

Data-parallel methods for fast and deep detection of asteroids on the Umbrella platform: near real-time Synthetic Tracking algorithm for near-Earth objects

Mălin Stănescu^{1,2}, Marcel M. Popescu^{3,1,2}, Lucian Curelaru¹, Ovidiu Vaduvescu^{3,4}, Daniel Bertesteanu^{1,2}, and Marian Predatu³

¹ Astronomical Institute of the Romanian Academy, 5 Cuțitul de Argint, 040557 Bucharest, Romania

² Astroclubul București, Blvd Lascăr Catargiu 21, 10663 Bucharest, Romania

³ The University of Craiova, Str. A. I. Cuza nr. 13, 200585 Craiova, Romania

⁴ Isaac Newton Group (ING), Apt. de correos 321, E-38700, Santa Cruz de La Palma, Canary Islands, Spain

Received xx mm 2024 / Accepted DD Mmmmm YYYY

ABSTRACT

Context. Further activities in detecting Near-Earth asteroids (NEAs) using the blink method are hampered by the required size of the telescopes. Synthetic Tracking (ST) is an effective solution, but the computational demands make operation difficult at survey data rates for fast moving objects.

Aims. We aim to show that, through efficient use of hardware, judicious pipeline design and choice of algorithms, ST at survey levels of data rates is possible even for objects with very fast apparent motion, such as NEAs.

Methods. We developed a GPU-accelerated Synthetic Tracking pipeline, called Synthetic Tracking on Umbrella (STU), targeting real-time detection of fast NEAs through algorithms designed to make efficient use of data-parallel hardware. STU was developed as part of the Umbrella software suite, which we have expanded to provide an end-to-end data reduction pipeline.

Results. We demonstrated the capabilities of the STU pipeline to scan for moving objects faster than the acquisition rate at search radii of 10 arcseconds per minute, with good detection rates on several archival datasets without specific tuning.

Conclusions. Our investigation shows that Synthetic Tracking is viable for large scale surveys and that STU may perform such a role.

Key words. Minor planets, asteroids: individual: – techniques: photometric – methods: observational – methods: numerical

1. Introduction

Surveys targeting the discovery of near-Earth objects (NEOs) are of significant importance in astronomy and for human life. While current surveys have discovered most NEOs larger than 1 km and have made significant progress in the detection of asteroids larger than 100 m, smaller objects tend to be discovered only during close flybys (e.g. Popescu et al. 2023).

Synthetic Tracking (ST, but also known as Digital Tracking Allen et al. (2002)) is a computational technique for detecting minor planets by combining multiple images of the same region of the sky taking into account the apparent motion of a potential object across those images. By aligning and stacking the images along potential motion paths, synthetic tracking enhances the signal-to-noise ratio (SNR) of moving objects, enabling the detection of objects too faint to be detected in a single exposure.

With the exception of a few isolated cases (Yanagisawa et al. 2021), most minor planets have been found with the blink method, which involves comparing temporally separated images of the same region of the sky to detect objects with an apparent motion relative to the background sources. This can be done either using difference images, such as the software described by our previous paper (Stănescu & Văduvescu 2021), or using difference of detections, such as (Chambers et al. 2016), or other comparison techniques used by different Moving Ob-

ject Processing System implementations (such as Denneau et al. (2013)). The main reason for the widespread use of blink is that methods such as Synthetic Tracking are regarded as requiring vastly more computational power to analyze all possible object trajectories. This is particularly true for NEOs, since the apparent velocity of such objects is much higher than for other types of minor planets, even when using hardware acceleration (such as the works by Shao et al. (2014), Whidden et al. (2019) and Yanagisawa et al. (2021)).

The blink method has several disadvantages compared to synthetic tracking, due to its brightness requirements. At modern discovery magnitudes, $V > 21$ mag, due to multiple factors, new objects can be discovered using the blink method only with telescopes that have a large aperture ($\gg 1$ meters). First, skyglow becomes an important hindrance at these magnitudes. Even on dark nights (i.e. without the presence Moon on the night sky), for large telescopes or good cameras (and especially when both are combined), airglow becomes the dominant source of noise (larger than the noise from image sensor — readout or dark current). Furthermore, faster asteroids streak across the image and therefore cover more of the sky area with their source spread. As the flux of the noise through an area (of uncorrelated noise sources) increases with the square root of the area, the SNR of the streaking asteroids drops even for algorithms that take streaking into account (by integrating along the potential motion vectors). As such, the blink method yields diminishing returns in detection of new objects.

Send offprint requests to: Mălin Stănescu, e-mail: malin.stanescu@gmail.com

In comparison, synthetic tracking, directly addresses two of the above points: streaking (through integrating many short exposures) and the requirement for large telescopes (through its lower brightness requirements, as shown in Yanagisawa et al. (2005)). Indirectly however, it opens up the possibility of using other types of telescopes for this task, such as orbital ones (as proposed in Shao et al. (2014)) or even stratospheric ones (which, due to the tracking difficulties are useful mostly for short exposures). As such, synthetic tracking can easily detect new objects at much lower magnitudes compared to blink methods.

In this paper we show that a careful implementation of the synthetic tracking concept can be sufficiently fast for real-time detection of fast NEOs (Section 3), present the software framework over which it was developed (Section 2) and discuss the early results from applying it to several archival datasets (Section 4). We finish with the Discussions and Conclusions of the results of our software implementation as well as their potential impact on the discovery of minor planets. The software implementation described in this paper is currently at Technology Readiness Level (TRL) 4¹.

2. From raw images to astrometric reports: the Umbrella software suite

A complete system for automated search and reporting of asteroid discoveries involves many steps and software packages, starting from the firmware that controls the motors of the instrumentation (used in the mount, optical path, instrument auxiliaries, etc.) to the web interface for detection validation. Due to its large extent, to keep such a system maintainable, it needs to be split hierarchically into multiple components. To maintain high on-sky efficiency, each of these components, including the system itself, has to operate as pipelines. In such a system, top level components may be scheduling, motion and instrument control, data reduction, validation and reporting.

The main focus of the EUROpean Near Earth Asteroids Research (EURONEAR)² group (Văduvescu et al. 2015) in the past few years has been the development of data reduction pipelines. We have approached this through activities including the first Umbrella version (Stănescu & Văduvescu 2021), the NEARBY online platform (Gorgan et al. 2019), the ParaSOL³ project (e.g. Popescu et al. 2023), and other derivative projects (e.g. Popescu et al. 2021; Kruk et al. 2022; Prodan et al. 2024). The data reduction pipelines developed in these projects extend from the handling the raw images acquired by the telescope, all the way to the human validation of detections, with plans to extend up to automated reporting of the astrometric positions to the Minor Planet Center database⁴ (MPC).

For this purpose, in the 4th version of the Umbrella software suite (continuing the work in Stănescu & Văduvescu (2021)), we have implemented the concept of pipelines, which help building modular, user-configurable and reusable data reduction and validation software components. The system architecture of a complete deployment of the Umbrella software suite is shown in Fig. 1.

2.1. Pipeline design

A pipeline is characterized by two critical parameters which capture the success of the process: the throughput and the latency. In the case of asteroid detection, the former gives the survey coverage (i.e., how much of the sky can be observed and have the data processed in a given time), while the latter is the lower bound on the time it takes to get an alert to the community after the images of the asteroid have been acquired. The throughput can easily be scaled horizontally by adding more computational resources. However, the latency cannot be improved without improving the parallelism over individual fields and/or reducing the computational load of each field.

The importance of latency in the detection pipelines cannot be understated. If the detection results are not available in a timely manner, recovery from the same survey program in the same night may become unfeasible. For the fastest moving objects, this can lead to inability to recover on a follow-up observation if the object is not posted on the NEO confirmation page or otherwise shared with the community, which may happen for various reasons, including sufficient evidence of the detection being real.

The following terminology is used in this paper and by the Umbrella software suite:

- The **Pipeline elements** are independent modules that may be loaded and assembled into a detection pipeline or a raw image reduction pipeline. Each element has a certain type (such as measurement element, filtering element, pairing element, reporting element, etc.), and can be plugged into a pipeline assembly into its corresponding role. Certain parts of the pipeline (such as measurement and filtering) allow applying multiple elements in succession.
- A **Pipeline host** is an executable that can load the Umbrella pipeline elements, configure them according to a specified user configuration, and launch the pipeline over a set of inputs.
- **Raw science images (RAWSCI)** are the science images as received from the image acquisition software as FITS files.
- **Bias, flat, and dark (BFD)** are calibration images typically acquired at the beginning or at the end of the observing night. These images are used to correct for systematic errors of the imaging sensor.
- **Reduced science images (REDSCI)** are the images obtained by applying the instrument-specific corrections, such as bias/dark current, flat field, bleeding/blooming (due to saturation from bright stars), crosstalk (due to amplifiers needed to read faster multi-chip cameras), and astrometric corrections (due to optical field distortions).

Below we describe the Webrella interface, the IPP and the reporting module and formats. The STU module is described in section 3, as it represents a key part of our software suite.

Finally, it should be mentioned that we have ported the reference blink pipeline from Stănescu & Văduvescu (2021) to the new Umbrella v4 pipelines. We have done so particularly to use the filtering methods developed as part of it. However, the effectiveness of synthetic tracking has obsoleted the blink implementation for us and the blink pipeline is no longer maintained.

2.2. Image pre-Processing Pipeline (IPP)

We developed the IPP to apply the image corrections necessary for the instrumentation used, allowing us to use customized algorithms to achieve the REDSCI images with the best properties

¹ see https://www.nasa.gov/pdf/458490main_TRL_Definitions.pdf for the definitions of each TRL level

² <http://www.euronear.org/>

³ <https://planet.astro.ro/ParaSOL/>

⁴ <https://www.minorplanetcenter.net/>

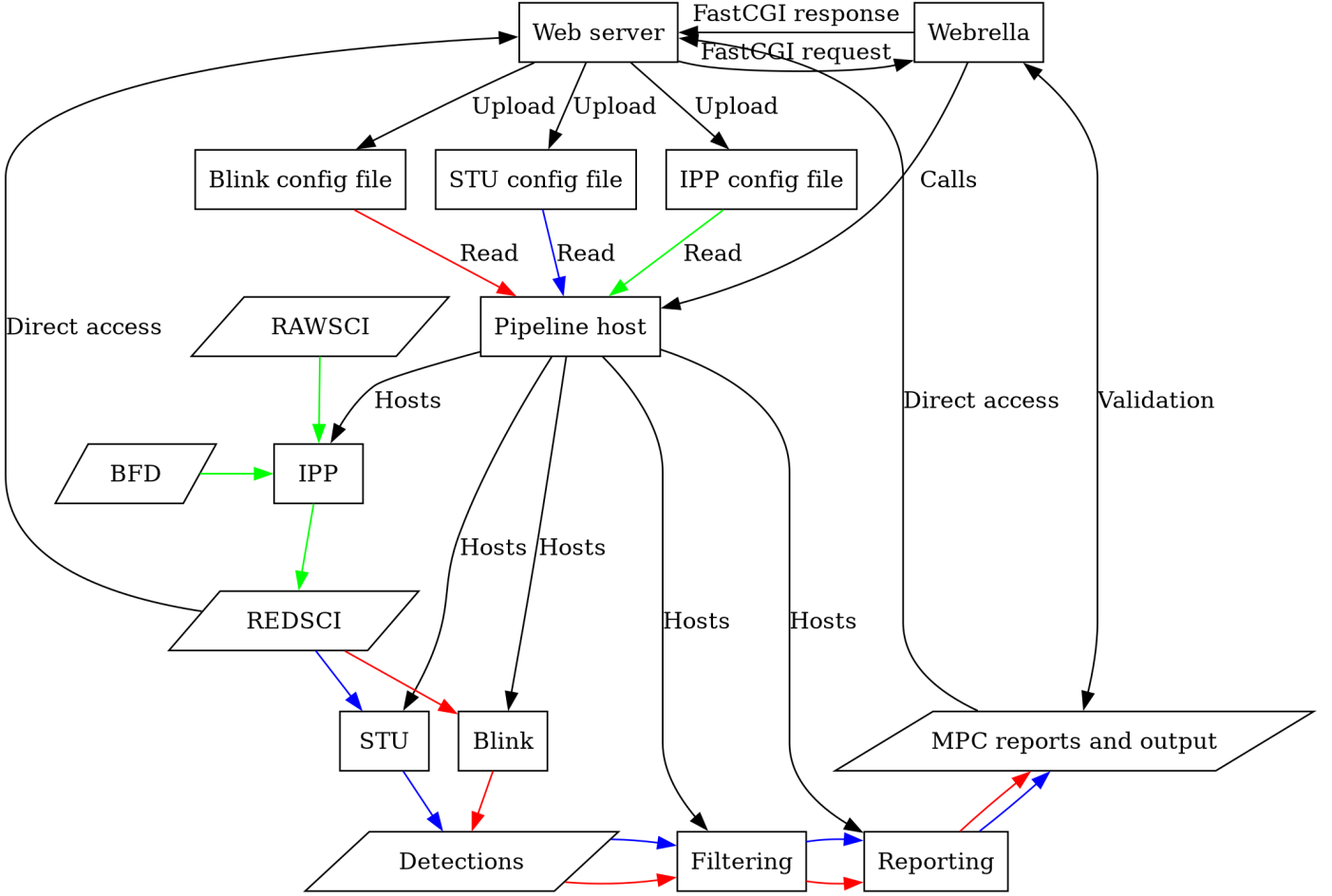


Fig. 1: **Full system architecture.** The various blocks (and their abbreviation) are described in the text. The methods to connect them are shown over each wire. Color coding of the data flows refers to the pipeline type, as all pipelines run as instances (processes) of the pipeline host: blue for STU, red for the reference blink pipeline, green for IPP.

for the Umbrella detection pipelines. IPP includes corrections due to the sensor electronics (bias, dark, bad pixel) and telescope/instrument optics (flat field and optical distortion). Several additional scripts were implemented to correct the crosstalk and bleeding defects of the CCDs used in some of the telescopes.

Prior to the ParaSOL project, the IPP was an independent software package implemented on top of the Umbrella2 library. As part of the ParaSOL, we ported this software package to the pipeline system. The same steps, shown in Fig. 2, apply to both versions of the pipeline (legacy and new). However, the broader architecture view brought by this project has resulted in a refactoring of the way the steps are applied. Due to the target of real-time detection and reporting and the extremely fast resulting detection pipeline, the first step, of pre-processing the raw images, is a major contributor to the throughput and latency characteristics of the complete pipeline.

Starting from its original design, IPP works in a multi-threaded approach over single images. As part of the latency focus of ParaSOL, the processing is now split into per-session operations and per-field operations. This improves the efficiency both in terms of throughput (the per-session data is computed only once in each session, i.e. observing night) and latency (only per-field data has to be computed once the telescope is in science mode, resulting in much shorter processing times).

To further reduce latency, the IPP was internally refactored to apply corrections to individual images as they are streamed

into the system, rather than processing them in batches after all of them are received. This enables the synthetic tracking pipeline to begin processing almost immediately after receiving a single image, significantly minimizing delays. In fields with many images (a common scenario in synthetic tracking), this approach reduces pre-processing time from minutes or even tens of minutes to just a few seconds.

2.3. Webbrella

The Webbrella software package has been introduced in Stănescu & Văduvescu (2021), where it provided a basic web interface to the blink pipeline. The quick growth of the Umbrella software, especially once the pipeline system has been developed, and the development of the IPP quickly obsoleted the version described before. However, the proven utility of the web interface for the detection pipelines, including the accessibility for users that no longer needed powerful hardware to validate detections, as well as the ease of collaboration, led us to develop a majorly overhauled version.

The new Webbrella now is designed with the Umbrella v4 ecosystem in mind and allows interacting with the multiple pipelines supported by it. The interaction of Webbrella with other components is shown in Fig. 1.

The organization of the interface has been improved to better handle re-reductions common when experimenting with dif-

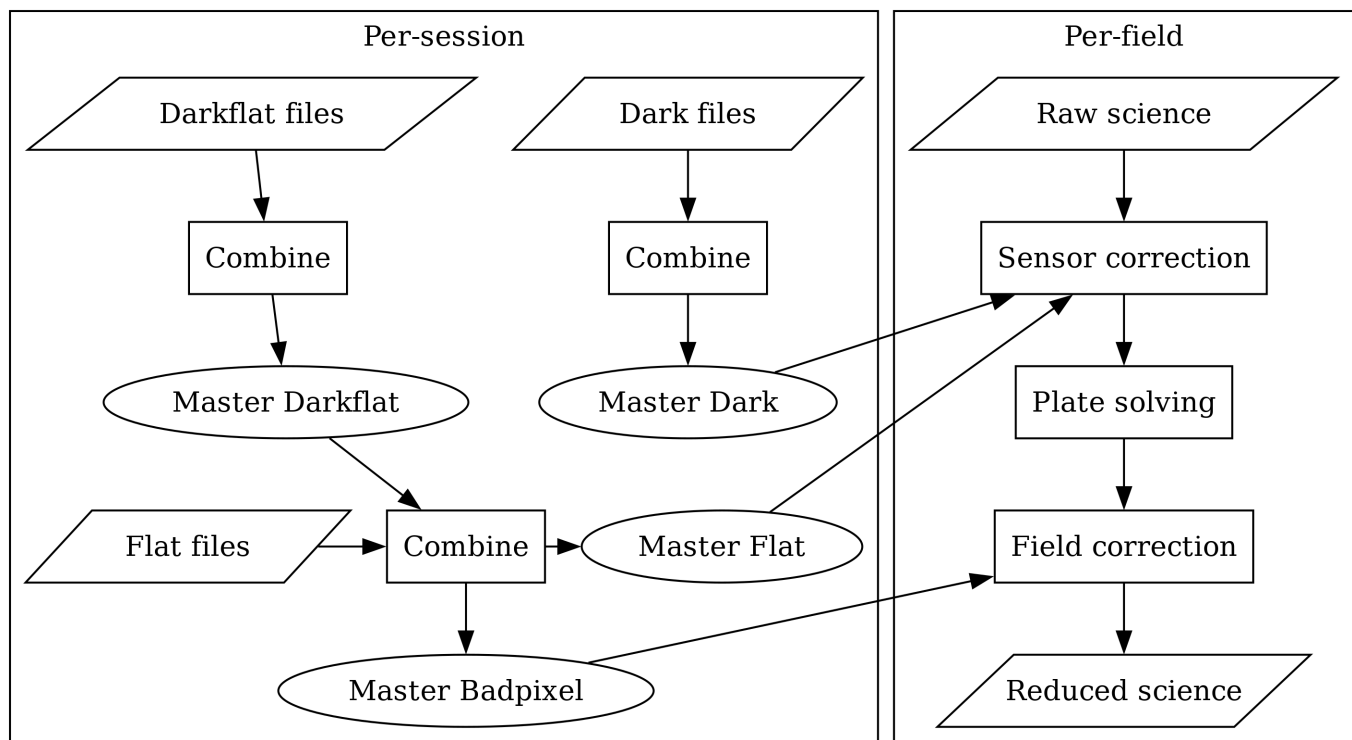


Fig. 2: IPP internal steps.

ferent observing strategies, while keeping the convenient fast-path for well-established survey configurations. The final major overhaul was the development of a permissions system for the Webrella users, to allow survey administration over its web interface, while still allowing unprivileged reducer access, as well as scoped access to experiment data.

Moreover, the new web interface went through a major refresh, taking into account the major blockers in the previous interface, while keeping the strong points of its predecessor. A screenshot of the field validation interface is shown in Fig. 3. We preserved the characteristic light-weight resource usage and low-latency response time. We used JavaScript judiciously for several key features: keyboard navigation, previously available only in the desktop interface, automatic blinking of detection stamps, copying to the clipboard the detection report in 80-column format and stamp scale configuration.

On the server side, Webrella is now a FastCGI application⁵, instead of having an in-process web server. This allows it to be used with dedicated web server software that can handle tasks not requiring content generated on the fly. Moreover, for web servers that support such mechanisms, it allows using the authorizer role to gateway access to experiments based on the user's permissions for the given experiment.

2.4. Reporting

For a detection pipeline to be useful, it must provide an effective and compatible output. The Umbrella v4 pipeline system has an internal, in-process representation of the detections. This allows for efficient processing within Umbrella, but it requires a report-

ing pipeline element to export it to a format that can be used by external systems.

The reporting pipeline element is designed to accept both graphical interfaces and batch operation. However, the format of the tests done as part of the ParaSOL project resulted in a focus on the batch operation, with the results of the detection pipeline stored in files on disk.

The following formats have been implemented in the Umbrella tooling: **MPC 80-column**, which although is a legacy format, it is still widely implemented in many asteroid detection tools. Our reporting tools output this format optionally from the batch reporting pipeline element and it is also available from the Webrella interface.

Comma Separated Values (CSV) file is used for storing the measured properties of the detection candidates. However, the focus here is to add these properties to the IPEF format described below.

Astrometry Data Exchange Standard⁶ (ADES) is the new format for storing asteroid detection information as accepted by MPC. The Umbrella software suite support reading and writing in the XML format of ADES, although our framework is not fully compliant yet, with not all the required information being available when generating the report (for example, the astrometric and photometric calibration is done by the pre-processing pipeline, while the reports are created by the detection pipeline) and the ADES character limits not being implemented. The batch reporting pipeline element optionally outputs in this format and it is also the format available for download from Webrella.

Inter-Pipeline Exchange Format (IPEF) is an internal format of our software. It is derived from ADES with the purpose of

⁵ <https://web.archive.org/web/20160119141816/http://www.fastcgi.com/drupal/node/6?q=node%2F22>

⁶ https://minorplanetcenter.net/iau/info/IAU2015_ADES.pdf

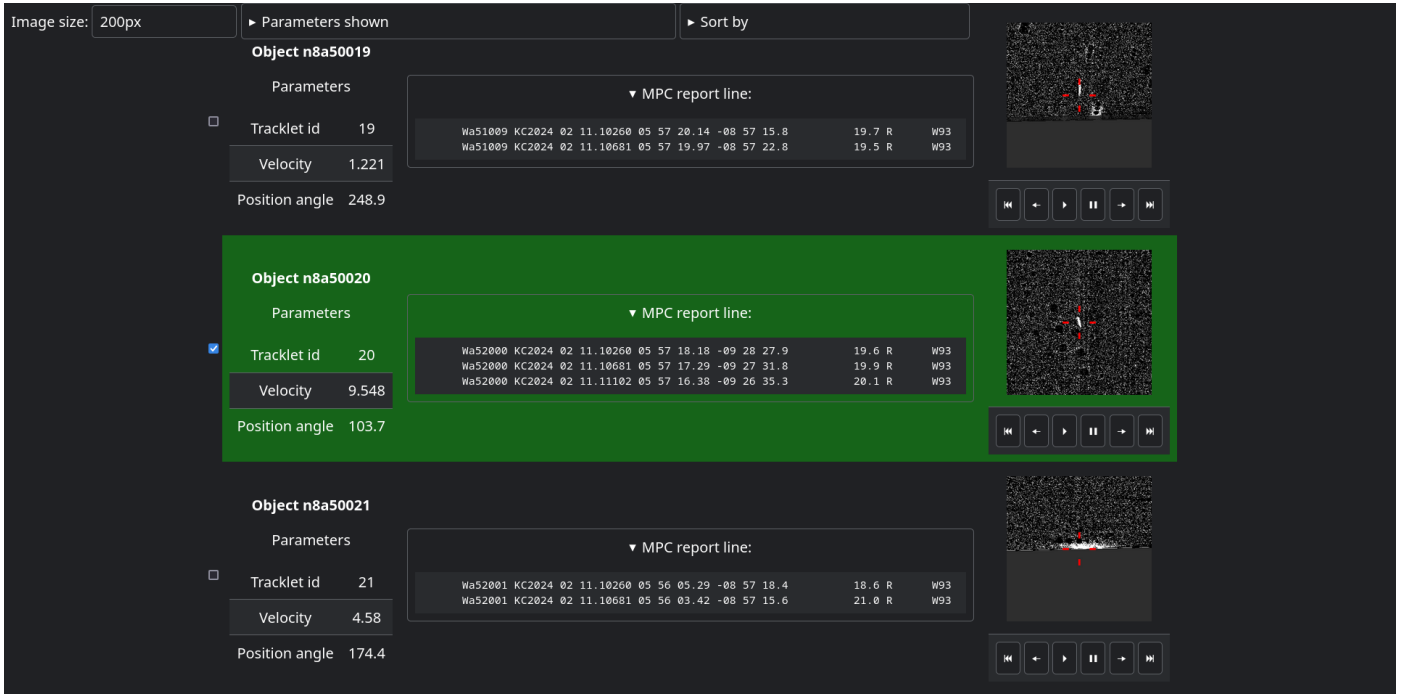


Fig. 3: **Webrella** interface, showcasing 2024 CW, a Near Earth Asteroid discovered with STU at a $9.5''/\text{min}$ proper motion. Validated objects are highlighted in green.

storing image and detection candidates properties and metadata for exchange between the image processing pipelines, detection pipelines, non-Umbrella candidate filters, as well as human validators and validation tools. This is the main format supported by the batch reporting pipeline. All other candidate report formats (80-column and ADES) are generated from it, both in the batch reporter and in the Webrella interface. It has been designed to be a companion format to ADES, where the latter is used to store and retrieve observation information from the MPC database, and the IPEF is used to carry detection information and metadata within asteroid detection software.

IPEF has currently been developed only to the degree that it can be used to carry information from the detection pipelines into Webrella; however there are future plans to further extend this format. First is to make IPEF bidirectionally convertible to/from the in-process Umbrella representation without significant loss of metadata, such that processing of detections can occur across process boundaries, allowing easy post-processing through Python scripts or for use in more advanced detection analysis tools running locally on the computers of astronomers, which can then be used to further refine the quality metrics for detection reporting. Another area of extension is on the pre-processing side, where it should carry the necessary image metadata for producing complete ADES reports from a detection pipeline or other downstream software (such as Webrella).

Due to the modular and extensible nature of the Umbrella pipelines, reporting the detections to the user involves an interplay of the detection element and the reporting element. Common properties are passed through the detection representation, which are used to generate ADES and 80-column reports. However, even more information is extracted from the detection pipeline, which is made available through extensible reporting properties. These are used to pass user-facing measurements to the user interface layer. Moreover, the stamps shown to the user are also generated by STU, through a callback function. This is done such that interactive GUIs, with re-sizable stamp sizes, can

be developed for STU. Multiple stamps are generated as a 2D list, over the different types of images (originals, masked, etc.) and sub-stacks. The design also supports sending plots as stamps (such as radial intensity plots), although this functionality has not been used yet.

The automated detection pipeline system is envisioned to provide the STU output to Webrella. For this reason, a reporter element called BatchReporter stores the results of the detection pipeline in IPEF format, as well as ADES XML and MPC 80-column converted from the IPEF representation. The extra information is stored in CSV files. Stamps are saved as PNG images, with the types of images, co-addition methods and size of the stamps configurable. The IPEF data, CSV files and stamps are read and displayed by the Webrella pipeline. The CSV data is made available in the list of properties of the web interface. The IPEF data is converted to MPC 80-column and displayed in the interface; upon validation, the selected detections are stored on disk on the server in the ADES format and downloaded to the reducer's device at the same time. The stamps are shown on the right side of the detection line, with both dimensions accessible by keyboard navigation.

3. Synthetic Tracking implementation

At the time Stănescu & Văduvescu (2021) was being published, our group began focusing on the synthetic tracking method. After considering and exploring the available options for achieving real-time synthetic tracking for surveys, the importance of co-addition modes and cache locality (see Chellappa et al. (2007) for explanations of the computing terminology) became clear to our group. To verify our hypotheses, we developed an initial prototype, which was a simplified version of the pipeline described in Fig. 4. This prototype became the cornerstone of the ParaSOL project, which aimed to develop the software suite, particularly the synthetic tracking implementation from TRL2 to TRL4.

Therefore, the concepts presented in this paper are valid across its versions.

The following paragraphs and the pipeline diagram directly apply from v0.3, the first complete pipeline on the new module system, to v0.6. Newer STU versions apply equivalent operations, with different implementations and intermediate representations that improve runtime, but their detection performance is not yet characterized. For completeness, we mention that the results in this paper obtained primarily with v0.6.10, which is the latest release of the v0.6 series.

The following terms will be used throughout the paper to explain the working of the pipeline:

- **Granularity** is the sampling step of the trajectory space. STU is using pixels on the farthest image as its measure of granularity. For comparison, these are inversely proportional with the percentages from Tycho Tracker (Parrott 2020). A granularity of 1 px should correspond to a setting of 100% in Tycho and one of 2 px to a setting of 50% in Tycho, although differences in the two implementations make this equivalence less exact.
- **Rays** are hypothetical motion vectors, together with a starting point for the ray. In this paper, the term motion vector will either imply a known starting point (in which case it is synonymous with a ray starting from the implicit pixel) or any starting point (such as when stacking whole images along a motion vector).
- An **isochrone** is an image plane of equal time. Unless otherwise specified, it is an image or a set of images taken at the same time (i.e. for which the exposures are overlapping). We prefer this definition to highlight the intersection of a ray with an image, as well as cover the multi-sensor case, where multiple images may be used in generating a single isochrone of a macrotile. Finally, non-image temporal planes are also useful for defining coordinate system references, such as the center isochrone below.
- **Macrotiles** are image cubes (containing multiple isochrones, as described above) fed into the detection pipeline. In the STU versions described here, only a single macrotile may be processed from a set of input images (limiting the combined input image size to that of the GPU VRAM). Support for processing multiple macrotiles for a single field has been added since v0.7 of STU.
- The **center isochrone** is a real (for odd isochrone count) or virtual (for even isochrone count) isochrone, at the median of the temporal axis of the macrotile. This is used as the reference for the ray coordinates. Note that STU assumes that the center isochrone occurs roughly in the middle between the first exposure in the macrotile and the last exposure in the macrotile. While deviations from this assumptions are expected to be functional, they may lead to sub-optimal performance of the software, particularly in terms of increased runtime.

A visualization partially compatible (our ray origin is on the center isochrone and not on an edge one) with our definitions of ray, isochrone and macrotile is presented by Whidden et al. (2019) (see Fig. 2 from page 9).

The STU pipeline is engineered from the ground up to attain the high processing throughput necessary to handle large volumes of data in real time (or near real time). Since the highest computational cost occurs in the brute force scanning phase, this step is surrounded by pre-processing and post-processing stages to hoist as much computation out of the scanning loop as possible. The stages of this pipeline are illustrated in Fig. 4.

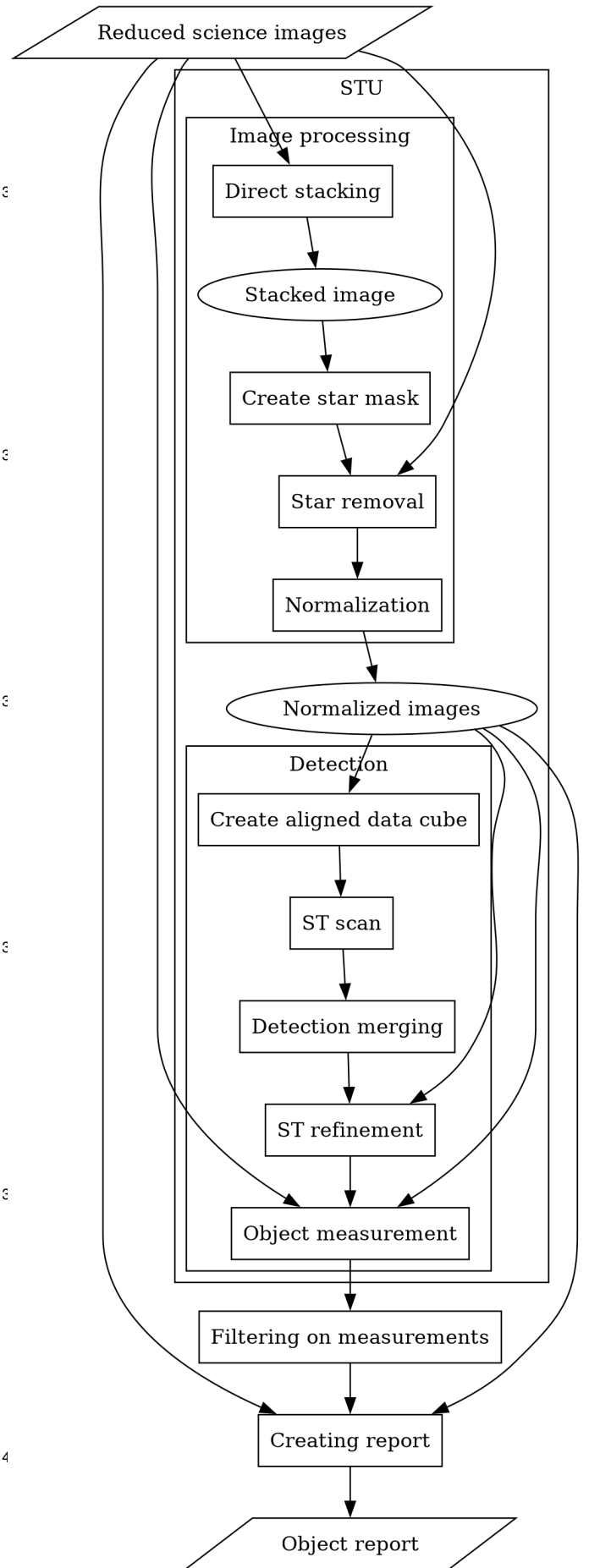


Fig. 4: **STU internal pipeline stages.** The image processing and the detection blocks are part of STU proper. The reduced science images are obtained with IPP or other pre-processing pipelines. The filtering and reporting is handled by other Umbrella modules.

In the pre-processing stage, the images are brought into the shape of a data cube that can be directly iterated over. This means resampling the images into a common coordinate system (such that the search occurs over pixel coordinates), as well as applying a mask that removes fixed sources. On the other side, the detection and post-processing part are designed around the concept of hypothesis rejection. This type of pipeline uses increasingly more expensive and effective filters to reject candidate detections from the list of motion vectors. This way, the brute force search can use an inexpensive quick check that is easily parallelizable over each pixel and motion vector.

3.1. Image processing

On the pre-processing side, the first step is to evaluate the background level of the images, which is obtained from the first two statistic moments (mean and standard deviation) on small sub-blocks of the image, out of which the median values are used. The input images are then normalized them to a common level set in the pipeline configuration file, and co-added to obtain an image of the fixed sources. Co-addition is performed with a median averaging method, which leaves only faint residuals of the sufficiently fast moving objects (residuals which decrease with the number of input images, making this method perfect for deep synthetic tracking). The masking co-addition is necessary to mask faint stars that can be still picked up by the co-addition in the synthetic tracking although not bright enough to be detected in individual images. In particular, the signal to noise ratio is comparable to that of the detection method, which allows this method to be used in the blind as long as good enough astrometric data is available. The classic Umbrella blink blob detector (based on connected component of pixels above detection threshold, see Stănescu & Văduvescu (2021)) is used to identify the bright fixed sources, which are directly and completely masked off. As discussed in Stănescu & Văduvescu (2021), completely masking off the bright sources eliminates the residuals from the slight differences in PSFs of the input images. Notably, for large scale surveys affording the use of a sufficiently deep (potentially locally-generated) reference star catalog, these steps may be bypassed in the future, using a list of known stars. To date, the usage patterns of STU have not suggested the need of such a method, for which reason no support for loading lists of known fixed objects has been implemented in STU up to now.

An optional high-pass filter is then used to remove large-scale patterns due to atmospheric effects (including halos from bright stars, Moon gradients, etc.). When the input data is of high quality and lacks such large-scale patterns, it is possible to turn off this filter, as it may introduce extra noise (due to it being generated from neighboring pixels), particularly on the edges of bright stars, where the plain mask shifts the background level lower than the local sky value, thus boosting the star edges. Except for such known minor inconveniences, the rolling median filter used for the removal of low-frequency patterns from the image has yielded satisfactory performance, comparable with the results given by *astnoisechisel* (Akhlaghi 2019).

The data is then resampled with nearest neighbor interpolation into a common World Coordinate System, currently chosen as that of the first image in the stack. Albeit imprecise, for non-photometric usage this method has proven to be adequate while being much faster than the alternatives. Moreover, we intend to strongly recommend that input images should be already resampled to a common World Coordinate System from the pre-processing stage, leaving the internal resampling as a slower fall-

back option when the images cannot be prepared as such in advance.

3.2. Brute force scan

The core synthetic tracking operation is a *brute-force* scan over the whole data cube. The scan uses an associative score method (ASM), which is a highly effective generalization of the usual co-addition techniques used for synthetic tracking. The method used by STU has a low operation count per pixel, while maintaining robustness to noise and high selectivity, which are typical ASM advantages over its predecessors. These advantages and underlying concept of associative score methods as a generalization to the usual averaging co-additions will be described in an upcoming synthetic tracking review paper currently in preparation.

The scan is parallelized such that each workitem (one lane of the Single Instruction Multiple Data units of the processor) handles a group of 4x2 pixels (called microtile), while the motion vectors are scanned in a program loop. A further inner loop accumulates data over multiple images. The parallelization and loop order results in improved cache locality. The low computational cost method used by STU is normally dominated by the bandwidth required to access all the data once per each motion vector. For this reason, the much higher bandwidth between the execution units and the cache memory results in large speedups for STU. Furthermore, on newer versions of STU, the data accumulation is further sped up by small integer dot product instructions (which are typically used for quantized machine learning inference) on the accelerators that support them (including all modern AMD, Intel and ARM GPUs, but not nVidia). Currently, each microtile is split into 2 separate rays of 2x2 pixels; each of which is binned after sampling. The advantages of our approach compared to binning before sampling will be discussed in a future paper.

The scanned motion vectors are directly generated from the loop control variables, in a hollow cone pattern. There are two, configurable, velocity radii for the cone. First, there is an inner lower limit to avoid evaluating fixed sources that could yield scores higher than the desired target options. This is especially important for faint stars and stellar residue which remain unmasked (which is necessary to avoid threshold lowering of passing bright objects). The other is the maximum proper motion limit. The sampling step is also configurable, in terms of pixels on the farthest image, that is the distance in pixels between the intersection of consecutive motion vectors with the isochrone most distant from the center isochrone. In the ParaSOL project we customarily use a value of 0.99 pixels on the farthest image. This ensures that there are no gaps in the sampling, hitting each possible pixel within the motion cone.

The brute-force scan can be further sped up by a "thirding" process, where the initial scan occurs only on a subset of the entire data, which would quickly eliminate implausible vectors before attempting to evaluate the score on the complete data cube. Limitations of the current implementation described below in this paper, particularly pixel hijacking have prevented the widespread use within the ParaSOL project. We expect this thirding process to become relevant after solutions are implemented to the described limitations and to be refined in the future into hybrid synthetic tracking with detection linking methods.

The result of the brute force scan is a list of the motion vectors. The current implementation provides at most one such motion vector for each input pixel, resulting in a highly selective filtering of the search space.

3.3. Hypothesis rejection

The individual motion vectors do not map, by themselves, directly into detections. Due to the PSF, a single object will be spread over multiple pixels and generally multiple microtiles. Moreover, the motion vectors from multiple microtiles of the same detection could be different. For this reason, the first stage following the brute force scan is the clustering of the motion vectors and pixels into detection blobs. The clustering is based on a connected component algorithm in the image space, keeping all motion vectors associated with the pixels in the detection. This is followed by scanning a stamp of the detection along each motion vector, using a trimmed mean co-addition. Each pixel of each stamp is evaluated together with its neighbors to determine the highest scoring ray of the detection. At this point, each individual detection corresponds to an individual candidate object, and filtering begins on the individual detection level.

A larger stamp (of configurable size) is then co-added along the ray on one or more sub-stacks (depending on the user supplied configuration) and the blink detector is applied to the stamp of each sub-stack to measure and extract the object. A set of filters is then applied based on the measured properties (such as Full-Width Half Maximum, ellipticity, flux and magnitude, number of pixels, etc.) of the object and a tracklet is generated from the detections on the sub-stacks. There is a two-fold reason for doing so: first, the barycenter of the detection may be determined with potentially greater accuracy than the sampling step of the motion rays. Second, upon reporting to the Minor Planet Center, there is a requirement to provide independently estimated positions for each detection. By measuring the tracklet independently on each sub-stack, the software can provide such independent positions.

Finally, if the functionality is enabled, the detections are compared against the SkyBoT database to determine whether the objects are known or not.

4. Results

As a proof of concept for our methods, we present the results obtained from observations performed with three telescopes of varying apertures. Observations made with the 1.52 m Carlos Sánchez Telescope (TCS) were used to validate the detection of bright targets in a fully automatic manner with the default settings. The 0.25 m T025-BD4SB served as a test bench for generating various scenarios, including observations of extremely fast objects with different brightness levels relative to the detection limit. Here, we present the detection of a satellite. Finally, we utilized observations from the 2.54 m Isaac Newton Telescope, equipped with the old Wide Field Camera (WFC), to evaluate the performance of our methods on a noisy detector and to analyze potential errors and artifacts.

All these tests were focused on STU, while the IPP was only used for the INT data. The TCS and T025-BD4SB have their own automatic image processing pipelines for data preparation.

4.1. Hardware setup

Most of our tests, especially the early ones reported in this paper were done on the ParaSOL1 computer. This computer had an AMD Ryzen 5700X CPU paired with 64GB RAM, which we upgraded in the second half of 2023 to a Ryzen 5950X with 128GB RAM. The other system specifications remained the same. The GPU used for Synthetic Tracking is an AMD Radeon RX 6800 XT, with 16GB of VRAM. The software and the Ubuntu 22.04

LTS operating system have been installed to a Samsung SSD 980 1TB drive. The science data is stored on a RAID6 array of 6 Seagate IronWolf Pro 4TB hard drives, with an effective storage capacity of 16TB. The same RAID array was also used for intermediate data to minimize wear on the SSD until the RAM increase allowed us to use a tmpfs ramdrive to store the intermediate images.

A second PC (ParaSOL2) using the replaced Ryzen 5700X paired with 128GB and an AMD Radeon 7900XTX was installed in the second half of 2023. A similar storage setup was deployed using 6 Seagate Exos X16 16TB drives for an effective capacity of 64TB. Due to the release cadence of Ubuntu, the OpenCL stack for this GPU was not available on the Ubuntu OS at the time of the installation, requiring the use of a container, for which we chose ArchLinux. This made the setup more difficult to use.

For both PCs we have used the ROCm OpenCL driver. To run the STU software, just as with any other Umbrella software, we have used the latest mono⁷ package available at the time, which provides the implementation of the Common Language Runtime for running the Umbrella binaries.

4.2. The TCS data

We used the asteroid observations obtained with TCS as part of a survey conducted to characterize the color indexes of near-Earth objects (Popescu et al. 2021). The TCS is located at Teide Observatory (latitude: 28° 18' 1.8" N; longitude: 16° 30' 39.2" W; altitude: 2386.75 m).

It has a 1.52 m primary mirror mounted on an equatorial mount, with a focal ratio of $f/13.8$ in a Dall-Kirkham type configuration⁸. The survey employs the MuSCAT2 imaging instrument (an acronym for ‘a multicolor simultaneous camera for studying atmospheres of transiting exoplanets’) (Narita et al. 2019), which is installed at the Cassegrain focus of the telescope. A lens system reduces the focal length to a ratio of $f/4.4$.

This setup enables simultaneous photometric observations in four visible broad-band filters: g (400 - 550 nm), r (550 - 700 nm), i (700 - 820 nm), and z_s (820 - 920 nm). Each of the four channels is equipped with an independently controllable CCD (charge-coupled device) camera (1024 × 1024 pixels) with a pixel scale of $\sim 0.44''/\text{px}$ and a field of view (FoV) of $7.4 \times 7.4 \text{ arcmin}^2$. For testing the STU pipeline, we used only the data acquired with the r-band filter. The exposure duration was constrained by the tracking system. The telescope operates in equatorial tracking mode, but the aging mechanics of the instrument become unreliable for exposures exceeding 30–60 seconds, depending on the telescope’s orientation. Consequently, exposure times were generally limited to this range.

To test our data we selected 242 fields acquired between May 28, 2018 and June 12 2022. Each field includes on average 112 images. These were already calibrated by the survey program. The pre-processing of the data consists of dark subtraction and flat-field correction. The calibration images were acquired at the beginning and end of the night. Then, the PHOTOMETRYPIPELINE (PP)–Mommert (2017), a software written in Python, performs the astrometric registration, aperture photometry, photometric calibration, and asteroid identification. PP uses the *Astromatic* suite⁹, namely SExtractor for source identifica-

⁷ <https://gitlab.winehq.org/mono/mono>

⁸ <http://research.iac.es/00CC/iac-managed-telescopes/telescopio-carlos-sanchez/>

⁹ <https://www.astromatic.net/>

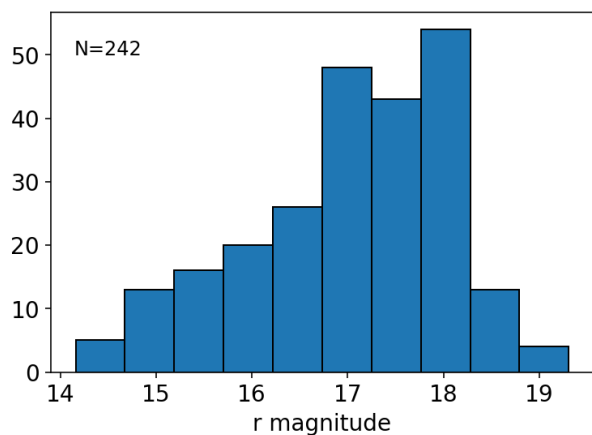


Fig. 5: Distribution of the apparent average r magnitude of the NEAs observed by TCS.

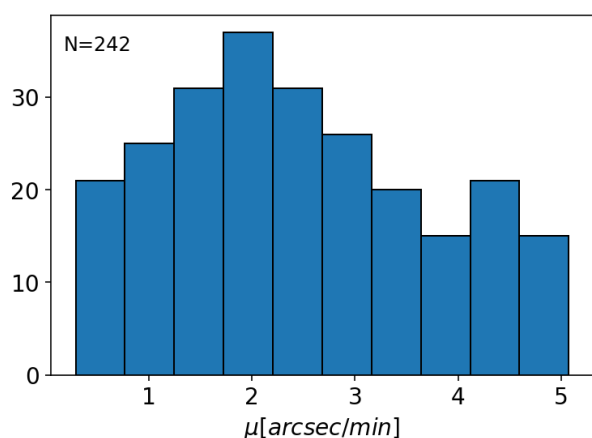


Fig. 6: Distribution of the apparent motions of NEAs observed by TCS.

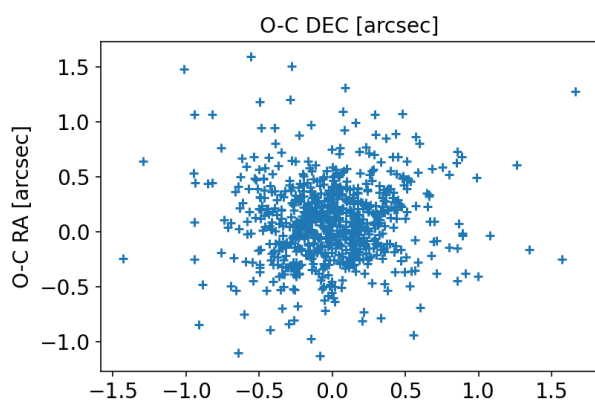


Fig. 7: Observed minus Computed (O-C) residuals for the 726 positions reported by STU for the 242 fields observed with TCS.

The selected dataset includes asteroids with apparent motion lower than 5 arcsec/min (in compliance with the default setting for STU). Since the observations for this dataset were performed for photometric purposes, all asteroids are detectable in individual images. The distribution of the observed magnitudes (as reported by PP) and the apparent motion (as reported by the JPL Horizons¹⁰ system) are presented in Figs. 5 and 6.

All tests were conducted using the default settings for STU, including the apparent motion limit. The targeted asteroids were detected in all fields, and three positions were reported. The differences between the observed positions (as reported by STU) and the ephemerids (as computed using the JPL Horizons system) are shown in Fig. 7. The values are: $(O - C)_{\text{RA}} = -0.01 \pm 0.36$ arcsec and $(O - C)_{\text{Dec}} = +0.09 \pm 0.37$ arcsec. These results validate the detection of all bright targets using STU. Due to the small field of view of the TCS telescope, only the targeted NEA was within the field of view. Additionally, we note that for all these fields, no other candidates were reported by STU, indicating that no artifacts were introduced.

4.3. Satellite detection using T025-BD4SB observations

The detection of very fast objects (with apparent motions on the order of hundreds or thousands of arcseconds per minute) presents a challenge for the synthetic tracking algorithm due to the extremely high apparent motion, which requires a large search interval to detect the motion vector. Additionally, short exposure times must be used to limit the trail size, but this results in the detection of only a few stars in the field.

To test our STU algorithm in these cases, we observed several satellites using the T025-BD4SB telescope. The 0.25 m aperture T025-BD4SB telescope was built as a collaboration between professionals and amateurs, alongside amateur astronomers from Astroclubul București. This telescope is a Newtonian design with a focal ratio of f/4, mounted on a robotized equatorial mount (Skywatcher EQ6-Pro). Images were captured with the QHY294M¹¹ CMOS camera, operating in 11.7 megapixels mode, with a resolution of 0.956 arcsec/pixel.

A total of 36 observations were acquired on the evening of March 19, 2023, using the T025-BD4SB telescope. The exposure time was set to 1 second, and the satellite's trail measured 35 pixels in length. After adjusting the configuration file, STU detected the satellite after approximately 3 minutes of computation, with 2 minutes and 32 seconds spent on the GPU. The velocity range used was 1800''/min – 2200''/min with a granularity of 0.99 px (corresponding to 1.54''/min), yielding a total of 4.8 trillion rays checked (average of $\sim 4 \times 10^5$ trajectories for each pixel), due to three quarters of the trajectories falling outside the field of view. The result is shown in Fig. 8.

Finally, because the object was bright, we checked and confirmed that the object can be detected with a subset of only 7 exposures. In this case, the 0.99 px granularity resulted in a sampling step of 8.6''/px, yielding an average of 0.54 trillion rays checked. In this case, the GPU runtime was merely 5 seconds.

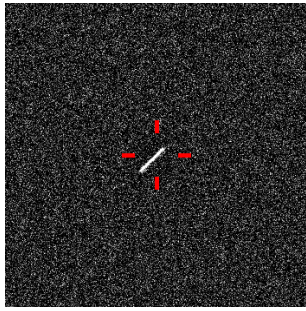
4.4. Applications on the INT Image Archive

Nine fields were observed close to opposition or to the ecliptic (within 5 degrees) between 2019-2022 for synthetic tracking projects using the 2.5 m Isaac Newton Telescope (INT) at Roque de los Muchachos Observatory (ORM) located atop La Palma is-

¹⁰ <https://ssd.jpl.nasa.gov/horizons/>

¹¹ <https://www.qhyccd.com/astronomical-camera-qhy294/>

tion and aperture photometry (Bertin & Arnouts 1996), SCAMP for astrometric calibration (Bertin 2006), and SWarp for image regridding and co-addition (Bertin et al. 2002). The GAIA catalog (Gaia Collaboration et al. 2018; Gaia Collaboration 2018) was used for astrometric registration. The PanSTARRS catalog (Tonry et al. 2012; Chambers et al. 2016) was used for photometric calibration of the data acquired with the Sloan filters.



```
0250000 KC2023 03 19.72184 07 27 04.60 +37 27 32.7 00.0 R 073
0250000 KC2023 03 19.72211 07 27 54.70 +37 38 02.7 00.0 R 073
0250000 KC2023 03 19.72239 07 28 44.96 +37 48 31.0 00.0 R 073
```

Fig. 8: Detection of a satellite using STU algorithm. Left side shows the STU output stamp obtained from stacking 12 exposures. The detection report in 80 columns format is shown on the right.

land. Endowed at its $f/3.3$ prime focus with the Wide Field Camera (WFC) which consists of an L-shape mosaic of 4 2154×4200 (2048 × 4096 available optically) 13.5 μm CCDs (0.33''/px) which cover in total 0.25 sq.deg, the INT covers an *etendue* of 1.3 which is feasible for some deep survey works (discovery and recovery), and during the past decade we used this facility for such related projects (Vaduvescu et al. (2013), Vaduvescu et al. (2015), Vaduvescu et al. (2018), Vaduvescu et al. (2021)). In Table 1 we include the observing log, listing the field name, observing date and UT interval, telescope pointing (α and δ), the exposure time (E) and number of images taken in each field (N), the Solar elongation, airmass interval, average seeing (measured stellar FWHM), and the Moon phase (when Moon was visible).

We used the ParaSOL Umbrella bundle v21 (STU v0.6.9) with its default configuration (i.e. we did not tune the parameters) to reduce the observed fields, detecting in total 184 valid moving sources, which include 142 known asteroids and other 42 unknown moving sources also cross-identified with the Tycho Tracker software.

We used IPP legacy version for image calibration (including astrometry) and STU for synthetic tracking. We measured all known and unknown asteroids cross-correlated with Tycho detections, using the Pan-STARRS 1 catalog as the astrometric and photometric reference (due to the match of the INT Sloan r filter), which typically includes at least one hundred catalog stars in each reduced field of each CCD of the WFC.

According to the MPC database as of April 2024, queried by SkyBoT, there are 169 known asteroids within the 9 observed fields covered by the WFC (one of which, due to its uncertainty, potentially outside the observed field). Following careful analysis of the individual reduced images, we counted 25 known asteroids invisible to blink (15% from the entire known sample), which actually represent an opportunity to test synthetic tracking. From the sub-sample of 25 known asteroids invisible to blink technique, STU v0.6.9 detected 10 targets and missed other 15. It also detected other 2 unknown asteroids invisible to blink which could be correlated with the Tycho Tracker.

Therefore, in its default configuration, STU detected 84% from all 169 known asteroids falling over the observed fields. From these 142 known asteroids detected, 8 asteroids have only 2 positions (6% of the detected sample). Also, 4 asteroids (3%) had duplicate detection reports (resulting from disjoint clusters). Out of the other 27 known asteroids that could not be detected (16% of the entire known sample), some of them were lost in the noise caused by the known WFC CCD3 vignetted corner (where IPP masked the entire region due to increased noise), others due to slow motion (5 objects had proper motions $\mu < 0.2''$) or due to involvement with fixed sources (stars). Finally, very bright targets (V magnitudes 15.8–18.3) were also not detected,

most probably due to being filtered out as image artifacts. We expect that this detection ratio will be significantly improved once we deploy an automated tuning method. Reasons other than software processing for missed objects include faintness (expected apparent magnitudes V between 22.5–24.3), or/and fading due to lightcurves, weather and atmospheric factors (including Moon gradients) and instrumentation issues (such as slippage of the telescope tracking). As a point of reference, Tycho Tracker missed 10 known asteroids (6%).

In Fig. 9 we plot the Observed minus Calculated (O-C) residuals for 407 synthetic tracking measurements of the 142 known asteroids detected by STU v0.6.9 in the 9 INT archive fields. We found an average total residual of 0.19'' (0.02'' in α and 0.03'' in δ), and a total standard deviation of 0.12'', which is about one third of the size of a WFC pixel. We reported to Minor Planet Center (MPC) all positions for all moving sources measured by STU, including all known asteroids plus the unknown detections matching Tycho findings.

In Fig. 10 we plot the histogram of all the apparent magnitudes of the 184 asteroids detected by STU in the 9 fields, using relative photometry (Sloan r band matching the filter used for this INT survey) of the SDSS reference stars detected by IPP in the observed fields. An apparent limiting magnitude around $R \sim 24.0$ is visible, consistent with the INT SIGNAL¹² predictions, giving the variable observing conditions of the 9 fields (R between 23.8–24.3 mag) and assuming signal to noise limit of 3. In comparison, using the classic blink detection with individual images taken in the same observing condition, the limiting magnitude was supposed to be only $R \sim 21.9$.

4.5. Runtime

To compare the performance of this software across different workloads, we will use a metric named pixel processing rate (PPR). The PPR represents the number of image pixels visited (read from memory) in the brute force scanning stage over the time span needed to visit them. The associative scoring implementation used by STU allows it to visit each image only once per ray, hence the PPR scales linearly with the search space. Moreover, it is related more directly with the GPU specifications, such as memory bandwidth or floating operations per second (FLOPS) rating. The following equation makes the PPR definition precise:

$$C = WHJN = v_{\text{PPR}} \tau \quad (1)$$

Where C is the search space (number of pixels visited), W and H are the width and height of the data cube, J is the average num-

¹² <https://astro.ing.iac.es/signal/>

Table 1: Nine fields observed between 2019-2022 with the INT/WFC facility for synthetic tracking aims.

Field	Date (UT)	RA/DEC	E x N	Elo	Airmass	FWHM	Moon
OPP0	20190306 21:36-23:21	10:30:00 +09:30:00	30x105	170	1.48-1.13	2.0	dark
OPP1	20210827 00:17-00:45	22:00:00 -06:00:00	30x30	174	1.23-1.21	1.0	79%
OPP2	20210827 00:47-01:15	22:00:00 -07:00:00	30x30	174	1.23-1.24	1.0	79%
OPP3	20210827 01:17-01:45	22:00:00 -08:00:00	30x30	174	1.26-1.29	1.0	79%
OPP4	20210827 01:47-02:15	22:00:00 -09:00:00	30x30	174	1.31-1.36	1.0	79%
OPP5	20210827 02:17-02:45	22:00:00 -10:00:00	30x30	174	1.39-1.48	1.0	79%
OPP6	20210827 02:46-03:15	22:00:00 -11:00:00	30x30	174	1.51-1.66	1.0	79%
ESTU	20220728 02:46-04:23	23:00:00 -05:00:00	30x100	140	1.25-1.21	1.8	dark
EST2	20220728 04:25-05:13	23:00:00 -04:00:00	30x50	140	1.20-1.28	1.8	dark

ber of trajectories tested per pixel (size of the search cone) and N is the number of images. Similarly v_{PPR} is the pixel processing rate and τ is the time to scan all the pixels. As an example, we have recorded the detailed runtime information from running on the OPP3 field as well as the subset of the first 22 images. We chose the subset of 22 images due to a sharp discontinuity in the pixel processing rate (PPR) after this value. This suggests that 22 images better fit the GPU cache, especially since the pixel processing rate for the full OPP3 field is nearly identical with the PPR at 23 images. In both cases, we ran with a search cone with a maximum proper speed of $20''/\text{min}$. Due to the large maximum proper speed compared to the image size, many trajectories fall outside the data cube and are not tested. In both cases the granularity was set to 0.99 px/s . However, the sampling steps are different, due to the time between the first and last images being different between the full field and the subset.

For the following examples, we provide the values of W and H for the data cube (which can be different from the individual image sizes) from the compiler arguments of the GPU code. The maximum trajectories per pixel is computed from the sampling step and the search cone radius, both also passed as compiler arguments. For C , STU reports the value by computing it from WHJ , the total number of trajectories actually scanned, and N . The total number of trajectories scanned is measured on the GPU (for performance reasons implemented with parallel private counters on each workitem, which are atomically summed into a global counter once the workitem finished processing the microtile).

On the full OPP3 field, with $N = 30$, $W = 4240$, $H = 2224$, we have $C = 3.9 \times 10^{14}$, giving an average $J = 1.38 \times 10^6$, out of 2.6×10^6 maximum trajectories per pixel. With a $\tau = 419 \text{ s}$ (7 minutes brute force scan, 8.1 minutes complete STU execution), we have a PPR of 931 Gpx/s .

On the first 22 images of OPP3, we have $N = 22$, $W = 4240$, $H = 2224$, and $C = 1.87 \times 10^{14}$, giving an average $J = 9.01 \times 10^5$ out of 1.4×10^6 maximum trajectories per pixel. With a $\tau = 132 \text{ s}$ (2.2 minutes brute force scan, 3.5 minutes complete STU execution), we have a PPR of 1412 Gpx/s .

To put these results into perspective, this GPU has 512 GB/s bandwidth to the VRAM, and a 20 TFLOPS rating (can execute 10 trillion instructions per second). These results also demonstrate the ability of STU to exploit cache locality, as an implementation that co-adds the images would under no circumstances be able to exceed 512 Gpx/s , the maximum bandwidth to the VRAM of the card used, when co-adding 8-bit images. By comparison, STU achieves twice to thrice this bandwidth, at an equivalent of 7 – 11 instructions per pixel.

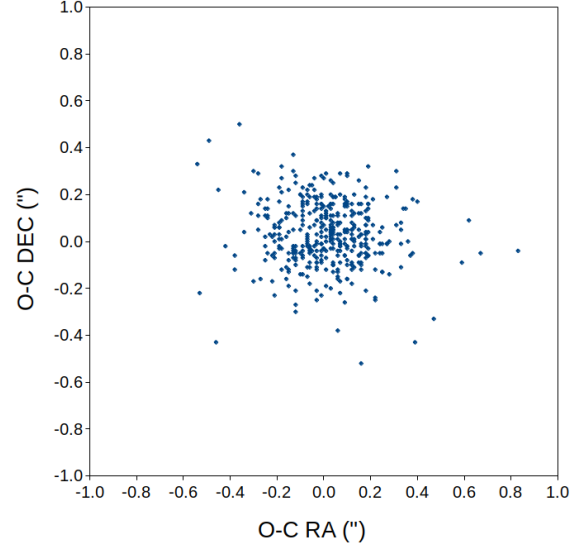


Fig. 9: Observed minus Calculated (O-C) residuals for 142 known asteroids detected in the 9 INT observed fields. Root mean square residuals for 547 positions were detected and measured using STU, based on the Pan-STARRS DR1 catalog reference. An average O-C total residual of $0.19''$ and a total standard deviation of $0.12''$ were calculated.

4.6. Limitations of our software

Among the limitations of the method used in STU, we have observed two forms of detection hijacking, both different from the ‘detection hijacking’ concept introduced in our previous blink pipeline (see Stănescu & Văduvescu (2021)). Nonetheless, all three types result in losing a true positive from overlapping false positives.

Pixel hijacking One of the detection hijacking modes observed in STU is pixel hijacking. Due to the output of the first stage containing at most one motion vector per pixel, whenever many bright image artifacts appear in the raw images, or, in the case of very low SNR of the true positives, even noise, which may line up and cross the same center isochrone as a real detection, then these artifacts may end up as the highest-scoring vector for the given pixel on the center isochrone. In a quick investigation of the loss of a few true positives when activating the thirding mode of STU, this has been identified as the most common cause, followed closely by low visibility on the central images (due to crossing image gaps, from either sensor gaps or areas with dense masks of image artifacts).

Notably, pixel hijacking demonstrates the data dredging effects of the ST technique. If a true positive is suppressed by false

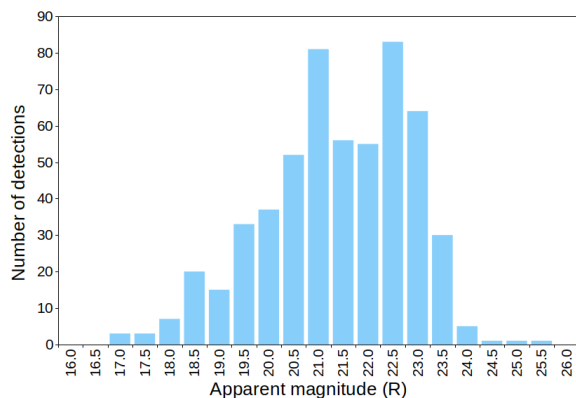


Fig. 10: Apparent magnitude (measured R values) for the 547 positions of the 184 asteroids detected in the 9 INT observed fields, based on the photometric SDSS catalog reference. An apparent limiting magnitude cut around $R \sim 24.0$ is visible given the observed conditions of the nine amalgamated fields.

positives, then a similar set of false positives would be present in the output of the stage. By using other correlating information, false positives will be suppressed, however the correlating information typically loses its information value to the latter validation steps (especially human validators), as the filtered results cherry-pick the candidates which already fit the expected profiles of real detections.

This way we would like to express support for opinions of caution in the reporting of candidates from ST – some of the criteria human validators have been using to tell apart false positives from real detections may already have been used in the reduction pipelines and the confidence of detecting a real object may in practice be lower than expected by validators not trained and knowledgeable in the filtering stages of the detection pipelines.

Cluster hijacking STU is also susceptible to hijacking in the second (clustering) stage leading to either candidate elimination in the third stage or third stage false positive hijacking. This occurs by including the real objects inside large false positive clusters. In such cases, a true positive may be suppressed by the neighboring false positives, especially in areas of high background variability, where large clusters of false positives may appear.

Catastrophic percolation Percolation is the most obvious and an extreme kind of cluster hijacking. Unlike the regular cluster hijacking which occurs on a small scale, when operating the software close to the noise floor of the method (using detection thresholds that allow a large percentage of noise to exit the brute force stage), a phase transition occurs where clusters ‘precipitate’ from the random noise in large amounts and sizes. As the detection threshold is lowered, clusters merge and true positives decrease up to a catastrophic loss of detections.

This is an effect of site percolation in the clustering stage. As the detection probability of random noise on a single pixel approaches and exceeds the critical site percolation probability of the square lattice (approximately $p \approx 0.593$, according to Akhunzhanov et al. (2022)), random noise starts to cluster quickly on large scales (size distribution becomes sub-exponential, approaching a power law with the Fisher exponent of $\tau = 187/91$), and therefore begins cluster hijacking all across the macrotile. This has been observed when optimizing parameters over recently acquired data from the KASI telescope; the

original analysis is presented in Vaduvescu et al. (2025). At detection thresholds under 3 standard deviations of the co-added image, we have often observed one cluster larger than half the size of the entire image.

Limiting magnitude Due to the percolation limitation above and the lack of a prioritization scoring method for the human validation, with our usual detection settings, STU has an effective detecting limit ~ 0.5 magnitudes above the limit of the Tycho Tracker software.

5. Discussions and Conclusions

We have implemented and proven the operation of a real-time synthetic tracking pipeline for detecting near-Earth asteroids. The results from the early phases of the development also prove its detection capabilities, especially on datasets where pre-processing eliminated effectively the image artifacts. The lower detection rates on the datasets with lots of sensor artifacts (such as INT above, or the telescopes of the Korean Microlensing Telescope network) highlight the need for improved tuning capabilities for our software in order to reach mission grade technology readiness levels.

Nonetheless, even at the current development stage, the ability of STU to scan for objects moving at much faster rates has yielded several independent discoveries that would have been difficult to achieve with other methods. For example 2023 XA3 was detected at 7.8 arcsec/min in the $2^\circ \times 2^\circ$ FOV of KASI telescope. This object and other results from applying STU to live mini-surveys are presented in Vaduvescu et al. (2025). The degree to which such a fast implementation can impact the capabilities in the minor planet discovery are discussed in the next paragraph, followed by the steps we identified that we need to take in order to bring our implementation to the maturity needed to achieve the impact described. Some of these steps have already been implemented, and are in the testing phases; their results will be available in an upcoming paper, including a more detailed analysis of runtime, energy draw and other factors of interest over a wider variety of hardware.

5.1. Implications of fast Synthetic Tracking

Our opinion is that a fast Synthetic Tracking implementation is capable of fundamentally changing the approach to detecting new, fainter, asteroids. We expect that once such a method becomes widely available, Synthetic Tracking will rapidly and completely replace the classical blink.

On a short timescale, we expect to see shallow deployments (namely ST using few, often less than 10, exposures) replacing blink pipelines in existing large surveys, driven by the need to scan deeper than the long-reached limits of the blink method of the telescope. These shallow deployments are able to re-use the instrumentation designed for blink (which may not have particularly fast readout speeds) to go deeper than previously possible. In parallel, we expect an explosive growth in proposals for new surveys using dedicated hardware (many small telescopes with very large etendue, as well as fast sensors). This should not be a surprise, as there already are very effective dedicated ST surveys, such as Maury – Attard – Parrot (MAP)¹³, which have already demonstrated the capabilities of this method, by discovering in just a few years hundreds of NEAs using a few RASA 11" telescopes and the Tycho Tracker software developed in-house.

¹³ <https://www.spaceobs.com/fr/Blog-de-Alain-Maury/MAP-historique-et-description>

Finally, as the benefits of this method become obvious, there will be extensive focus on ramping up the know-how of the community on the practical aspects of deploying ST surveys.

On longer timescales, we expect batteries of smaller telescopes to become the norm for asteroid surveys, due to their much lower cost than total etendue equivalent large telescopes, given a limiting magnitude and sky coverage targets. As more surveys will use ST, even large mirror telescopes will have difficulty discovering faint asteroids compared to massive arrays of small telescopes, if these large telescopes will keep using the blink method. For this reason, we expect that all large-scale surveys will end up using synthetic tracking.

For the extremely wide fields allowed by massive telescope batteries (which can cover almost the whole sky), the next major bottleneck will be the maximum integration time allowed by the method, especially when considering the fastest moving objects, such as those passing within a few Lunar Distances or even inside the Moon orbit. Similarly, dredging the list of potential objects for cross-night confirmations can push the minimum SNR for valid detections even lower. For these, we expect the extension of the current ST methods with blink techniques of pairing sparse detections, which would allow extending the total integration time far beyond the current pure ST approaches.

However, the largest impact might not even be for the ground-based instruments. This method has the potential to unlock niche approaches to detecting asteroids, through the use of balloon-borne and small space telescopes. Both these approaches alleviate or remove the effects of the atmosphere, which currently are the main roadblocks in detecting fainter objects. In particular, on space telescopes, where weight is typically far more expensive than the complexity of the optics, it is far more convenient to use a small mirror with a large field of view optical system. For these systems, synthetic tracking is a good fit, trading faintness for a narrower scan area, which allows the detection of asteroids at a distance. A side advantage to such free-floating telescopes is the method's innate image stabilization compatibility (as it can work with very short exposures directly and does not necessarily require a separate recombination step as digital stabilization does or complex hardware for optical stabilization). These advantages are particularly valuable when considering objects approaching Earth from the side illuminated by the Sun, which require space-based telescopes to detect ahead of impact, such as the proposed NEOMIR (Licandro et al. 2024).

Finally, the increase in detection capabilities from the use of high-speed ST will expand the database of known NEAs, some of which are far more easily accessible for space missions, including those of industrial nature (for example asteroid mining).

On a side note, we expect that ST will lead the way to other computationally intense operations being accelerated by fast programmable hardware. One notable such example is the image pre-processing step, since applying dark and flat-field photometry corrections is a massively parallel problem. Similarly, thanks to highly efficient index structures of plate solving software (such as Astrometry.net), one major cost of plate solving is the (robust) detection of stars and their measurement, which can also be parallelized to a large degree.

5.2. Future steps

Thanks to the many observing runs on which we achieved real-time or near real-time data reduction, we have discovered several practical bottlenecks not initially envisioned in the project. The biggest issue we have yet to work on is the quality control of our intermediate steps, particularly the REDSCI images. Having

a human check the REDSCI images introduces significant delay and makes automating pipelines difficult. For quick cadence surveys, having a high image throughput and/or real-time reporting, where the latency introduced by checking the intermediate images and synchronizing with the status of the pipelines, checking these intermediate results should be automated. Another related issue is the difficulty of adjusting parameters on the fly if the outputs are wrong. This can easily happen due to variance of the sky conditions (cirrus or variable amount of clouds, seeing, background gradients due to Moon, etc). For example, if the default parameters for Astrometry.net (Lang et al. (2010)) fail (due to differences in the number of detected stars, for example), different ones are needed, and this change should happen as soon as possible. Similarly, if the PSF changes significantly, tuning the detection parameters may be necessary to keep detecting objects or drop false positives. For these, the degree of image measurement and self-tuning needs to be improved. Moreover, existing tuning tools (such as the Object Tracer in STU) need improved infrastructure, such as the ability to automatically retrieve known objects in the field of view, identify the detection blockers and automatically identify tuned parameters that would allow detecting said objects.

Also on the tuning side, we plan to improve our parameter tuning setup based on the accumulated database of known valid and invalid objects. This again requires more infrastructure around STU's Object Tracer and improved usability. Currently, the Object Tracer is just a logging tool that enables some logging logic in a certain image region and search cone. For this and the previous purpose, it also needs to be able to output structured data, such as the thresholds that need to be set for a certain object to pass. Moreover, to improve the speed of the tuning, it needs the ability to trace multiple objects.

On a shorter term, we have implemented re-tiling of large mosaic sensors (making it possible to run synthetic tracking on images larger than the VRAM limits of the GPUs) and we re-wrote our pipeline stages to improve the runtime, including merging some GPU operations, changing algorithms, and moving a large number of operations to the GPU, which are available starting with STU v0.7.0. These, together with an improved measurement stage, will be presented in an upcoming paper.

On a longer term, we plan to improve our IPEF specification and make it public. In particular, we plan to better define the REDSCI-associated information, including information on the astrometric and photometric calibration, which is needed for the ADES conformance of the detection pipelines output reports. We also plan to use digest2 (Keys et al. (2019)) Vereš et al. (2023)) and Find_Orb¹⁴ to highlight potentially interesting objects.

Based on the observing runs and our reducers' experience with Tycho Tracker, several inconveniences were identified with the Web GUI, such as the inability to re-stack with larger window sizes. The lack of the desktop GUI for STU (which we have yet to adapt from the old blink pipeline), which would have solved the previous problem and slightly increased responsiveness, has also been raised several times by our reducers. Finally, a point was raised internally on using the pipeline directly from Python, bypassing the use of the executable pipeline host. This was demonstrated possible, but cumbersome, and made difficult by the lack of convenient documentation formats. For this, we plan to add doxygen to the CI scripts to export the documentation in all the pipelines, and not just the core library (umbrella2).

Acknowledgements. The work for this article was supported by a grant of the Romanian National Authority for Scientific Research (UEFISCDI), project number

¹⁴ https://www.projectpluto.com/find_orb.htm

PN-III-P2-2.1-PED-2021-3625 (PI: Marcel Popescu), contract number 685PED
registered on 21/06/2022.

References

- Akhlaghi, M. 2019, arXiv e-prints, arXiv:1909.11230
- Akhunzhanov, R. K., Eserkepov, A. V., & Tarasevich, Y. Y. 2022, *Journal of Physics A: Mathematical and Theoretical*, 55, 204004
- Allen, R. L., Bernstein, G. M., & Malhotra, R. 2002, *AJ*, 124, 2949
- Bertin, E. 2006, in *Astronomical Society of the Pacific Conference Series*, Vol. 351, *Astronomical Data Analysis Software and Systems XV*, ed. C. Gabriel, C. Arviset, D. Ponz, & S. Enrique, 112
- Bertin, E. & Arnouts, S. 1996, *A&AS*, 117, 393
- Bertin, E., Mellier, Y., Radovich, M., et al. 2002, in *Astronomical Society of the Pacific Conference Series*, Vol. 281, *Astronomical Data Analysis Software and Systems XI*, ed. D. A. Bohlender, D. Durand, & T. H. Handley, 228
- Chambers, K. C., Magnier, E. A., Metcalfe, N., et al. 2016, arXiv e-prints, arXiv:1612.05560
- Chellappa, S., Franchetti, F., & Püschel, M. 2007, *How to Write Fast Numerical Code: A Small Introduction* (Berlin, Heidelberg: Springer-Verlag), 196–259
- Denneau, L., Jedicke, R., Grav, T., et al. 2013, *Publications of the Astronomical Society of the Pacific*, 125, 357–395
- Gaia Collaboration. 2018, *VizieR Online Data Catalog: Gaia DR2* (Gaia Collaboration, 2018), *VizieR On-line Data Catalog: I/345*. Originally published in: 2018A&A...616A...1G
- Gaia Collaboration, Brown, A. G. A., Vallenari, A., et al. 2018, *A&A*, 616, A1
- Gorgan, D., Văduvescu, O., Stefanut, T., et al. 2019, arXiv e-prints, arXiv:1903.03479
- Keys, S., Vereš, P., Payne, M. J., et al. 2019, *PASP*, 131, 064501
- Kruk, S., García Martín, P., Popescu, M., et al. 2022, *A&A*, 661, A85
- Lang, D., Hogg, D. W., Mierle, K., Blanton, M., & Roweis, S. 2010, *AJ*, 139, 1782
- Licandro, J., Conversi, L., Delbo, M., et al. 2024, in *European Planetary Science Congress*, EPSC2024–882
- Mommert, M. 2017, *Astronomy and Computing*, 18, 47
- Narita, N., Fukui, A., Kusakabe, N., et al. 2019, *Journal of Astronomical Telescopes, Instruments, and Systems*, 5, 015001
- Parrott, D. 2020, , 48, 262
- Popescu, M., de León, J., Licandro, J., et al. 2021, in *European Planetary Science Congress*, EPSC2021–820
- Popescu, M. M., Văduvescu, O., de León, J., et al. 2023, *A&A*, 676, A126
- Prodan, G. P., Popescu, M., Licandro, J., et al. 2024, *MNRAS*, 529, 3521
- Shao, M., Nemat, B., Zhai, C., et al. 2014, *The Astrophysical Journal*, 782, 1
- Stănescu, M. & Văduvescu, O. 2021, *Astronomy and Computing*, 35, 100453
- Tonry, J. L., Stubbs, C. W., Lykke, K. R., et al. 2012, *ApJ*, 750, 99
- Văduvescu, O., Birlan, M., Tudorica, A., et al. 2013, *Planet. Space Sci.*, 85, 299
- Văduvescu, O., Gorgan, D., Copandean, D., et al. 2021, *New A*, 88, 101600
- Văduvescu, O., Hudin, L., Mocnik, T., et al. 2018, *A&A*, 609, A105
- Văduvescu, O., Hudin, L., Tudor, V., et al. 2015, *MNRAS*, 449, 1614
- Văduvescu, O., Hudin, L., Tudor, V., et al. 2015, *Monthly Notices of the Royal Astronomical Society*, 449, 1614
- Văduvescu, O., Stănescu, M., Popescu, M., et al. 2025, *New Astronomy*, 119, 102410
- Vereš, P., Cloete, R., Weryk, R., Loeb, A., & Payne, M. J. 2023, *PASP*, 135, 104505
- Whidden, P. J., Bryce Kalmbach, J., Connolly, A. J., et al. 2019, *AJ*, 157, 119
- Yanagisawa, T., Kamiya, K., & Kurosaki, H. 2021, *PASJ*, 73, 519
- Yanagisawa, T., Nakajima, A., Kadota, K.-i., et al. 2005, *Publications of the Astronomical Society of Japan*, 57, 399