation years 2018–2019 that the epoch bias had a value of $t_b = 67.7$ ms $\pm$ 6.8 ms. Once t_b is identified, it can be applied to the observations by applying the simple formula:

$$t_c = t_m + t_b \tag{26}$$

where t_m is the measured time present in the FITS header and t_c is the corrected time reported in the final output data.

Annual aberration is an apparent displacement of the stars (or non-Earth bounding space object) by the observer on Earth due to the Earth's motion around the Sun. The light velocity vector from the star adds to the Earth's motion velocity vector, resulting in an apparent change of star position up to 20.5 arc-min. Annual aberration is usually automatically applied within the astrometric reduction processes, such as the one used in Astrometry.net. However, this effect should not be applied to objects on geocentric orbits and therefore needs to be removed from the solutions obtained from the Astrometry.net engine. How to apply the annual aberration to the astrometric measurements can be found in [24].

2.9 Objects Identification

IPE-OI serves to confirm whether the observed object corresponds to the planned observations. On the other hand, it can also serve as accurate identification in the case of performing surveys of the sky.

Object identification is a straightforward problem with little room for innovation (see Sect. 1.2.7). The task consists of calculating angular distance, apparent angular velocity, and position angle for both a tracklet and a catalogue object and evaluating if they are similar enough. IPE-OI uses SatEph [25] for majority of calculations.

The algorithm is as follows:

1. Angular velocity of the tracklet is calculated by taking the first and the last object from the tracklet. For example, the angular velocity of the catalogue object is calculated by using SatEph and TLE catalogue.
2. Position angle of the tracklet is calculated in a similar manner to the angular velocity. Analogically for the TLE catalogue.
3. Angular position between tracklet and the TLE catalogue is calculated.

Equations for all three steps mentioned above can be found in Sect. 1.2.7. Obviously, the TLE catalogue contains many objects. Iterating over them and calculating the three properties for each is a must.

IPE-OI threshold for considering if the object matches is as follows:

- 5 arc-seconds for angular velocity,
- 10° for position angle,
- 360 arc-seconds for angular position.

2.10 Data Conversion

This IPE does not contain any algorithms or calculations. The main goal is to reformat the tracklet produced by IPE-TB into several standards accepted by international agencies to contribute to the effort of tracking space debris.

Among these formats there are:

- IAU Minor Planet Center format (MPC) [26],
- CCSDS Tracking Data Message format (TDM) [27].

2.11 Interfaces and Master Script

An overarching piece of software needs to exist that would accept initial input data and pass output data from one IPE to the next to ensure that IPS can be used as one application/system. While it is still possible to execute each IPE on its own and have the user maintain data validity within the IPS' each step, having a different solution that does it saves time (by leaving out the human factor), provides more flexibility concerning execution options, and also improves the quality of life for users, who now have to only run one script, instead of 10. Some IPEs also require numerous input parameters that can change from one processing to another.

The solution to the listed problems is creating a script that would call IPEs in the proper order, format outputs of one IPE into the input of another, and catch any problems that might unexpectedly happen during runtime. This would ensure that the user would not have to call each IPE in a console window with numerous input parameters, possibly reformatting the data. Such script could also have a configuration file that would store all values for the parameters and read it itself during execution. The script's performance is not critical because the heavy algorithmic workload is delegated to each IPE, optimised during its development. Furthermore, the performance of copying and moving of files (input/output data of the IPEs) is most dependent on hardware; therefore, potentially limiting factor of Python being used for the script and an interpreted language is irrelevant.

The script in IPS is called Master Script (IPS-MS).

IPS-MS benefits are most visible when performing observations of LEO objects, for example, during an observation campaign with Graz SLR. Selected LEO objects with high apparent velocities and short observation windows were observed by AGO70, processed by IPS-MS, and the results were sent to the Graz SLR. They parsed the results provided by AGO in real time and used them to target the space debris object and track it.

IPS-MS is still a console application at its core but has three modes:

1. Console mode,
2. User interface (UI) mode,
3. Fast mode.

The console mode is deprecated. A user had to provide every required parameter as text input into a console window. This means that every path to a file, for example, a master dark image generated previously, had to be copied and pasted into the console window.

The UI mode uses EasyGUI library [28] to provide the user running IPS-MS with pop-up dialogs containing options at each step of the IPS, significantly reducing the labour required by the console mode. Choosing a file from the example in the previous paragraph is in this mode done by displaying a file dialog, clicking on the file and confirming. Instead of typing "y" or "n" to the question if an IPE should run is done by a dialog asking the user the same question. The possibility of using the keyboard is preserved, but mouse control is possible in this mode too. Inputs needed from the user are a subset of all the parameters in the configuration file. However, parameters provided by the user take priority over those in the configuration file in this mode.

Lastly, the fast mode is entirely autonomous. It only uses the configuration file—therefore, every possible input parameter must be listed inside it. This mode was done for real-time LEO object observations processing. Moreover, it does not perform IPE-IR, and it times its execution.

Referring to Fig. 4, IPS-MS in fast mode performs IPEs in this order: IPE-BE, IPE-SC, IPE-AR, IPE-TB, IPE-PC, IPE-OI, IPE-DC.

IPS-MS makes heavy use of the *subprocess* Python library for spawning applicable IPEs as subprocesses and overseeing their execution. For IPEs such as IPE-SC and IPE-TB, if the mode supports it (typically UI mode), it allows the user repeated execution of the IPE—in the case that the results (which are also displayed by IPS-MS) are unsatisfactory.

Every partial result of the IPS—the result of each IPE—is saved into an appropriately named folder. This simplifies the debugging process and allows users to determine if the processing was of high enough quality.

The older version of IPS-MS used TSV file format after IPE-SC. The newer version uses JSON files, even though TSV files are still supported.

3 Validation

Two different types of data, astrometric positions, and photometric measurements are of focus to validate the IPS. For astrometry purposes, the object's relative positions on the sky are extracted. Those data are used for orbit determination (OD) and orbit improvement (OI) [29] and should help to maintain the catalogue of space objects for the close conjunction analysis [30]. Astrometric positions can be validated by observing objects with high accuracy predictions, which ephemerides—calculated positions on the celestial sphere for a specific time—are accurate below arc-seconds. Such data can be considered the ground truth.

For photometry purposes, acquired are photometric data to be used for space object's characterization [31] and attitude state estimation [32]. As a baseline to assess the quality of the photometry data provided by IPS the established astronomical tool AstroImageJ [18] was used.

3.1 Astrometry Validation

For astrometry validation, there are applicable two types of object groups: the so-called Global Navigation Satellite Systems (GNSS) and geodetic satellites situated on LEO/MEO, used for scientific purposes [33]. GNSS satellites are on high orbits with a mean altitude of around 23,000 km above the earth's surface. Their angular velocities are in order of tens of arc-sec/sec. Observations of GNSS with AGO70 are created

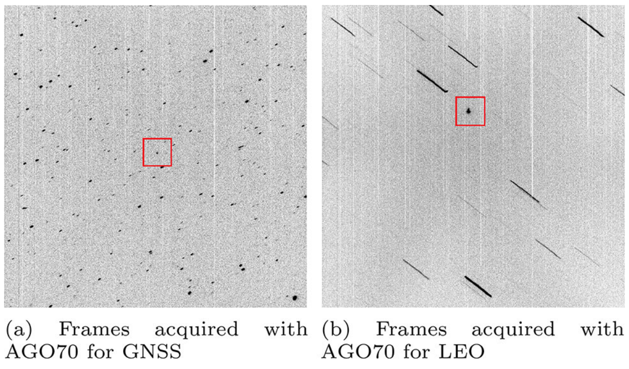
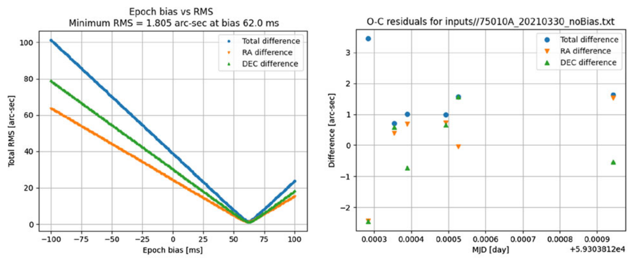

(a) Frames acquired with AGO70 for GNSS (b) Frames acquired with AGO70 for LEO

Fig. 10 GNSS object is GALILEO 6 (262) (object ID 14050B) acquired with C filter and exposure = 0.2 s. LEO object is SL-8 R/B (object ID 78053B) acquired with C filter and exposure = 0.1 s

FITS frames containing slightly elongated stars. For geodetic satellites, the apparent angular velocities are higher, order of a few tenths of degree/sec. This results in frames with stars being strongly elongated and with a relatively low number of stars with sufficient signal-to-noise ratio (SNR) to be used for further processing. Examples of frames acquired for GNSS and LEO objects are plotted in Fig. 10, objects are marked by the red square, while other objects on the frames are stars.

IPS performance can be validated by comparing measurements of calibration targets, GNSS and LEO geodetic satellites to the data published for these objects in Consolidated Prediction Format (CPF) provided by the International Laser Ranging Service (ILRS) [33]. This is done by our internal Data Validation System (DVS) software [34]. The basic principle of DVS is to calculate the ephemerides by using available CPF predictions; these are the ground truth positions (C-calculated), which are then compared to the angular measurements acquired by AGO70 (O-observed). By varying observation time, e.g. within an interval of -100 ms to 100 ms, we search for the smallest observed-minus-calculated (O–C) residuals represented by the root-mean-square (RMS). Once the minimum RMS is found, DVS estimates the total astrometric accuracy of the measurements. For the SST application, the aim is to reach RMS around 1–2 arc-sec [23]. Additionally, the minimum RMS also provides information about the system's epoch bias, independent of the IPS.

As seen in Fig. 10, astrometry of LEO objects is much more challenging than astrometry of objects on higher orbits such as GNSS. This is because the stars are more elongated, which leads to a more significant centroid estimation error, as well the number of stars is much smaller, which means fewer stars are available for the astrometric reduction. Therefore, we will demonstrate the IPS astrometric accuracy only on the LEO series in the next example. The result of the epoch bias analysis using LEO satellite Starlette (object ID 75010A, mean altitude of 956 km) observed during night 30th of March 2021 with the AGO70 telescope is plotted in Fig. 11.



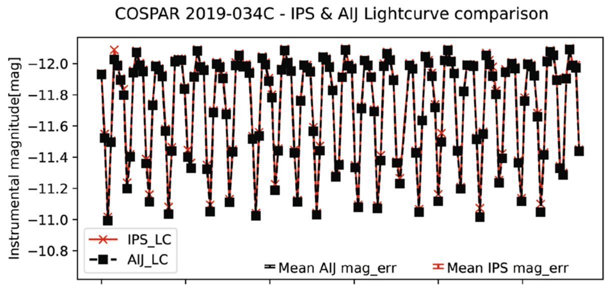
(a) RMS of RA, DEC and total residuals as function of epoch bias (b) O-C results for found epoch bias value of 62.0 ms

Fig. 11 Results of the internal Data Validation Suite (DVS) for object Starlette (object ID 75010A) observed by AGO70 during night 30th of March 2021

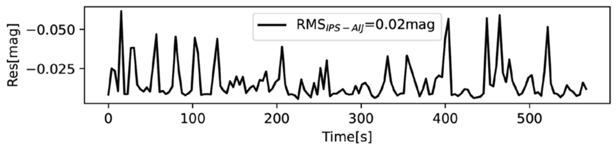
The object was used to identify the epoch bias, which was—according to DVS—around 62.0 ms and for this value, the RMS error was 1.8 arc-sec. The obtained epoch bias and astrometric accuracy values are fully consistent with the values obtained in the past by the AGO70 telescope but using a different tool applied only to GNSS orbits [16]. The astrometric data were further validated during orbit determination analysis performed in [34].

3.2 Photometry Validation

The IPS should be used for the automated processing of light curves acquired by the AGO70 system. To validate its quality performance, several different photometric series with IPS, as well with the reference tool AIJ [33], were processed. An example of the photometric series processed by the IPS can be seen in Fig. 12. It shows a light curve of Ariane 5 upper stage (object ID 19034C) observed by AGO70 on the night of 10th of March 2022. There are three types of information plotted in Fig. 12: instrumental magnitudes extracted by using AIJ [mag]; instrumental magnitudes extracted by using IPS [mag]; and their difference Res[mag]. One can see a small discrepancy between results obtained by IPS and AIJ. This is caused by the different types of apertures used during pixel intensity extraction. While AIJ uses a circular aperture with a diameter of 8 pixels, IPS uses a square aperture with a length of 8 pixels. This leads to a slightly different total area covered by the aperture, namely 201 square pixels compared to 256 square pixels, respectively, and covering different areas of the object's pixels compared to background pixels. The total RMS between AIJ and IPS was 0.02 mag for the instrumental brightness, an acceptable value for an object with brightness variation two orders larger than the RMS.



(a) Data processed by AstroImageJ (black squares) and by IPS (red crosses)



(b) The total RMS between these two data sets was 0.023 mag

Fig. 12 Top: Light curve of object Ariane 5 upper stage (object ID 19034C) observed by AGO70 during the night 10th of March 2022 by using 1.0-s exposure and clear photometric filter. Data processed by AstroImageJ (black squares) and by IPS (red crosses)

4 Conclusions

This work presents novel solution—the Image Processing System, which is used for the reduction of data acquired for space debris objects situated on low-Earth orbits. This software is deployed at Comenius University's telescope and is the first system in Slovakia capable of observing and processing any type of space debris with mean altitudes above the Earth's surface higher than 550 km.

Our proposed and implemented solution considers every step required for gaining scientific output from raw data by using state-of-the-art and own experimental algorithms. The proposed the solution efficiently connects each processing step securing that relevant outputs from one step are used as input for the successive step. This makes the system user-friendly and capable of processing observations in real time.

The system was validated using real observations and common calibration objects from the astronomical community. Additionally, external experts validated the system during orbital determination analysis.

Funding Funding was provided by ESA Contract No. 4000136672/21/NL/SC "Validation of re-entry models by using real optical measurements obtained by AMOS global network (Amos-Reentry)". (Grant Number 4000136672/21/NL/SC).

References

1. US Space Surveillance Network. SPACETRACK (2021). https://www.space-track.org/documentation#/faq. Accessed 2021

2. European Space Agency. Space Situational Awareness (2021). https://www.esa.int/Enabling_Support/Operations/Space_Situational_Awareness. Accessed 2021
3. Bertin, E.; Arnouts, S.: SExtractor: source extractor. In: Astrophysics Source Code Library. p. ascl–1010 (2010)
4. Mink, J.: Exploring space, time, and data with WCSTools. Astron. Data Anal. Softw. Syst. XXVII **523**, 281 (2019)
5. Lang, D.; Hogg, D.W.; Mierle, K.; Blanton, M.; Roweis, S.: Astrometry.net: Blind astrometric calibration of arbitrary astronomical images. Astron. J. **137**, 1782–2800 (2010) arXiv:0910.2233
6. Park, J.H.; Yim, H.S.; Choi, Y.J.; Jo, J.H.; Moon, H.K.; Park, Y.S.; et al.: OWL-Net: a global network of robotic telescopes for satellite observation. Adv. Space Res. **62**(1), 152–163 (2018)
7. Kouprianov, V.: Distinguishing features of CCD astrometry of faint GEO objects. Adv. Space Res. **41**(7), 1029–1038 (2008)
8. Schildknecht, T.; Hugentobler, U.; Verdun, A.; Beutler, G.: CCD Algorithms for space debris detection. In: ESA Study Final Report (1995)
9. Sotak, G.E.; Boyer, K.L.: The Laplacian-of-gaussian kernel: a formal analysis and design procedure for fast, accurate convolution and full-frame output. Comput. Vis. Graph. Image Process. **48**(2), 147–189 (1989)
10. Chen, B.Y.: An explicit formula of Hessian determinants of composite functions and its applications. Kragujevac J. Math. **06**(36), 27–39 (2012)
11. Kim, D.W.: ASTRiDE: Automated streak detection for astronomical images (2016)
12. Wang, H.; Xu, E.; Li, Z.; Jingjin, L.; Qin, T.: Gaussian analytic centroiding method of star image of star tracker. Adv. Space Res. **09**, 56 (2015)
13. Virtanen, J.; Poikonen, J.; Säntti, T.; Komulainen, T.; Torppa, J.; Granvik, M.; et al.: Streak detection and analysis pipeline for space-debris optical images. Adv. Space Res. **57**(8), 1607–1623 (2016)
14. Vereš, P.; Jedicke, R.; Denneau, L.; Wainscoat, R.; Holman, M.J.; Lin, H.W.: Improved asteroid astrometry and photometry with trail fitting. Publ. Astron. Soc. Pac. **124**(921), 1197 (2012)
15. Gander, W.; Golub, G.H.; Strebel, R.: Least-squares fitting of circles and ellipses. BIT Numer. Math. **34**(4), 558–578 (1994)
16. Šilha, J.; Krajčovič, S.; Zigo, M.; Tóth, J.; Žilková, D.; Zigo, P.; et al.: Space debris observations with the Slovak AGO70 telescope: astrometry and light curves. Adv. Space Res. **65**(8), 2018–2035 (2020)
17. Šilha, J.; Krajčovič, S.; Zigo, M.; Tóth, J.; Kornoš, L.; Zigo, P. et al.: AGO70 telescope Slovak optical system for space debris research surveillance and SLR tracking support. In: First International Orbital Debris Conference (2019)
18. Collins, K.A.; Kielkopf, J.F.; Stassun, K.G.; Hessman, F.V.: AstroImageJ: image processing and photometric extraction for ultra-precise astronomical light curves. Astron. J. **153**(2), 77 (2017). https://doi.org/10.3847/1538-3881/153/2/77
19. Pence, W.D.; Chiappetti, L.; Page, C.G.; Shaw, R.A.; Stobie, E.: Definition of the flexible image transport system (FITS), version 3.0. Astron. Astrophys. **524**, A42 (2010)
20. Pence, W.D.: CFITSIO, v2.0: A New Full-Featured Data Interface. In: Mehringer, D.M., Plante, R.L., Roberts, D.A. (eds). Astronomical Data Analysis Software and Systems VIII. vol. 172 of Astronomical Society of the Pacific Conference Series, p. 487 (1999)
21. Krajčovič, S.; Durikovič, R.; Šilha, J.: Selected modules from the Slovak Image Processing Pipeline for space debris and near earth objects observations and research. In: 2019 23rd International Conference Information Visualisation (IV), pp. 112–117. IEEE (2019)
22. Krajčovič, S.; Ďurikovič, R.; Šilha, J.: Masking and tracklet building for space debris and NEO observations: the Slovak image processing pipeline. In: Advancements in Computer Vision Applications in Intelligent Systems and Multimedia Technologies, pp. 38–56. IGI Global (2020)
23. Jilete, B.; Flohrer, T.; Mancas, A.; Castro, J.; Siminski, J.: Acquiring observations for test and validation in the space surveillance and tracking segment of ESA's SSA Programme. J. Space Saf. Eng. (2019). http://www.sciencedirect.com/science/article/pii/S2468896719300278
24. Green, R.M.: Spherical Astronomy, vol. 520. Cambridge University Press, Cambridge (1985)
25. Šilha, J.; Tóth, J.: Observations of orbital debris and satellites in Slovak Republic. In: 38th COSPAR Scientific Assembly, vol. **38**, p. 3 (2010)
26. The International Astronomical Union. Format for Optical Astrometric Observations of Comets, Minor Planets and Natural Satellites. https://minorplanetcenter.net/iau/info/OpticalObs.html. Accessed 01 April 2022
27. The Consultative Committee for Space Debris Systems. Recommendation for Space Data System Standards: Tracking Data Message (2020). https://public.ccsds.org/Pubs/503x0b2c1.pdf. Accessed 01 April 2022
28. Ferg, S.R.: EasyGUI: A module for simple GUI programming (2014)
29. Beutler, G.: Methods of Celestial Mechanics, Volume II: Application to Planetary System, Geodynamics and Satellite Geodesy. Cambridge University Press, Cambridge (2005)
30. Kerr, E.; Sánchez-Ortiz: state of the art and future needs in conjunction analysis methods, processes and software. In: 8th European Conference on Space Debris (2021)
31. Šilha, J.; Zigo, M.; Hrobár, T.; Jevčák, P.: Light curves application to space debris characterization and classification. In: Proceedings of 8th European Conference on Space Debris, ESA/ESOC, Darmstadt, Germany, 2021 (2021). https://conference.sdo.esoc.esa.int/proceedings/sdc8/paper/168
32. Santoni, F.; Cordelli, E.; Piergentili, F.: Determination of disposed-upper-stage attitude motion by ground-based optical observations. J. Spacecr. Rocket. **50**(3), 701–708 (2013)
33. Pearlman, M.R.; Degnan, J.J.; Bosworth, J.M.: The international laser ranging service. Adv. Space Res. **30**(2), 135–143 (2002)
34. Šilha, J.; Zigo, P.; Zigo, M.; Jevčák, P.; Krajčovič, S.; Steindorfer, M.; et al.: AGO70: passive optical system to support SLR tracking of space debris on LEO. In: Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS) (2021)