Streak detection of space debris by a passive optical sensor

Rudolf Haussmann[1], Paul Wagner[2], Tim Clausen[2]

[1]Kormoran Technologie, Alte Poststraße 24, 88690 Uhldingen-Mühlhofen, Germany
[2]German Aerospace Center (DLR), Institute of Technical Physics, Pfaffenwaldring 38-40, 70569 Stuttgart, Germany

Abstract

We are developing a passive optical system for the detection of space debris and the determination of their orbits to allow subsequent laser ranging to uncatalogued objects. In this contribution the focus is on the software for processing the images and performing the subsequent calculations. The hardware is a sensitive camera which records pictures every few seconds of a fixed segment of the sky. The software runs in an independent process, loads the images one after the other and starts the calculations. The hardware and the software communicate via inter process communication using a *named pipe* or a *message queue*. The software works in four steps. In a first step a smoothed background image is calculated and subtracted from the original image. In this way the spectral curve in the histogram is contracted into a sharp peak with a small width which represents the background light of the blue sky. A star, a satellite or a space-debris part must be brighter than the background light. On the upper intensity side of the background peak we define a robust threshold value which is used to identify the objects to be observed as regions with larger intensities than the threshold. This method works quite well and independent of the position of the pixel, whether the position is close to the center of the image or close to the boundary or somewhere in between. The main challenge is to establish a robust detection of the objects under a wide range of optimal, normal, and weird conditions which include black night, gray dawn, partly clouded skies, and lens flares.

Thus, in a second step the objects (a *point* for a star or a *streak* for a satellite or space-debris part) are detected. The related x/y coordinates are calculated in pixel units in the image coordinate system. This is done by calculating intensity moments and by adjusting an intensity function. Then, in the third step we transform to ra/dec (right ascension/declination) coordinates of the equatorial coordinate frame. For this purpose we use *astrometry*. The observed stars are compared with known star positions of a catalogue in a database so that the parameters for the transformation are determined. Then the positions of the satellite and space-debris streaks are transformed from x/y to ra/dec. In a fourth step the streaks of satellites and debris objects observed in several different images are combined into traces in the sky which represent the orbits of space objects. Streaks are correlated by the angular velocity and direction. Finally, we fit a straight trace line with constant velocity into the observed data. This way we obtain a circular fit of the satellites and the debris trajectory. Eventually, for each found orbit the results are written into a “tracking data message” (TDM) which allows data sharing with other stations or databases. We present results for images taken under some representative conditions which are clear skies at deep night, partly clouded skies, and weird conditions with severe overexposure and lens flares.

Keywords: passive optical staring, space situational awareness, low Earth orbit, initial space debris detection, image processing, astrometry

1 Introduction

A higher number of space debris leads directly to an increased collision risk of active satellites. Already small debris can lead to major damage to an active satellite. To prevent collisions between active satellites and debris, the space debris in the orbits must be regularly cataloged. The orbit of the space debris must be measured and predicted with a high degree of accuracy. For objects in low Earth orbit (LEO), the laser ranking method provides accurate predictions [1]. In order to measure the range of an object using laser runtime measurements the object needs to be tracked. Due to the small field of view (FOV) of a tracking system the object orbits need be known prior to observation. This requires a separate sensor network to deliver orbital predictions to a laser ranging system. Currently only radar systems are used for detection of unknown objects in LEO. Their downside is their high hardware and operating cost. This is why a passive optical staring system is developed, to detect and measure unknown orbital objects in LEO for subsequent laser tracking.

Such a passive optical staring system is a camera system which records images of the night sky with a fixed line of sight (LOS). With a reasonable short exposure time (depends on focal length) stars are recorded as points and LEO objects appear as streaks. This requires the object to be illuminated by the sun while the observer is in the earth shadow (at night).

The DLR Institute of Technical Physics reported in 2015 [2] and 2016 [3] first results of such a system. For reliable operation further improvements are required to our system. One particular important extension is a reliable image processing chain, which ensures reliable detection of streaks from orbital objects and rejecting images that contain unwanted objects or distractions, like clouds or flares.

The paper is organized as follows. In chapter 2 we briefly describe the hardware setup and the image recording software. The main part of the paper is chapter 3 which provides a detailed description of the image processing software and its four processing steps. In chapter 4 we present results for some pictures taken under representative conditions. Finally, in chapter 5 we give our conclusions.

2 Hardware setup

The system contains an imaging camera with a lens, a GPS receiver for time synchronization and a computer for image recording and processing. For autonomous operation a weather station is used to automatically toggle image acquisition and a weather proofed housing protects the equipment from the environment. As observation location the first tests were performed at the UFO, Sternwarte Stuttgart [2] and later moved on top of the roof of our office building in Stuttgart Vaihingen [4]. A picture of the current system is shown in Fig. 1.

Fig. 1: The staring system (right) during observation located on the top of the DLR office building in Stuttgart, Germany. The ISS is visible as a streak in the left of the image. The bright object in the center is the full moon.

2.1 Camera and Lens

The system is based on passive optical measurements and the main components are a camera and a lens to perform angular measurements. As camera we are using large area CCD (charge coupled device) imaging sensors, namely the FLI PL16804 and the FLI PL09000, see Tab. 1 below for their specifications.

Properties	FLI PL16803	FLI PL09000
Sensor	On Semi KAF-16803	On Semi KAF-09000
Sensor size: width x height, diagonal	36.8mm x 36.8mm, 51.7mm	
Pixels	4096 x 4096	3056 x 3056
Pixel count	16.8Mpx	9.3Mpx
Pixel size	9µm	12µm
Full well capacity	100 000e^-	110 000e^-
Typical system noise (@8MHz read out speed)	14e^-	15e^-
Dark noise (@-35°C)	0.005e^-	0.02e^-
Shutter	Mechanical blade shutter (Uniblitz CS-65)	
Common exposure speed	1s	
Common binning setting	2	
Read out rate	8MHz	
Transfer interface	USB 2.0	

Table 1: Camera specifications and settings as used by the staring system. [5]

These cameras were combined with 2 different lenses, a 135mm and a 200mm consumer single lens reflex camera lenses. The specifications for these 2 lenses can be found in Tab 2.

Properties	Canon EF 135mm f/2 L USM	Canon EF 200mm f/2 L IS USM
Bayonet	Canon EF	
Focal length	135mm	200mm
f#	2	2
Aperture diameter	67.5mm	100mm
FOV: width x height/diagonal (@FLI)	15.2°x15.2°/21.0°	10.4°x10.4°/14.5°
Pixel scale (@PL16803)	13.8arcsec/px (67µrad/px)	9.3arcsec/px (45µrad/px)
Pixel scale (@PL09000)	18.3arcsec/px (89µrad/px)	12.4arcsec/px (60µrad/px)

Table 2: Lens properties of the two lenses used for observation campaigns. [6]

2.2 GPS synchronization

For the GPS synchronisation, a self-developed Arduino GPS timer based on the Adafruit Ultimate GPS receiver and the Arduinio Uno microcontroller is used [7]. It can be used to process the UTC timestamp of an incoming TTL pulse from the camera. The Arduino GPS Timer compares the incoming TTL pulse signal from the camera with the PPS signal coming from the GPS receiver. This way, it measures the time of a TTL signal with 100µs of precision [8].

2.3 Module for hardware control and image acquisition

The image recording is managed by the so-called Staring Daemon, which is a Python 3 program based on the OOOS software package [9]. The task of the Staring daemon is to control the required hardware (weather station, GPS timer, camera, enclosure), to start the image acquisition and to manage the data transfer between the individual processes.

The OOOS software package is designed in a modular way and is therefore not hardware-bound. This enables the user to use different hardware depending on the respective requirements. The weather station is managed by the so-called Environment Deamon. The weather data is stored within this process. The camera is controlled by the Acquisition Process, which synchronize the acquisition time by the connected GPS timer. The captured images are handed over by the Staring Deamon to the image processing program (presented later in chapter 3) using a message queue. Measurements, such as weather data, are made accessible online via the so-called Internet Deamon. The following Fig. 2 gives an overview of the external programs (Space Debris,

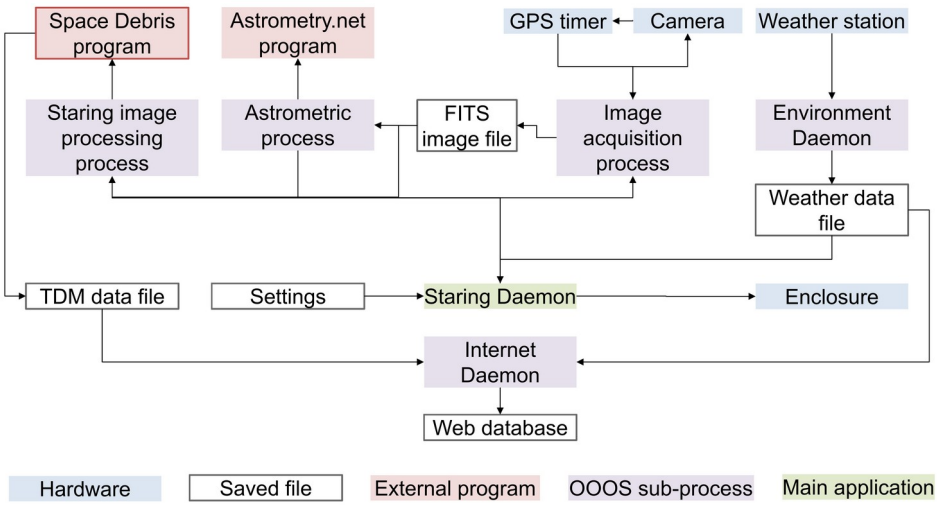

Fig. 2: A schematic illustration of the different sub-processes (purple) and hardware interfaces (blue), which are connected to the main “Staring Daemon” program (green). External software programs which are used by the system are marked in (red). Data file storage with in- or output is shown with a solid black contour.

Images are recorded as FITS files and important meta data is saved in the image header. This includes geodetic coordinates, line-of-sight, sensor specifications, lens properties, GPS synced time of start and end of the exposure. The image processing module is separate from the python program, which is why the image processing only takes the FITS files as an input. As output the image processing saves the angular equatorial coordinates in a Tracking Data Message (TDM) file for each measured object. The TDM file format is a standardized format to save and exchange measured positional information of measured residential space objects (RSO). The staring daemon also handles the data exchange of the TDM files to a database, which allows further processing and handover. In this paper we present this image processing pipeline in more detail.

3 Software for image processing and object detection

The software is written in c++ for a desktop computer with an Ubuntu Linux operating system. We mainly use Ubuntu 18.04 LTS, however, the software was tested successfully also on Ubuntu 20.04 LTS and on the current openSUSE Tumbleweed rolling release (of March 2021). Two versions were developed. First, a GUI version using the Qt 5 library [10] allows the operator to watch each step of the image processing and to adjust and optimize the parameters. Secondly, a console version is provided for an automatic operation of the machine. Both versions are designed to process a whole folder of recorded images. The second version is designed additionally for online image processing where the staring daemon can be connected by the inter process communication (IPC) of the Linux system using either a *named pipe* or a *message queue*. In the operational mode of the entire system, the communication via a message queue is used.

The images are read in the FITS format which is widely spread in astronomic and astrometric applications. For this purpose we use the software package *cfitsio* [11]. Since this package is written in the programming language c while our software uses c++, we need the wrapper *CCfits* which makes the functions available in c++. The image processing is done by self-developed algorithms supported by some functions of the computer vision library *openCV* [12]. For the astrometric calculations we use the software package *Astrometry.net* [13]. It turns out that all those software packages are already included in the Ubuntu 18.04 LTS operating system so that we may just install them. Alternatively, if we want to use the most recent versions of the *Astrometry.net* we must compile this and some other software packages by ourselves.

The software works in four stages. First, the image is preprocessed by searching for bright objects like stars and streaks of satellites. A background image is calculated, smoothed, and subtracted. The contrast of the image is enhanced so that the objects are more clearly seen and optimally detectable. Second, the stars and satellite streaks are detected. Their positions are calculated in pixel coordinates of the image. Third, astrometry is applied in order to transform the positions into the equatorial frame of the earth in ra/dec (right ascension/declination) coordinates. Fourth, the resulting streaks in ra/dec coordinates are correlated on the sky and combined into orbits of the satellites. Finally, a TDM of the satellite data is generated and saved in a file. The four stages are

3.1 Preprocessing of the image and enhancement of the contrast

Once an image is loaded from a file, we first calculate the histogram of the image in order to obtain an overview over the distribution of the intensities. In Fig. 3 an original image is shown on the left-hand side and its histogram on the right-hand side. We have chosen a rather problematic image with strong front lighting and strong reflections in the lens system so that we can show the capabilities of our software. This characteristic is typically seen in images when the full moon is present and dust accumulated on the front window of the weather proofed housing.

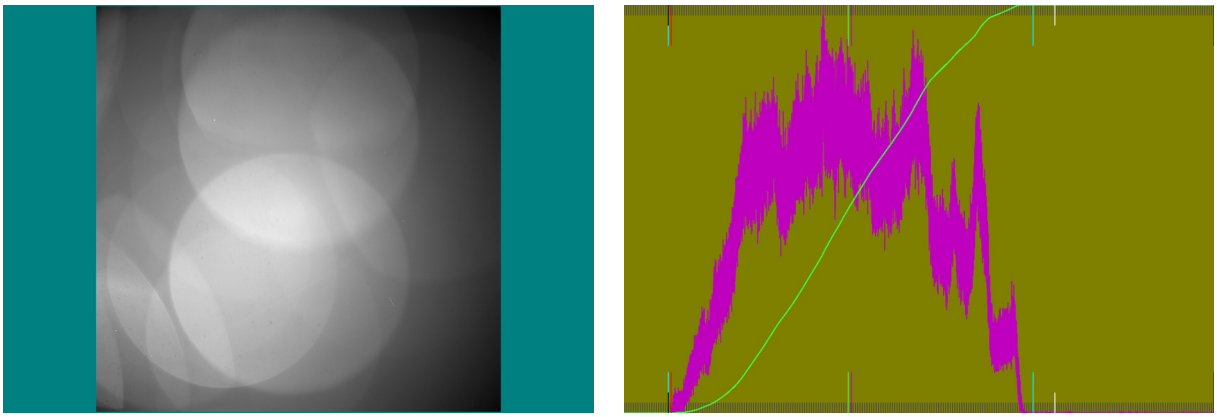

Fig. 3: An original image with strong front lighting (left) and the related histogram (right).

We use 16-bit images which means that we have 2^{16}=65536 discrete intensities from full black to full white. These 65536 intensities are shown on the horizontal axis of the histogram. The violet curve is the intensity distribution which represents the *histogram*. Since we have used maximum resolution for the binning (one bin for each of the 65536 intensities) this curve is rather noisy. A smoother curve is the integrated intensity which is shown by the green curve. It starts with value zero on the left-hand side and ends with value unity (the normalized maximum) on the right-hand side. The green curve is a staircase function. However, due to the high resolution, the stairs are not visible so that the green curve is almost continuous.

The violet curve shows a big intensity peak which represents the scattered light of the blue sky and the reflections in the lens system of the optics. A visible star or a visible satellite must have an even higher intensity than the blue sky. For this reason, their intensities must lie beyond the right side of the big peak, sufficiently far away in the right tail of the curve. Only in this case the star or satellite can be detected. We calculate the mean intensity and the median intensity of the histogram distribution function and indicate them as violet and green vertical lines, respectively, pointing from the lower boarder and from the upper boarder. Both intensities are close to each other and located near the middle of the big peak. These may be interpreted as the mean intensity and the median intensity of the scattered background light. Furthermore, we calculate two threshold values, a lower and an upper one, which mark the left end and the right end of the big intensity peak. These threshold values are indicated as light blue vertical lines left and right of the peak. They are calculated by a sophisticated extrapolation scheme from the green curve. Later the upper threshold will be used to identify the stars and the satellite streaks. Any pixel with an intensity above the upper threshold might belong to a star or to a satellite streak. All other pixels with lower intensities are excluded. The minimum intensity of the image is shown by the red vertical line, which is found on the left-hand side of the peak near the light blue line of the lower threshold. The maximum intensity is shown by a blue vertical line. Since here the maximum value is 65535, the blue vertical line is on the right border of the histogram and hence hard to see.

The 65536 values represent a rather large resolution of the intensity. If we plot an image with *black*=0 and *white*=65535, then in most cases we do not see very much because the intensity interval is too large. The main contrast of an image is restricted to a subset of the intensities. For this reason, we must select a small window of the intensities in order to enhance the contrast. We choose *black=lower threshold* and *white=upper threshold+offset* where on default we use *offset=2420*. The offset value is chosen by experience in practice. In between black and white the gray values are linearly interpolated. In the histogram the black and white values are indicated by the shorter black and white vertical lines, respectively. The image on the left-hand side is plotted using this scale for the gray values between black and white. Since the peak in the histogram is rather broad, in the image not very much is seen. A very faint streak line of a satellite can be seen if one looks from the lower

We obtain a background image if we remove all pixels which have intensities above the *upper threshold*. Then we have holes in the image. These holes are closed by an inpainting method based on a fast-marching algorithm using the intensities of their surroundings. Next, the background is smoothed by applying a box filter several times. Then, we subtract the smoothed background from the original image. In order to keep all intensities in the interval between 0 and 65535 we shift the intensities by a constant value so that the minimum intensity is fixed and the red vertical line in the histogram does not change its position. The result is shown in Fig. 4. The image on the left-hand side has much more contrast now. Stars are seen as small bright points. A satellite streak can be seen clearly 25% upward from the lower border and 25% left from the right border. In the histogram the intensity peak has become very narrow. The light blue vertical lines which mark the upper and the lower threshold are close to each other. Furthermore, the vertical black line and the vertical white line are close to each other. They mark the contrast window for the gray values from black to white. In this way the contrast of the image is considerably enhanced which is clearly seen on the left-hand side of Fig. 4. The maximum intensity is seen by the blue vertical lines far on the right-hand side of the histogram.

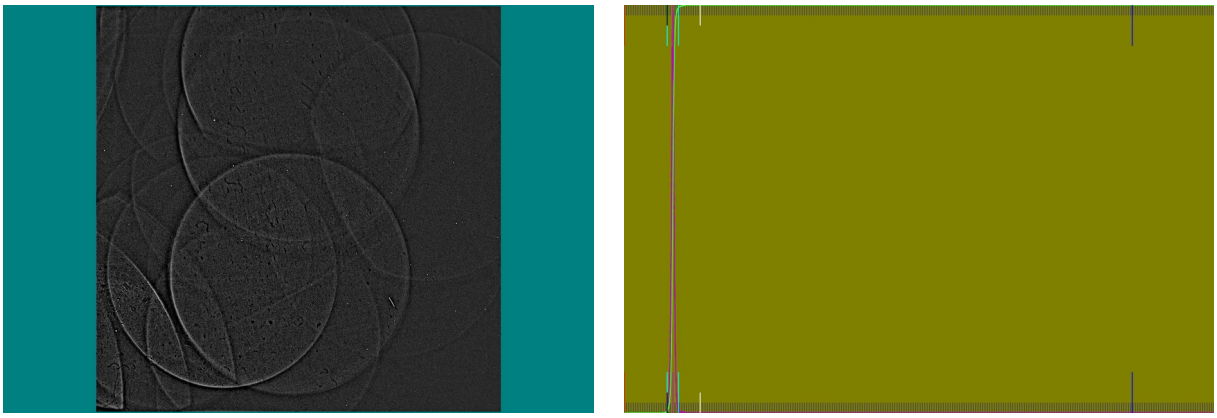

Fig. 4: A processed image with strong front lighting (left) and the related histogram (right).

We have enlarged the pixel resolution of the contrast enhanced image in Fig. 5 below. Here one clearly sees the strongly enlarged satellite streak. Furthermore, in Fig. 5 we have strongly increased the horizontal resolution of the histogram. Now, the peak of the scattered light is rather smooth and has a shape similar to a Gaussian distribution function with a small width described by a standard deviation of about only 200 intensity values. The distance of two light blue vertical lines which represent the lower and the upper threshold value is only about 1000 intensity values. The tails of the peak will be larger than the tails of a common Gaussian function. One should notice that the high intensity tail on the right-hand side represents the stars and the streak lines which we want to observe. In Fig. 5 the right light blue vertical line is the upper threshold. All pixels which have intensities above this threshold may belong to stars or satellite streaks.

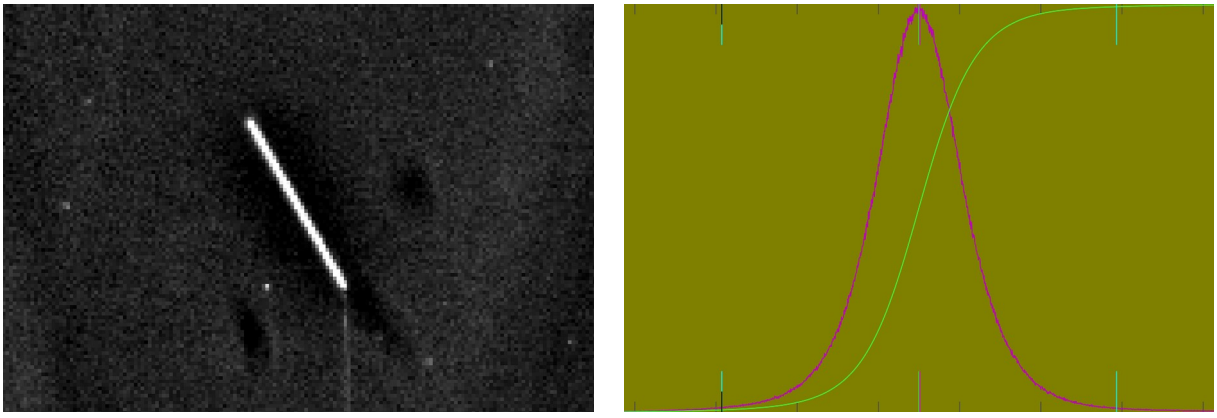

Fig. 5: Strongly enlarged: The processed image with strong front lighting (left) and the related histogram (right).

If we consider Fig. 4 we may ask the question why in the histogram the intensity peak becomes so sharp once we subtract the smoothed background image. The answer is: The sharp peak represents the scattered light of the blue sky. The width of the peak is related to the fluctuations of the scattered light and to the fluctuations of the hardware electronics. The first fluctuations are a physical effect of the atmosphere while the second depend on the measurement system. The position of the peak depends on the brightness of the blue sky. At dawn the blue sky might be rather bright so that the position is located at large intensities more right in the histogram. At deep night the blue sky is rather dark and nearly black so that the position is located at small intensities more left in the histogram. It turns out that the position of the peak does not influence our algorithm for the detection of the stars and the streak lines because the relevant criterion is due to the upper threshold which is located just right of the peak and indicated by the right light blue vertical line.

On the other hand, if we consider Fig. 3 we may ask the question why the peak is so broad for the original image. One reason is that the lens optics implies a mapping of the intensity values which depends on the pixel position within the image. Near the image center located at the optical axis the mapped intensity of the blue sky is maximum. It decays with the distance of a pixel from the optical center. This effect will broaden the intensity peak in the histogram. Another source for broadening may be inhomogeneous conditions for the scattering of light in the atmosphere. One cause might be clouds in the air. Another source for inhomogeneities and broadening may be reflections in the optical lens system which may appear strongly enhanced under front lighting conditions. The basic concept of our algorithm subtracting a smoothed background image relies on the fact that the mean intensities of the light peak are local and vary with the pixels in the image. If we subtract the local mean intensities we remove the broadening.

In our example image the broadening is very large because of the front lighting. We have chosen this worst-case example in order to show how much the subtraction of the smoothed background can improve the image and sharpen the intensity peak. Under optimal conditions when the images are taken deep in the night at clear air conditions with no clouds, the intensity peak in the histogram of an original image is much sharper so that the effect of our algorithm is much less. However, the resulting peak width in the histogram is nearly the same whether we start with original images taken under optimal conditions or taken under worst case conditions.

3.2 Detection of the stars and satellite streaks

Once the images have been preprocessed and an optimally enhanced contrast image has been generated we are ready to detect the stars and satellites. We search for all pixels which have intensities above the *upper threshold* given by the right light blue vertical line in the histogram. We then investigate whether the selected pixels form *connected* areas of certain sizes and certain shapes. We count the pixel numbers of a connected area and require a lower minimum number for a star and a higher minimum number for a satellite streak. If these requirements are not satisfied the connected areas are discarded. We calculate moments of first, second and higher order in terms of averages and correlations functions using the intensities as weights. From the first moments, we obtain the mean position of the object. From the second moments we obtain the main axis lengths and directions of an ellipsis which is fitted into the connected pixel area. If the lengths of the main axes are nearly equal and not too large, the object is identified as a star. If one main axis is very large and the other very small, then we have found a satellite streak. Investigating the second moments more closely we have several conditions with several adjustable parameters in order to identify an object as a star, as a satellite or something else. We calculate even some higher invariant moments similar to the Hu moments in order to have one more criterion. With these efforts we try to distinguish a star or a satellite from something else like an airplane or a cloud fraction.

Once we have safely detected the stars and satellites we create three new types of images from the image with enhanced contrast. In the first image we keep the stars and remove all other objects. In the second image we keep the satellite streaks and remove all other objects. Finally, in the third image we remove all objects and all pixels with intensities above the upper threshold. We remove the objects by using our inpainting method we have described above. In this way we have several images for each object class which can be watched by the user in the GUI version of the software (see Fig. 10 below for an example). Next, we calculate the positions of the stars in pixel coordinates from the first moments. The satellites appear as streaks which are short lines with two end points. We calculate the pixel positions of the two endpoints by combining our results for the first and second moments. The results from the moment calculation proved to be very precise and accurate. For the enlarged streak line in Fig. 5 we show our result in Fig. 6 on the left-hand side. There is a thin blue square which surrounds the nearly white streak line. This square is the *bounding box* of all the pixels of the streak line with intensities above the *upper threshold*. Inside the white streak line there is a red thin line which represents the results of the moment calculation. The endpoints of the red thin line represent our results for the endpoints of the

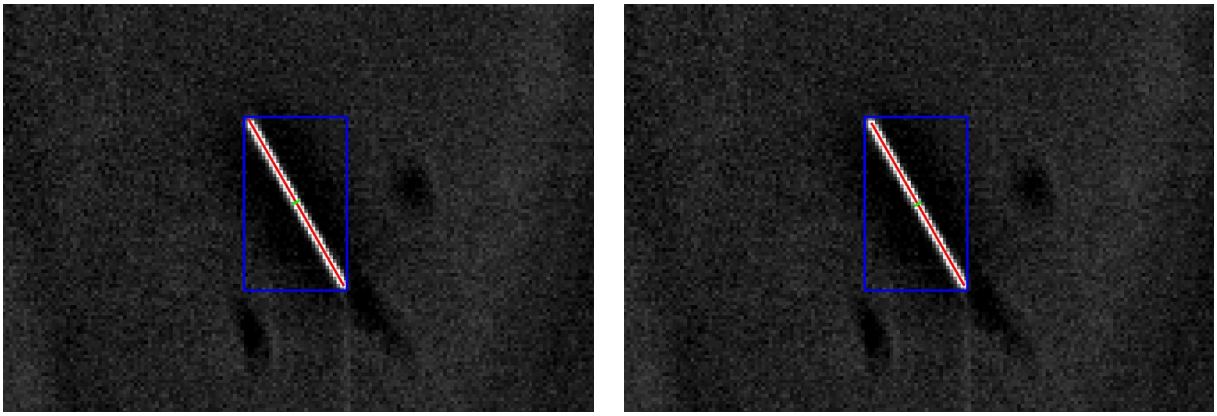

Fig. 6: A satellite found by momenta (left) and detected by adjusting a special fit function (right). One should focus on the red thin line. The changes of this line are very small from the left to the right figure.

In order to improve the accuracy of the endpoint positions, in a following step we adjust an intensity distribution function to the streak line. In transversal direction we use a Gaussian weight function. In longitudinal direction along the streak line we set the intensity to be constant. At the end points we extend the constant longitudinal distribution by half Gaussian functions in order to confine the streak line. In a minimization calculation we adjust several parameters. We have an underground intensity and an amplitude for the Gaussian functions. Furthermore, we have parameters to vary the position, the length, and the direction of the streak line. Finally, we allow a slightly curved streak line and adjust the curvature. Moreover, we have implemented a special algorithm to handle and discard outlier intensities. The adjustment works quite accurate and robust. The curvature is an additional parameter to distinguish if a streak line is a satellite or something else. The curvature must be sufficiently small and below a given maximum value so that a satellite is identified. The result for our example is shown in Fig. 6 on the right-hand side. Differences are very small between the left and the right figure so that one should enlarge the figures and look closely. One should focus on the red thin line which has slightly moved, especially in the upper left corner. Now, the positions of the two endpoints are located somewhat better and more accurately. While the effect is very small we note that we want to detect the endpoints with subpixel accuracy as good as possible. Eventually, we use these end point positions for our further calculations.

3.3 Applying astrometry to find ra/dec coordinates

Until now we have calculated the positions of the stars and of the streak-line endpoints in x/y coordinates which represent the pixel positions in the images. In the next step we must transform the positions into ra/dec coordinates of the equatorial coordinate system of the sky. For this purpose we use the software package *Astrometry.net* [13] which provides the functions *solve-field*, *wcs-xy2rd*, and *wcs-rd2xy*. First, we write the positions of the stars in the image in x/y pixel coordinates into a file. Then, we start the executable program *solve-field* with some parameters set in the GUI and with this file as an input. The program compares the measured star positions with the positions of known stars in a catalog which is read from a database. It optimizes the parameters for the transformation between the x/y pixel coordinates and the ra/dec equatorial coordinates. At the end, it calculates the ra/dec coordinates of the stars detected in the image. We note that the star positions we have detected are necessary for the function *solve-field* to find the parameters of the coordinate transformation.

Now, we have the parameters for the coordinate transformation between x/y and ra/dec for a single image. There exist two further programs of the software package with names *wcs-xy2rd* and *wcs-rd2xy*. With these programs we can transform forward from x/y to ra/dec and backward from ra/dec to x/y, respectively. Thus, we save the x/y coordinates of the endpoints of the streak lines into a file, apply the program *wcs-xy2rd*, and then read the calculated ra/dec coordinates from a new generated file. In this way, we obtain the ra/dec coordinates of the satellite streaks in the equatorial coordinate frame.

We note that here we rely on the existing software package *Astrometry.net*. We cannot influence the accuracy of and the CPU time needed for the calculations. In our test runs of the software we find that the calculation time needed for the astrometry is somewhat larger than the calculation time needed for the two preceding steps, the image preprocessing and detection of the stars and satellites.

3.4 Correlation of the objects into satellite orbits

Our space-debris surveillance system works continuously. It takes images one after another each few seconds. Whenever an image is saved in a file the image processing software is called for processing the image. In the three steps described above satellite streaks are detected, and the positions of their endpoints are calculated as ra/dec coordinates. Further data are saved like the date and the time when the picture was taken and the exposure time how long the camera was open. Thus, from the calculations up to here we obtain lots of satellite streaks with lots of data saved.

Each image observes an area of the sky with a certain diameter. This diameter represents an angle of about 15 to 20 degrees (see Tab. 2 above for FOV). A satellite moves slowly over the sky. If it is seen by the camera, it will move slowly through the observed area from one side to the other. Consequently, a satellite will be detected several times in several images. As a result, not all of the observed satellite streaks are independent. They can be put together into groups which belong to the same satellite. This grouping is done by the fourth step which is described here in this subsection.

The ra/dec coordinates parameterize the surface of a sphere onto which we map all objects observed on the sky. We assume that a satellite moves approximately on a straight line through the observed area. Thus, we may assume that an orbit is a *great circle* which lies on the sphere and on a plane which hits the center point of the sphere. The normal vector of this plane parameterizes this great circle uniquely. We calculate the normal vectors for all satellite streaks. Now, if we compare the normal vectors within certain tolerances, we obtain a *first criterion* for the grouping of the satellite streaks.

We have an additional *second criterion* for the grouping. The satellites move through the observed area approximately with a *constant velocity*. Since we know the exposure time for each image and the length of each satellite streak, we can calculate the velocity of each observed object on the sky. Furthermore, we know the positions and the times for different images so that we calculate a further velocity for the motion of a satellite from one image to the next. We check if all these velocities are equal and have the same direction within some tolerances. All those satellite streaks which have the same velocity belong to the same group.

There is a *third criterion* for the grouping. A satellite needs a finite time to move through the observed area in the sky which has a diameter of about 15 to 20 degrees. We assume that the needed time is less than 100 seconds. Thus, only those satellite streaks can belong to the same group whose times do not differ by no more than 100 seconds.

Our surveillance system operates in a continuous mode. It takes one image after the other and provides one detected satellite streak after the other. Consequently, our grouping method using the three criteria also operates continuously. Once a satellite streak is found, we check by the above criteria if it belongs to an existing group. If yes, we put it to the existing group. If no, we open a new group. Afterward we check all open groups if the time since the beginning is less than 100 seconds. If the time is larger, a group is closed so that no further satellite streaks can be added any more.

For a closed group we check how many streaks it contains. It must contain at least two streaks, otherwise this group is discarded and deleted. This check is a further test whether an object is a real satellite or not. Then, we perform the next step. We note that a valid group contains several satellite streaks belonging to a particular satellite. We calculate the satellite orbit on the sky by two optimization processes using two least squares fits to all the streaks. First, we adjust the normal vector of the plane which parameterizes the *great circle* on the sky. Secondly, we adjust a linear function along the great circle so that we obtain a *particular position* for a particular time and the *velocity* on the sky. Since we use angular coordinates on a sphere the position is an angle and the velocity is an angular velocity. While the first fit determines the transversal coordinates, the second fit yields the longitudinal coordinates of the satellite motion. In order to obtain measures for the accuracy of the fit we calculate several root-mean-square (rms) deviations. In this way we find a rms deviation for the transversal coordinates, a rms deviation for the longitudinal coordinates, a rms deviation for the direction of the orbit, and a rms deviation for the velocity on the sky.

Once the calculation is done, a tracking-data-message (TDM) is generated for each found satellite. In the comments we write the algorithm which specifies the fit function and the parameters which describe the satellite orbit. Furthermore, in the comments we save the rms values which describe the accuracy of the fit. Then, using the great circle and the linear function we calculate each position for each time in ra/dec coordinates on the sky. Thus, in the data section of the TDM for each position we write the date and the time, the right ascension (ra=ANGLE_1), the declination (dec=ANGLE_2), and the magnitude (MAG) of the object. Once the TDM is

no longer used by our computer program. Then, the surveillance system is ready to continue with the next image. This means it is ready to take and to process the next image of the sky.

In Fig. 7 we provide the results for two satellites where two orbits are determined by our circular fit process. We have processed 15 images of our worst-case example with strong front lighting and considerable reflections in the lens system. The first 6 images provide 6 streaks with overall 12 endpoints for the first satellite shown in the left figure. The last 9 images provide 9 streaks with overall 18 endpoints for the second satellite shown in the right figure. Even though the optical conditions are quite weird, we unexpectedly obtain quite accurate results for the two orbits.

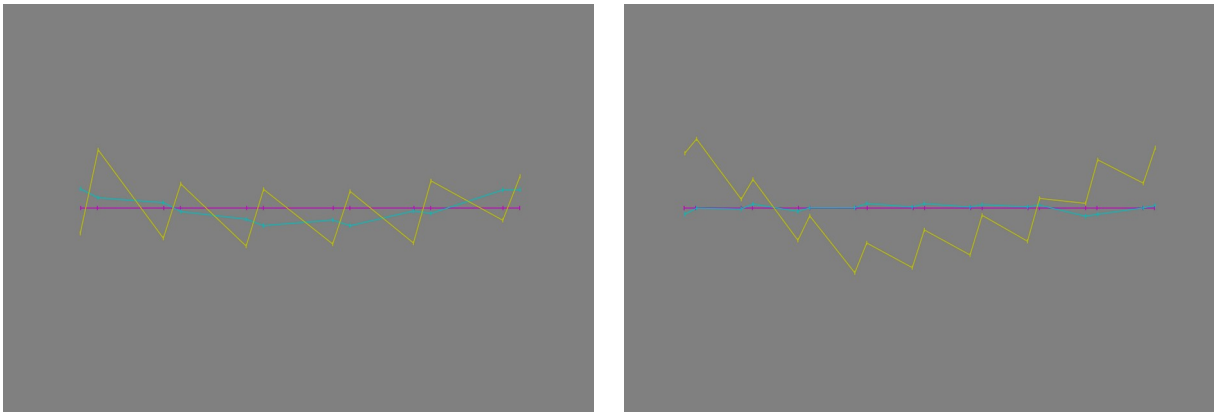

Fig. 7: Two satellites found in the image sequence with identification codes
20181201_044752_001_station (left) and 20181201_051328_001_station (right).

The horizontal direction in these two figures represents the longitudinal angle along the great circle. The lengths of the two orbits are 15 degrees and 16 degrees, respectively. The vertical direction represents angle deviations enlarged by a factor of 70. The red straight line is the fitted satellite orbit. The light blue line shows the satellite streaks observed by the camera. Since the vertical direction represents the strongly enlarged transversal angle one clearly sees that the *transversal fit* is quite good. The endpoints of the streaks are marked by short ticks on the lines. Two ticks close to each other represent one streak. One clearly sees that the distance between two streaks is about four times the length of a streak. In order to judge the accuracy of the *longitudinal fit* one should compare the positions of the ticks on the red line with those on the light blue line. The deviations are very small. We have depicted the deviations strongly enlarged in vertical direction by the yellow line. Also in this case the fit appears to be quite good. Nevertheless, we observe that the fit of the longitudinal coordinates is less accurate than the fit of the transversal coordinates. The reason may be that either the length of the satellite streaks is less well determined or the exposure time is less accurately known. We infer this from the sawtooth form of the yellow curves. Finally, we conclude that in both cases in Fig. 7 the satellite orbits are quite well determined.

4 Results

The software of our passive optical system was thoroughly tested offline and online. In the offline mode folders of different sets of images were considered and processed by the software. The results are many TDM files each of which represents an observed satellite. The system was also tested in online mode. Here each few seconds an image was taken from the sky and transferred immediately to the software which then processed the images, detected satellites, determined the orbits and finally wrote the results into TDM files. The found satellites were then compared with known satellites which are catalogued in databases. Here we can compare the observed orbits with known orbits from the databases. As a result, we find good agreement. Thus, we conclude that our passive optical system can determine orbits of satellites with appropriate accuracy.

We have tested the software with different kinds of images. In chapter 3 we have considered a worst-case example in order to show the capabilities of our software. Here we present some more examples. First, we have selected a best-case example which is shown in Fig. 8. This is an image taken at night under clear air conditions.

lower threshold intensity and the white color for the upper threshold intensity plus an offset value which is chosen as 2420. The gray values interpolate the intensities linearly in between. The procedure is the same for all images shown in this paper, including Figs. 3, 8, and 9. In the left-hand image of Fig. 8 many stars are seen, and a quite long streak line of a satellite. The right-hand figure shows the processed image where the stars have been removed and the streak line has been detected. The blue rectangle is the bounding box of the satellite pixels which have intensities above the upper threshold value. The red line marks the streak line. The result appears to be quite good with subpixel accuracy. In the original image (left) one sees a higher intensity of the background light in the center where the intensity decreases towards the boundaries. In the processed image (right) this effect has been removed by the subtraction of the background image.

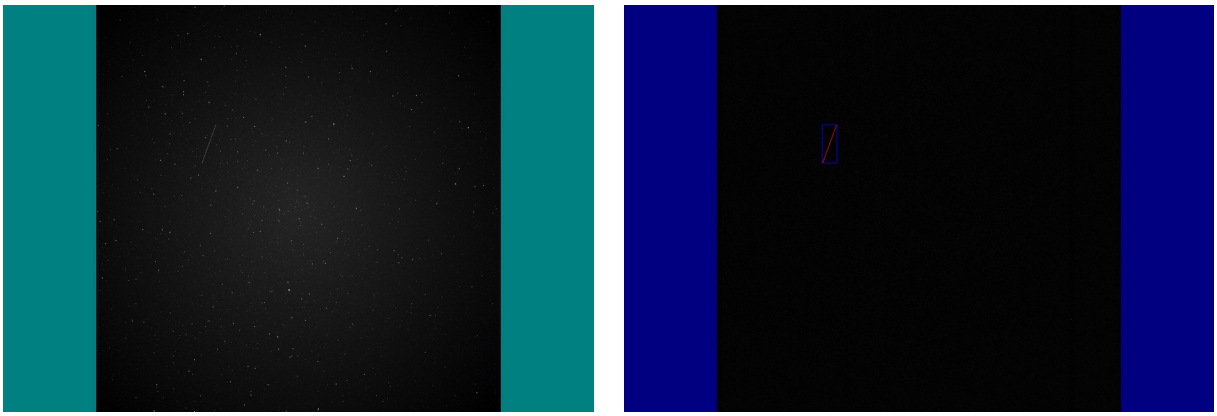

Fig. 8: Image taken under optimal conditions at night with clear sky. Left is shown the original image. Right is shown the processed image with the background subtracted, the stars removed, and the satellite detected.

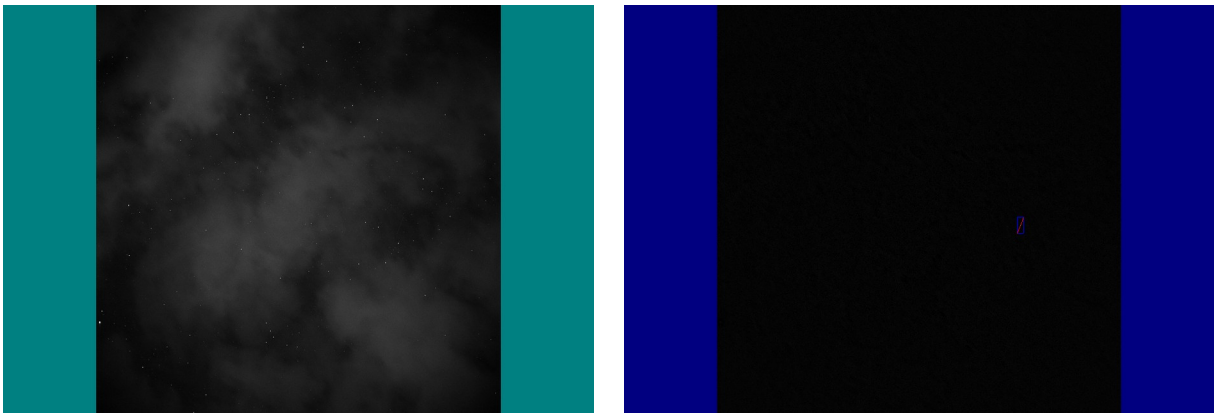

Fig. 9: Image taken at night with cloudy sky. Left is shown the original image. Right is shown the processed image with the background subtracted, the stars removed, and the satellite detected.

As a second example we have chosen an image of a cloudy sky. This is shown in Fig. 9. In the original image (left) one clearly sees the clouds. On the other hand, in the processed image (right) the clouds are removed by subtracting the smoothed background. Also, the stars are removed by our inpainting procedure. Eventually, there remains one streak line which has been detected successfully. This streak is indicated by the red line in the middle of the image somewhat to the right. The blue square represents the bounding box of the streak pixels. Also, in this case the satellite has been detected robustly with subpixel accuracy. One should note that in the original image with clouds (left) the streak line is present but very hard to see.

In an image there might appear artefacts generated by reflections of the lens system and objects other than satellites which we do not want to detect. As artefacts there occur reflexes which are vertical lines down from a star. Such an artefact might be detected as a satellite streak. However, in order to identify a real satellite and determine its orbit, there must be detected streaks in two or more pictures which fit together into an orbit. Here

we check that the streaks lie on a great circle in the sky and that the satellite moves with a constant velocity. For reflexes this check usually fails so that they are robustly excluded. Another type of objects are airplanes flying in the sky. They move similar like satellites and hence are more difficult to distinguish. Usually, an airplane has flash lights or shows several parallel traces in the sky. In most cases, the calculation of the higher moments provides a criterion which enables to identify and to discard airplanes. However, the method is not so robust and safe. Airplanes are not discarded absolutely in all cases.

A satellite may rotate and hence change its brightness periodically with time. For the streak lines this means that the intensity may not be constant. If we calculate the higher moments our criterion will then discard those satellites. It is hard to treat this case, and our software does not have a solution here. A further difficulty arises for stars which may lie close to a streak line. It may happen that the pixels with intensities above the upper threshold form a connected area which includes both the streak line and the star. If we then calculate the higher moments, our criterion will exclude this streak line. In principle, this fact should be no problem. Usually, for a satellite there are detected streaks in several images. If one of the streaks is discarded, the remaining streaks of the other images are still enough to find the orbit and to generate the TDM file. In order to see this one should look into Figs. 7 left and right and think what happens if one streak line is taken out.

On the next page, in Fig. 10 we provide one representative example of a streak line with a nearby star. The images in Fig. 10 are strongly enlarged so that we can see the details. Here the criterion with the higher moments and with the width of the line have been weakened so that this streak line was not excluded. In the top-left figure the original image is shown, similar like the left-hand images in Figs. 3, 8, and 9. One clearly sees the streak line and a nearby star. In the top-right figure the processed image is shown. Here, the contrast is even somewhat better. The following images are then obtained from this processed image. The middle-left figure shows the background image where all objects, stars, satellites, and artefacts, have been removed by our inpainting method. In the middle-right figure the stars are detected by crosses of red and green lines. The blue rectangles are the bounding boxes of the pixels with intensities above the upper threshold. The streak line has been removed by our inpainting method. The star positions are then used by the function solve-field of the software package *Astrometry.net* [13] in order to determine the transformation between the x/y pixel coordinates and the ra/dec equatorial coordinates. The bottom figures left and right show the streak line where the stars are removed by our inpainting method. The blue square again represents the bounding box of the pixels with intensities above the upper threshold. In the bottom-left figure the coordinates of the streak line are obtained by the momentum calculation. The result is shown by the red line. Here one clearly sees that the nearby star causes wrong moments so that the red line deviates from the streak line. Also, a larger thickness of the streak is found which is indicated by the short perpendicular green line. In the bottom-right figure a model intensity function is adjusted to the streak line in an iterative process where the results of the momentum calculation are used for the initial values. Here the bright intensity pixels of the nearby star act as outliers and are discarded in the iterative adjustment process. As a result, we obtain a read line which lies perfectly on the streak line and a short perpendicular green line which indicates the thickness correctly. In this way we obtain good results for the end-point coordinates with subpixel accuracy. However, we note that this iterative procedure is not robust and does not work in all cases. For these reasons, it is better to discard streak lines with nearby stars by the momentum criterion.

While for a satellite we calculate the positions observed in the streak lines as shown in Fig. 7, we also determine mean values for the parameters of an orbit. Additionally, we calculate root mean square (rms) values for the deviations in order to estimate the accuracy of an orbit. The results are shown in Tab. 3 below. First, we calculate the angle difference for the observed orbit length. Our images cover up to 20 degrees on the sky so that the observed orbit lengths of about 15 degrees on the sky are reasonable. Secondly, we calculate the angular velocity which indicates how fast the satellite moves on the sky across the FOV. Furthermore, we present the related rms values for these two longitudinal results. The transversal rms deviation indicates the deviation from the great circle in the sky. Finally, we provide the rms deviation of the direction of the orbit. All values are calculated for the two orbits shown in Fig. 7 left and right.

	Orbit of Fig. 7 left	Orbit of Fig. 7 right
Longitudinal angle difference (observed orbit length)	14.852983 deg	15.951709 deg
Longitudinal angular velocity (mean value)	0.562817 deg / sec	0.389015 deg / sec
Longitudinal angle deviation	0.015638 deg	0.020672 deg
Longitudinal angular velocity deviation	0.001805 deg / sec	0.001599 deg / sec
Transversal angle deviation	0.006262 deg	0.001989 deg
Deviation of the orbit direction	0.073568 deg	0.022663 deg

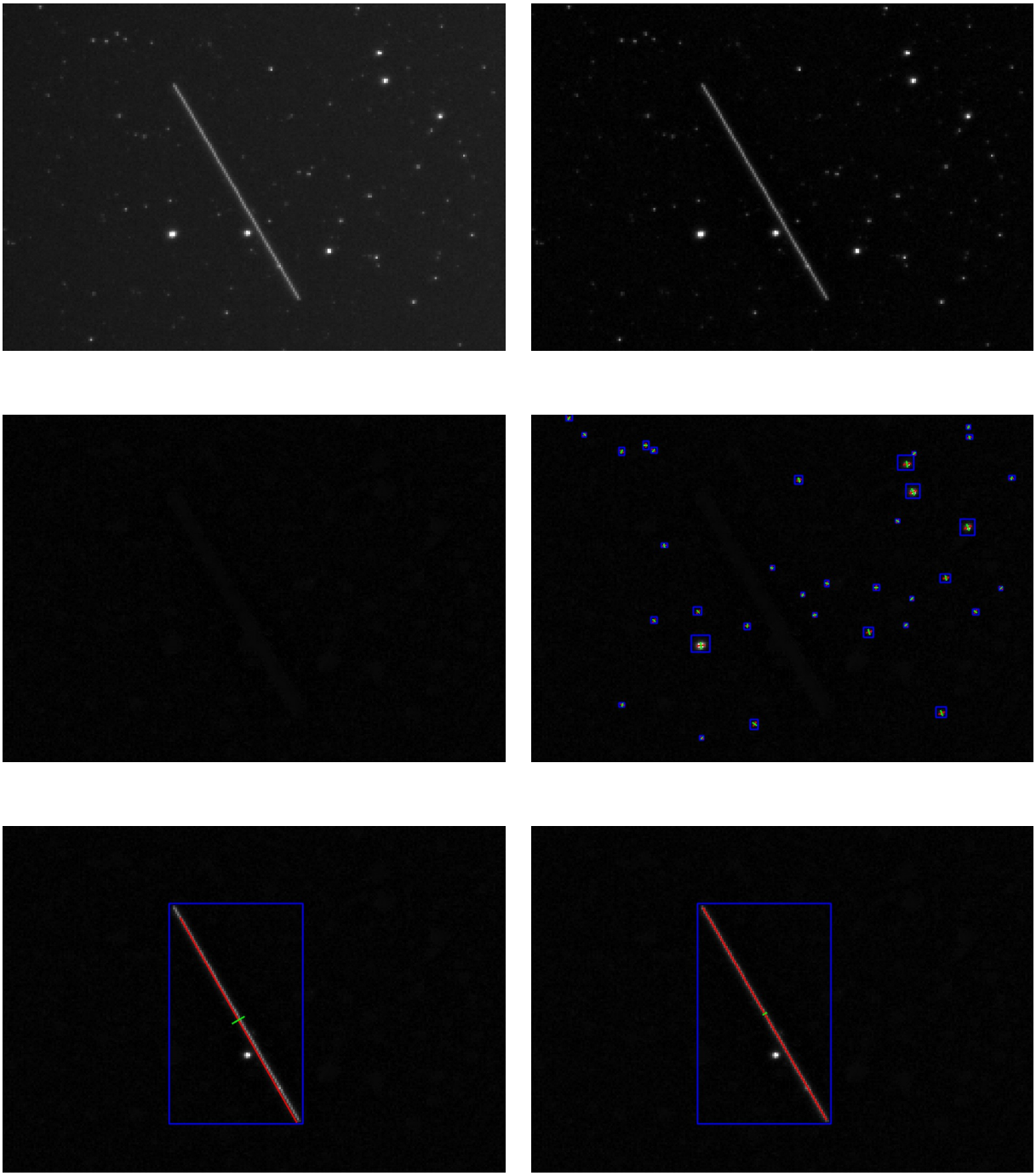

Fig. 10: Image taken at night with clear sky. Nearby the satellite streak there is a star which is too close so that it cannot be separated and hence disturbs the calculations. The image is strongly enlarged in order to see the effect. Top left: original image; top right: processed image; mid left: background, all objects are removed; mid right: stars detected; bottom left: satellite detected by moment calculation, bottom right: result improved by adjusting intensity function.

5 Conclusions

A reliable image processing chain was demonstrated which is currently deployed in the APPARILLO ("Autonomous Passive Optical Staring Of LEO Flying Objects") system. The system successfully demonstrated its first campaign between November 20th to December 23th 2020 using the presented algorithms. During this time, no false positive detections occurred during the operation period [4]. The presented image processing robustly handles distractions in the images and allows streak detection even when lens flares or any kind of

well controlled by the iterative processes. Furthermore, the TDM data export allows interchanging and processing the data easily.

6 Acknowledgments

We would like to thank W. Riede and Prof. Dr. T. Dekorsy for helpful discussions and comments on the manuscript. Parts of the presented system for passive optical staring in combination with subsequent laser ranging was protected under the utility patent DE 20 2017 101 831 U1 [14].

7 References

[1] Bamann, C., Hugentobler, U.: Accurate orbit determination of space debris with laser tracking. Proc. 7th European Conference on Space Debris, Flohrer, T. and Schmitz, F. (eds.). (2017)

[2] Wagner, P., Hampf, D., Riede, W.: Passive optical space surveillance system for initial LEO object detection. In Proceedings of 66th International Astronautical Conference, IAC-15.A6.3x28491 (2015)

[3] Hasenohr, T., Hampf, D., Wagner, P., et. al.: Initial Detection of Low Earth Orbit Objects through Passive Optical Wide Angle Imaging Systems, Proc. of Deutscher Luft- und Raumfahrtkongress 2016 (2016), ´, Accessed 18 December 2020

[4] Wagner, P., Clausen T.: APPARILLO: A fully operational and autonomous staring system for LEO debris detection. CEAS Aeronautical Journal (submitted Feb. 2021)

[5] Finger Lakes Instruments: ProLine 09000 data sheet. http://www.flicamera.com/spec_sheets/PL09000.pdf (2018). Accessed 02 August 2018

[6] Vincent Associates: Uniblitz CS65 datasheet. https://www.uniblitz.com/wp-content/uploads/2017/01/uniblitz-cs65-datasheet.pdf (2018). Accessed 15 March 2021

[7] Hampf, D.: GPS camera timer. Internal manual (December 15, 2016)

[8] CCSDS: Tracking Data Message - Recommended Standard, CCSDS 503.0-B-2, Blue Book, June 2020. https://public.ccsds.org/Pubs/503x0b2.pdf. Accessed 23 March 2021.

[9] Hampf, D., Sproll, F., Hasenohr, T.: OOOS: A hardware-independent SLR control system. In ILRS Technical Workshop, Riga (2017)

[10] Qt 5 is a software package for programming GUI applications on any platform. The website is https://www.qt.io. Accessed 7 March 2021.

[11] The software packages cfitsio and CCfits provide functions for reading the FITS images. The website is https://heasarc.gsfc.nasa.gov/fitsio. Accessed 7 March 2021

[12] OpenCV team: OpenCV Website. https://opencv.org/ (2020). Accessed 24 December 2020.

[13] The software package Astrometry.net provides functions for astromety calculations. The website is https://astrometry.net. Accessed 7 March 2021.

[14] Gebrauchsmusterschrift, DE 20 2017 101 831 U1 "System zur Bestimmung und/oder Vorhersage einer Position und/oder einer Flugbahn von orbitalen Objekten im Weltraum", Wagner, Paul; Hampf, Daniel; Sproll, Fabian; Humbert, Leif; Hasenohr, Thomas, (29.03.2017), https://patents.google.com/patent/DE202017101831U1/en (https://register.dpma.de/DPMAregister/pat/PatSchrifteneinsicht?docId=DE202017101831U1), Accessed 22 January 2021.