Selected modules from the Slovak Image Processing Pipeline for space debris and near Earth objects observations and research

Stanislav Krajčovič
Department of Applied Informatics
Comenius University
Bratislava, Slovakia
stanislav.krajcovic@fmph.uniba.sk

Roman Ďurikovič
Department of Applied Informatics
Comenius University
Bratislava, Slovakia
durikovic@fmph.uniba.sk

Jiří Šilha
Division of Astronomy and Astrophysics
Comenius University
Bratislava, Slovakia
jiri.silha@fmph.uniba.sk

Abstract—The Division of Astronomy and Astrophysics of Comenius University was granted resources from the first Plan for European Cooperating States (PECS) project as the ninth European Cooperating State with European Space Agency (ESA) to transform a newly acquired telescope to a professional observation device. One of the goals of the transformation was the design and development of an image processing pipeline which would be able to process an acquired raw image of space debris into object observations in time (tracklets), further correlate them with selected catalogs and identify them. The system contains 9 Image Processing Elements (IPEs) in total.

Index Terms—masking, tracklet building, algorithms, astronomy

I. INTRODUCTION

The risk of space debris has risen the moment humanity has become active in space exploration. Unmanned spacecrafts were rarely returned back to the Earth and this trend continues even now. It is more cost efficient to provide fuel only for the outbound journey and at their end of the life leave the man-made unmanned satellites burn up in the atmosphere, on their former orbit, or move them to farther orbits (such as the graveyard orbit). Because of this, the increase in launches has led to the exponential increase in the amount of space debris [2].

For the sake of future launches and the monitoring of the rising danger of space debris, many entities, either in private sector, government agencies or academia all over the world developed optical or radar based (or the combination of two) systems with different purposes. No matter the type of observational device, there is a lot of processing between acquiring the raw data and understanding what is in it. For optical systems, the consecutive steps are best done in a pipeline with the raw image as an input and the information we want as the output. There are several state-of-the-art pipelines all over the world employed by major entities - e.g. ESA OGS [3], APEX II [4].
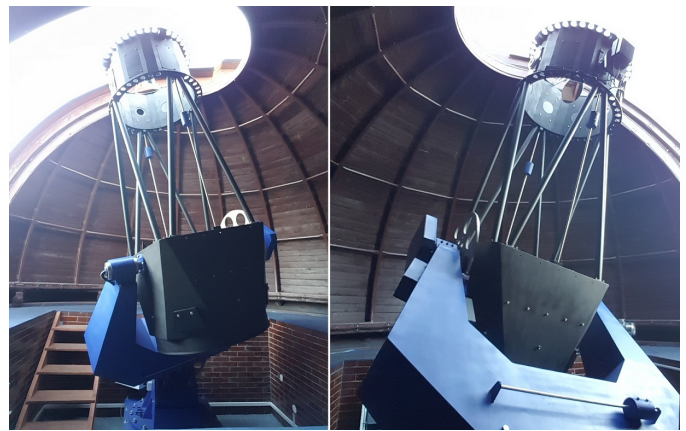
Fig. 1. AGO70 telescope in its dome at Astronomical and Geophysical Observatory in Modra, Slovakia, belonging to the Comenius University Bratislava.

Comenius University Bratislava has acquired the 70cm Newton telescope (see Fig. 1) for its Astronomical and Geophysical Observatory in Modra, Slovakia (AGO). The task contained many critical steps such as transportation of the telescope, installation of the cupola, installation of the telescope, acquisition of necessary (such as control unit, encoders, etc.) and the development of software [5]. As of now, it contains 9 Image Processing Elements (IPEs):

1) star field identification,
2) image reduction,
3) background estimation and subtraction,
4) objects search and centroiding (segmentation),
5) astrometric reduction,
6) masking,
7) tracklet building,
8) object identification and
9) data transformation.

II. IMAGE PROCESSING PIPELINE OVERVIEW

The pipeline has been purposefully divided into multiple discrete parts in order to create a modular design where each

module represents separate logical process. Nevertheless, the elements can and are joined together to perform the task from the beginning to the end seamlessly. However, the system is not yet fully automatic – inputs and corrections from a user are still needed [1].

This section shortly describes each IPE. See Fig. 2 for a diagram depicting the design of the pipeline.
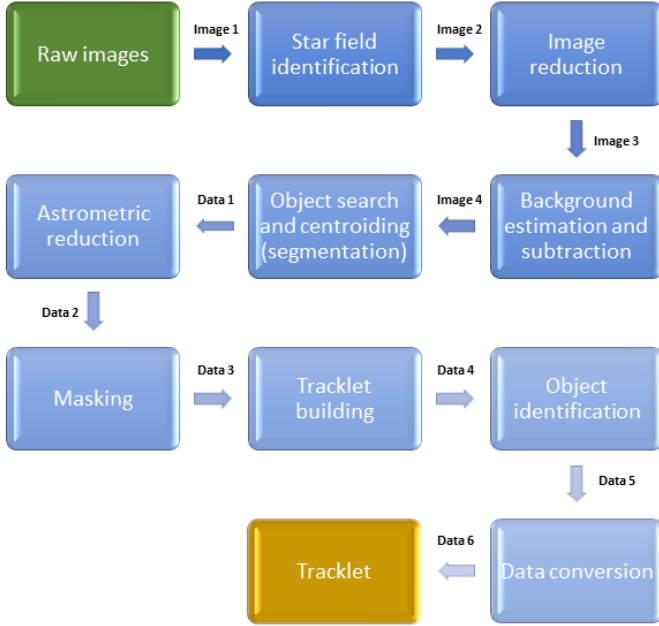

Fig. 2. Image Processing Pipeline design diagram showing all steps beginning from the top left and flowing according to the arrows. This paper is mostly focused on Masking and Tracklet Building blocks.

A. Star field identification

Star field identification is a wrapper on the well-known Astrometry.NET [6] scripts and is called each time an image is taken and saved in FITS format [7]. The FITS header is updated by the computed data - center of field-of-view in right ascension (α) and declination (δ), and plate constants [8] (also see Fig.3).

B. Image reduction

Image reduction is responsible for removal of additive and multiplicative errors (caused by heating of the charge-coupled device, while gathering data from the device, by the electrical current on diodes which transmit energy from photons, etc.) from the raw image. Additive errors are removed by subtracting a *dark* frame (taken at a specific exposure time to capture the dark current in pixels) or a *bias* frame (taken at zero exposure time; with the shutter closed). Multiplicative errors are removed by dividing the raw image by *flat field* frame (taken at dusk and dawn, close to zenith, on an evenly illuminated field). A corrected FITS image might look something like Fig. 5.
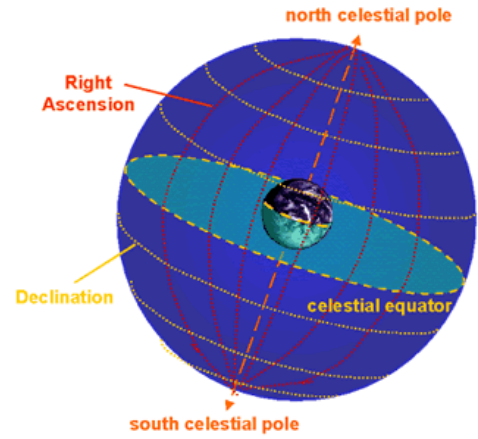

Fig. 3. Visualization of the equatorial coordinate system. Source: [13].

C. Background estimation and subtraction

Even after the previous step, raw images still contain a lot of noise (caused by cosmics (high energy particles hitting the diodes), imperfect observation conditions, etc.). This IPE uses subsequent sigma clipping [4] to estimate background and noise and remove it from the image.

D. Objects search and centroiding (segmentation)

This IPE combines three separate algorithms to yield results:

- the search algorithm to find a pixel which is above a predefined threshold (which depends on a frame object's signal-to-noise ratio (SNR)),
- the centroiding algorithm to measure a frame object's position (its centroid or center-of-mass) and the total intensity of the object,
- the touchdown algorithm which refines the centroid obtained in the previous step.

The output of this step is not an image but an interal text file containing vital information about objects (their intensity, position on the frame, etc.). See Fig. 4 for an example of a *.tsv* file.

Fig. 4. A .tsv file example. *cent.x* and *cent.y* are the coordinates of the objects' centroid in a two-dimensional space. *ra* and *dec* correspond to the same centroid in the equatorial coordinate system.

E. Astrometric reduction

Same as Section II-A, astrometric reduction uses Astrometry.NET scripts. The input of this IPE is the internal file mentioned in the previous section - it reads the (x, y) coordinates of every object in the *.tsv* file and

transforms them to the (α, δ) equatorial coordinates. The values are then appended to the *.tsv* file for further processing.

F. Masking

See Section III for an in-depth description of this IPE.

G. Tracklet building

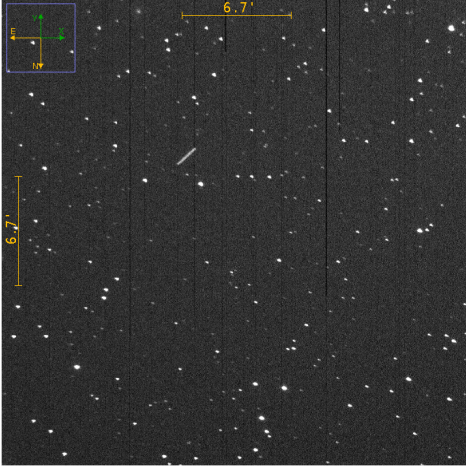See Section IV for an in-depth description of this IPE.

Fig. 5. A raw FITS frame example showing stars, noise and an object in the form of a streak.

H. Object identification

After creating a tracklet we need to correlate it with a catalogue to determine which object we were observing. We use three parameters - angular distance θ, position angle *PA*, and angular velocity ω. We calculate them for both the tracklet and the each catalogue object (by using SGP model and TLE data [11]) and compare the values. The output of this IPE is the object for which the difference between the two vectors of values was lower than the predefined thresholds.

I. Data format transformation

The conversion of astrometric positions from our internal format to Consultative Committee for Space Data Systems (CCSDS), Tracking Data Message (TDM) [12], AIUB's OBS, Minor Planet Center (MPC) format and Interagency Space Debris Coordination Committee (IADC) light curves formats is done in this IPE.

III. MASKING

By this point in the pipeline, we only work with the *.tsv* files (the IPE in Sec. II-D) provided us with the files). Each file contains information about the celestial objects the corresponding image contained.

Typically, each tracked celestial object is observed in a series of images. Since we track usually relatively slow celestial objects, the field behind the tracked celestial object remains unchanged for the most part - there are rare cases when a star visible in one frame can be shadowed by a different, less reflective celestial object or by a cloud, in the second frame.

On the other hand, it can become visible on the third frame and so forth. In short, masking is used to remove celestial objects if they appeared at the approximately same position in at least two different frames.

At a first glance, there are several problems with this approach. For one, what if we track an especially slow celestial object which can appear as if it doesn't move? Second, comparing each image with every other image can be slow and ineffective with hundreds of celestial objects in each *.tsv* file. Third, how do we determine if two celestial objects from different frames are actually the same celestial object? We solve these and many other hiccups in the design of our algorithm.

Naturally, the first step is to read all the provided *.tsv* files from the same series. Since this module is preceded by other IPEs which have modified the images and even the series themselves, it is safe to assume that the input text files are from the correct series and clean. However, as this IPE is the first one in contact with the freshly created *.tsv* files (files), their validity is ensured by checking each line and an exception is raised if there are missing or redundant columns. At this point in the algorithm, we have read all the files and have X frame objects in memory, where $X = NM$. Variable N is the number of files in series and M is the number of frame objects in a file. M can be calculated as

$$M = \sum_{i=1}^{N} m_i,$$

where m_i is the number of frame objects in the ith file.

It is vital to define which different frame objects actually represent the same celestial object. As mentioned before, there can be many disruptions, either internal or external, while tracking a celestial object. We therefore define a threshold σ_m [arcsec], which is a radius of a circle around a frame object. The next part is determining if a different frame object falls below this threshold. To do this, we calculate angular distance θ_{ij} [arcsec] of the two frame objects o_i and o_j by using cosine rule as follows:

$$\cos^{-1}[\sin \delta_i \sin \delta_j + \cos \delta_i \cos \delta_j \cos(\alpha_j - \alpha_i)].$$

Therefore,

$$o_i = o_j \Leftrightarrow \theta_{ij} < \sigma_m.$$

For a visual example of this mechanism, refer to Fig. 6 below. Object 2 from a different frame falls into the radius and is therefore considered the same celestial object as object 1 from an original frame. Object 3 from the same frame as object 2 is farther than the radius and is therefore considered a different celestial object.
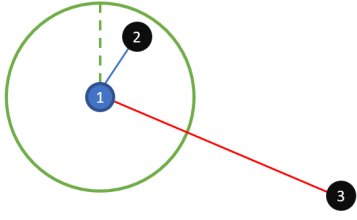
Fig. 6. A graph showing angular distance. The green ellipse around object number one represents the area of interest. Objects which fall in this region are considered identical and are removed. Object number three is considered a different object and is preserved.

Fig. 7. Ignored δ puts all the objects in one line and simplifies the problem. For visualization sake, we have "normalized" all δ values to an arbitrary value (in this case, 3).

For example, if two frame objects have $\theta_{ij} < 1$ arcsec/s, they are ruled as the same celestial object and removed from the memory. We are able to change the value of σ_m freely – for slow celestial objects, it is better to set it to a lower value than for faster celestial objects. As a general rule, the value needs to be slightly lower than the velocity of the tracked celestial object between frames. From our experience with data acquired by AGO70, the ideal value of σ_m is 6 arcsec/s.

Now, when we are able to identify same celestial objects, we need to actually check every frame object against every other frame object and compare them by using the aforementioned cosine rule. Naively comparing every frame object from one image with every frame object from other images can prove ineffective and the complexity is $O((k-1)\,n^2)$ at best, where k is the number of images. To mitigate this problem, we ignore the origin of the object – it is unimportant which file the frame object comes from. This reduces the time complexity to $O(n^2)$. Furthermore, it is nonsensical to compare two frame objects which are too far apart – there is no chance that the frame object in the left top corner could be the same as the frame object in the bottom right corner. The key to this problem is to sort the frame objects according to their right ascension α. As is depicted in Fig. 7, we ignore the δ dimension (y axis on the image) and look only at α (x axis). We iterate over the frame objects in such manner that we take o_u as the first frame object we are interested in and look at the closest frame object o_v in the list. The following property must be met if we want to proceed: $|\alpha_u - \alpha_v| < \sigma_m$. This characteristic optimizes our algorithm – there is no point in determining θ_{uv} if the difference between α_u and α_v is larger than σ_m. The time complexity after this step has been reduced to $O(n^2)$ in the *worst case scenario*.

After iterating over all the frame objects, we are ideally left with only unique stars and the tracked celestial object. In the worst case scenario, even the tracked celestial object could be removed from the data set. However, as can be seen in Fig. 10, in our testing cases, there were only two cases when this happened.
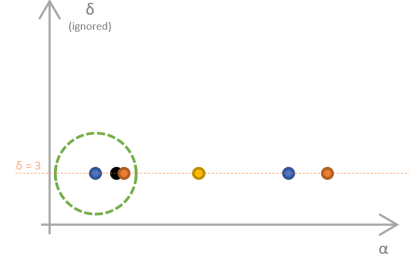
IV. TRACKLET BUILDING

A tracklet is most commonly defined as *a data structure containing consecutive observations of a frame object in time*. As is visible in Fig. 8, tracklet can be imagined as an imaginary trajectory of the celestial object.
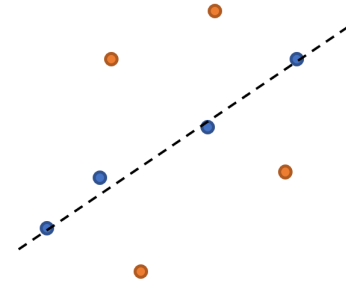

Fig. 8. One of many possible representations of a tracklet. Blue objects/objects which are closest to the line are the objects that represent a tracklet.

However, space debris, or other commonly observed celestial objects, orbit around the Earth. This problem can be simplified by the fact that the field-of-view in our case is so small that it appears that the celestial object moves according to the laws of linear motion [9]. We considered several algorithms which are commonly used in these types of problems, such as Hough transform [10], linear regression, etc. We chose Simple linear regression (SLR) which is a statistical concept modelling a relationship between a dependent variable, commonly denoted as y and an independent variable, commonly denoted as x. In principle, SLR models a line that describes trend in the data, or, in our case, the predicted trajectory of our celestial object.

The first step in applying SLR on our data is to take a frame object $p_{1i}(\alpha_{1i}, \delta_{1i}, t_{1i})$ from the first image and a frame object $p_{2j}(\alpha_{2j}, \delta_{2j}, t_{2j})$ from the second image, where t stands for epoch at which the object was observed. Then, we create a line l_{1i2j} such that $p_{1i} \wedge p_{2j} \in l_{1i2j}$. At the same time, we calculate baseline/ground truth values for:

- apparent angular velocity $\omega_{\alpha\delta,1i,2j}$ and
- position angle $PA_{\alpha\delta,1i,2j}$.

We do this for all combinations of all frame objects contained in the first frame c and in the second frame d. This yields $c_N d_N$ lines in total. The algorithm considers each

object for each line which means that the time complexity is exponential in the worst case scenario.

To determine if a next frame object $p_{3k}(\alpha_{3k}, \delta_{3k}, t_{3k})$ belongs to the tracklet, we firstly compare its distance from the previously created line. If the distance is lower than the predefined threshold σ_{tD}, we further calculate $\omega_{\alpha\delta,2j,3k}$ and also compare it to another predefined threshold $\sigma_{t\omega}$. If these two conditions are fulfilled, the last value compared with yet another predefined threshold σ_{tPA} is $PA_{\alpha\delta,2j,3k}$.

The process described in the previous paragraph is iteratively done with all the remaining frame objects in all the remaining frames, while the frame objects which fall below all the defined thresholds are added to the tracklet and the line is corrected in such manner that it reflects the new data. It is possible that several frame objects are considered at one step of the iteration. To clearly distinguish which frame object has the bigger possibility of belonging to the tracklet, we have introduced weights to the SLR. These weights are calculated by comparing the calculated values of D (as in distance), ω and PA with thresholds σ_{tD}, $\sigma_{t\omega}$ and σ_{tPA} and assigning bigger weights to those for which the absolute difference is smaller.

At the end of this process, we are left with a complex data structure containing ground truths, the line, successfully filtered frame objects, their values and weights. See Fig. 9 for a visualization of this data structure. Furthermore, we want to pass the frame objects which have the biggest fit to the tracklet into the next step. This is done by writing the tracklet into a text file. However, even though the previously mentioned complex data structure is 2-dimensional, we ignore the second dimension, which means that we effectively erase the additional frame objects which also fulfilled all the conditions but had lower weights.
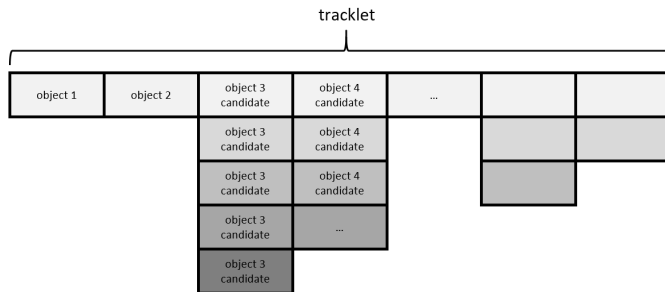

Fig. 9. A weighted tracklet example. Object candidates which are on the top are considered the most probable to be the object we are looking for.

V. SUMMARY AND RESULTS

We introduced a modular pipeline for space debris observations and described two of its nine complex Image Processing Elements in detail. It has been largely developed by the Faculty of Mathematics, Physics and Informatics (FMPI) of the Comenius University in Bratislava, Slovakia. The pipeline has been tested and validated at real examples taken by the AGO70 telescope. The fact that every IPE performs its operation in real time, we are able to process large amounts of data fast. Currently, the system is being used at AGO and
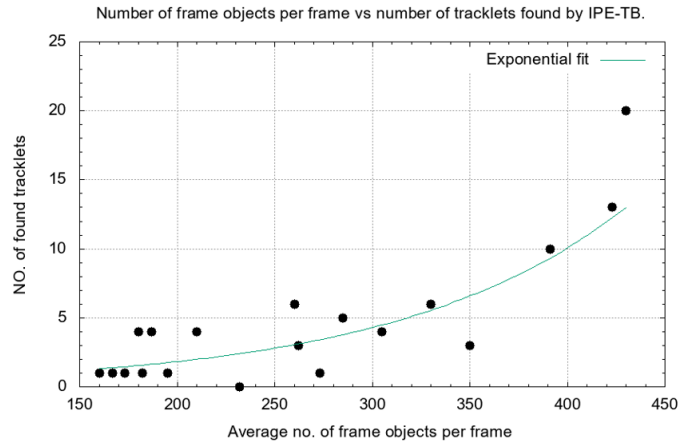

Fig. 11. Tracklet building results across tens of tracklets. Number of found tracklets is a function of average number of frame objects per frame.

is further improved with experimental algorithms and tested. In the future, we plan to adapt the pipeline to other optical sensors operated by the FMPI and test it on more difficult cases.

For results for the two picked parts, namely masking and tracklet building, see Fig 10 and Fig. 11 below.

Object	Orbit type	No. of images / frames	No. of frame objects	No. of removed frame objects [%]	No. of stars	No. of removed stars [%]	No. of real objects after removal
2018 CB	NEA	7	1383	**71.7**	737	**84.5**	7/7
11036A	GNSS	8	939	**80.6**	264	**95.5**	8/8
E10277A	eGEO	8	814	**77.0**	139	**99.3**	8/8
E14328A	Molniya	11	1139	**79.6**	209	**99.0**	7/8
E15204A	GEO	17	2160	**74.9**	772	**96.0**	16/17

Fig. 10. A small excerpt of all the tested cases. We have hand-picked few interesting cases, namely the two observations of real objects which were incorrectly deleted.

We only present a small sample of tested cases for the sake of brevity. The mean value of all (along with those not included in the figure above) removed stars is 94.8%, while the mean value of all objects removed is 75.5%. Out of 159 real celestial tracked objects, we lost only 2.

Fig. 11 clearly shows an exponential relationship between the average number of frame objects and the number of found tracklets in the tracklet building IPE (IPE-TB in the figure). This result has been expected – larger number of frame objects leads to more initial combinations and therefore more created lines. Also, resolution of the frames does not change, which means that higher object number frames have higher density of before-mentioned objects.

REFERENCES

[1] J. Šilha et al., "Slovak optical telescope and image processing pipeline for the space debris and NEA observations and research," ESA NEO and Debris conference, 2018.

[2] H. Klinkrad, "Space debris: models and risk analysis," Springer, Praxis Pub, ISBN 978-3-540-37674-3, 2006.

[3] T. Schildknecht, U. Hugentobler, A. Verdun, G. Beutler, "CCD Algorithms for Space Debris Detection, Final Report," 1995.

[4] V. Kouprianov, "Distinguishing features of CCD astrometry of faint GEO objects," Advances in Space Research, vol. 41, issue 7, p. 1029-1038, 2008.

[5] J. Šilha et al., "Slovakian Optical Sensor for HAMR Objects Cataloguing and Research," 69th International Astronautical Congress 2018 (IAC2018), 2018.

[6] D. Lung et al., "Astrometry.net: Blind astrometric calibration of arbitrary astronomical reduction," The Astronomical Journal, Volume 139, Number 5, 2010.

[7] D. C. Wells, E. W. Greisen, R. H. Harten, "FITS: A Flexible Image Transport System," Astronomy and Astrophysics Supplement Series, 44: p. 363–370, 1981.

[8] David A. Vallado, Fundamentals of Astrodynamics and Applications. Microcosm Press, El Segundo, CA., p. 157. ISBN 1-881883-12-4, 2001.

[9] H. Oda, H. Kurosaki, T. Yanagisawa, M. Tagawa, "Optical observation, image-processing, and detection of space debris in geosynchronous Earth orbit," 40th COSPAR Scientific Assembly, 2014.

[10] R. O. Duda, P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," Comm. ACM, Vol. 15, pp. 11–15, 1972.

[11] Vallado et al., "Revisiting Spacetrack Report #3," The AIAAAAS Astrodynamics Specialist Conference, American Institute of Aeronautics and Astronautics, pp. 1-88, 2006.

[12] Consultative Committee for Space Data Systems, "Tracking Data Message," CCSDS 503.0-P-1.0.4, CCSDS, Pink Book, 2017.

[13] B. Germany, R. Proctor, Ch. Fluke, A. Gaztelu, G. Mackie, S. Maddison, A. Lagos, V. Kilborn, M. Bailes, "About Cosmos: The Swinburne Astronomy Online Encyclopedia", Swinburne University of Technology, http://astronomy.swin.edu.au/cosmos/, accessed: 14.05.2019