NEAR EARTH BODY DETECTION AND LINKING

P. Rajan[1] *P. Burlina*[1,2] *M. Chen*[2] *B. Jedynak*[3] *C. Krupiarz*[2] *N. Mehta*[2] *A. Sinha*[1] *G. Hager*[1]

The Johns Hopkins University
[1]Dept. of Computer Science, [2]Applied Physics Laboratory, [3] Dept. of Applied Mathematics and Statistics

ABSTRACT

Most asteroid population discovery has been accomplished to-date by earth-based telescopes. It is speculated that most of the smaller Near Earth Objects (NEOs), up to 140 meters in diameter, whose impact can create substantial city-size damage, have not yet been discovered. Many asteroids cannot be detected with an earth-based telescope given their size or the location of the Sun. Our objective is therefore to develop an efficient asteroid detection and linking algorithm that can be hosted on-board a spacecraft. By having on-board algorithms, the system would also minimize the need to downlink entire images taken by a space-based telescope. We describe one such image processing pipeline we have developed for onboard asteroid detection and characterize its performance.

Index Terms— Asteroids detection, identification.

1. INTRODUCTION

NASA has a congressional mandate to discover all Near-Earth Objects (NEOs) at least 1 kilometer in diameter. Fortunately, 95% of the NEOs larger than 1 km that have been discovered are likely not to impact Earth. Near-Earth Object search programs [1] are currently almost exclusively accomplished by earth-based telescopes such as MIT's LINEAR [2] project, the NEAT [3] program, the Catalina Sky Survey, or Pan-STARRS [4]. An exception is JPL's spacecraft-based WISE telescope brought out of hibernation to characterize NEOs in the 3.4 and 4.6 micron infrared bands [5] (called NEOWISE).

It is notable, however, that the NEO that has impacted Chelyabinsk, Russia on 15 February 2013 was only about 17 meters in size. The impact of a 50 meter asteroid that caused the Tunguska Event of 1908 could have destroyed an entire city or metropolitan area. It is estimated that only a relatively small fraction of those so called "city-killing" asteroids, particularly objects less than 140 meters in diameter, have been discovered to date. Because of their size, atmospheric effects and the location of the sun, some of these NEOs cannot easily be detected with an earth-based telescope.

Our focus here is therefore on developing an algorithm that can be hosted on-board a spacecraft. Understanding that there are processing and resources (memory) constraints on-board, our algorithm design needs to not only meet the performance objectives of detecting and identifying asteroids using a space-based telescope, but it also needs to have a small footprint to be implementable with the limited on-board resources. This paper describes an agile algorithm candidate that is being investigated as well as our use of representative real and simulation imagery for testing it.

Prior image analysis work for asteroid detection for ground-based image observations is summarized: In [4], a reference processing system for detection and identification of asteroids – named the Pan-STARRS Moving Object Processing System (MOPS) – is described. This pipeline aims at identifying moving objects in our solar system and linking those detections within and between night observations. It attributes those detections to known objects, calculates initial and differentially corrected orbits for linked detections, recovering detections when they exist, and orbit identification. Most proposed pipelines for earth based detection include a step to combine images to create a high S/N static-sky image that is subtracted from the current master image to obtain a difference image containing only transient sources. Examples include [6], where a shift-and-add technique is used to improve signal to noise ratio and then synthetically creating long exposure images to facilitate the detection of trajectories. A related shift-and-add method using a median image rather than an average image is reported in [7]. A match filter is used for asteroid detection and matching in [8]. In [9, 10, 11], tree based searches (including KD-trees) are used for efficient linking of successive asteroids detections and finding sets of observation points that can be fitted with an inherent motion model, through an exhaustive search for all possible linkages that satisfy the expected model constraints.

In Section 2 we describe a small-footprint pipeline (with regard to memory and CPU usage) that can be deployed on an on-board system. This approach uses Principal Component Analysis based searches to link object detections into trajectories. Since space-based asteroids surveys such as NEOWISE are still rare, we describe in Section 3 a validation method using space-based simulated images as well as appropriate earth-based image datasets that contain associated ground truth.

This work was supported by an Early Stage Innovations grant from NASA?s Space Technology Research Grants Program NNX14AB04G.

2. APPROACH

We describe the main components of our image processing pipeline (shown also in the flow chart in figure 1) addressing detection and linking of asteroids seen in images acquired from space-based platforms. At a high level, the pipeline is summarized as follows: as input to the pipeline is a sequence of time-lapse images. Pre-processing and image registration are used to bring the images into alignment with a common image of reference. Detection of bright bodies is done via thresholding, followed by logical differencing between each image and a reference image containing objects that are common to all registered images in the sequence of images. After differencing, which allows us to detect moving objects, we use an additional step to filter detections based on size and shape considerations (using tools such as connected components and morphological filters). A list of detection coordinates is then generated. Candidate trajectories are then generated for these coordinates by efficiently checking rectilinearity by using both Principal Component Analysis (PCA) and 2d-trees. A last verification step further checks additional conditions such as the fact that candidate trajectories must be composed of temporally consecutive mover observations. Component module are detailed next.

Image Pre-Processing This step includes employing techniques to reduce noise (e.g. median filter) and artifacts that depend on the acquisition device.

Image Registration We aim to align each image in the sequence to a common reference image so that the stars in the background line up in all images. In the case of a triplet of images, typically the second image is used as the reference image. We estimate and then apply the necessary similarity transformation (translation, rotation, scaling) and/or skew (for a full affine transformation) to all images in the sequence such that image objects (stars) are mapped into the same pixel location and so that all transformed images have the same spatial resolution. Image Registration using mutual information [12] and cross-correlation as similarity measures are used.

Image Logical Differencing A global thresholding is applied to the registered image for detecting asteroids and suppressing background noise. As asteroids are typically very faint compared to the surrounding stars, the selection of the detection threshold impacts false alarm rate. Thresholding yields binary detection images. The set of all binary images is then used to generate an intersection image that contains objects that occur in all images in the sequence. This is followed by logical differencing whereby we produce a set of difference images by intersecting the corresponding binary image with the negative of the common intersection image. This operation provides a list of candidate detections for each image in the sequence. While the logical differencing results in good detections, additional artifacts such as crater-like formations (see Fig. 6) are seen as a result of some of the celestial bodies being over-exposed. To mitigate this artifact, we subsequently perform filtering out of hollow objects as well as filtering based on object size.

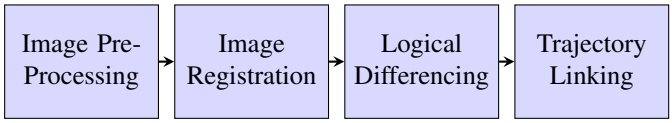

Fig. 1. Image Processing Pipeline

Trajectory Linking The list of centroids of moving objects obtained from image differencing defines a set of candidate rectilinear trajectories. The goal is to find a subset of centroids that fit a linear model. Even though the model is simple, the set of filtered centroids potentially has a high number of noisy points (falsely detected movers), and the cardinality of the set of all candidate trajectories increases exponentially with the number of detections, thus requiring subsequent pruning of this set. We use a combination of PCA and 2d-trees in order to find the trajectories efficiently. Unlike MOPS [4] and CSS[13], we do not set an upper limit on the velocity of the asteroid, and hence do not risk missing potential fast moving targets. Given a sequence of images, we form all the possible trajectories connecting the detections in the first and last frames. There are $O(n^2)$ trajectories, where n is the number of detections per image. We then find the point of intersection of each of these trajectories with the frames in the middle. We construct a 2 dimensional tree for all the frames in the middle, and perform a range search on the tree to generate the detections that lie within a small radius of the point of intersection. This query can be done on $O(log(n))$ time on average. Once we find a collection of such points that potentially form a linear trajectory, we perform PCA and compute the ratio of eigenvalues, $\lambda_1/(\lambda_1 + \lambda_2)$ in order to develop a line confidence score for each candidate trajectory, and choose lines for which this ratio exceeds a threshold. Using the candidate trajectories thus found, we then enforce temporal order of detection by using the sign of the projection on the principal eigenvector. As the final step, we further eliminate false positives by ensuring that the distance between projections is proportional to the time interval between images. Compared to a brute force line search of $O(n^3)$ for a triplet of images, our algorithm takes $O(n^2log(n))$ time.

3. EXPERIMENTS

We detail the experiments performed using a range of simulated space-based imagery generated using the JHU APL developed *Renderer and Camera Emulator* (RCE) as well as real imagery from the NEAT dataset.

3.1. Real Imagery

3.2. Simulated Space-Based Imagery

Using RCE, asteroids are modeled as spherical blackbody-like emitters (emissivity is less than 1), with a cross-sectional area that approximates the sizes of actual asteroids and sur-

Fig. 2. Left: An image simulated by RCE. Right: 31 simulated MWIR images super-imposed in order to visualize the trajectory of the asteroid in a single image. The true trajectory can be seen as a faint line towards the top center of the image

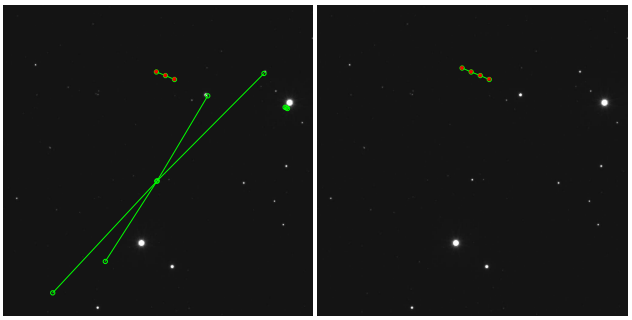

Fig. 3. The trajectories found by the pipeline are shown in green. The true location of the asteroid is marked in red. Left: Trajectory Detection on a simulated triplet. Right: Trajectory Detection on a quadruplet. Adding one more image to the triplet eliminates the false positives.

face temperatures typical of sun-illuminated asteroids in an Earth-like orbit. Similar to the way stars are modeled, the radiation emitted is modeled using a form of Planck's equation:

$$B_\lambda(T) = \epsilon \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k_B T}} - 1} \quad (1)$$

where $B_\lambda(T)$ is the spectral radiance at a given wavelength λ and temperature T (which in SI units would be $Wm^{-2}m^{-1}$). The value ϵ is the emissivity of the asteroid, which essentially converts the blackbody spectral radiance into spectral irradiance. The constant, h is the Planck's constant, c is the speed of light, λ is wavelength, k_B is the Boltzmann constant, and T is the temperature.

The asteroids are assumed to have a nominal temperature of 200 K due to solar heating and emissivities in the range from 0.9 to 0.98. The RCE uses stellar data available as part of the Two Micron All Sky Survey (2MASS), a stellar survey that scanned the entire sky in three IR bands (centered at 1.25 m, 1.65 m, and 2.17 m, respectively). The 2MASS catalog also incorporates data in two visible bands from other surveys. An example of the simulated MWIR image and the ground truth trajectory derived from one set of simulated imagery is shown in Fig. 2. Fig. 3 shows the final trajectories detected by our algorithm for one triplet and one quadruplet of the simulated MWIR dataset. As is shown in Fig. 3, using

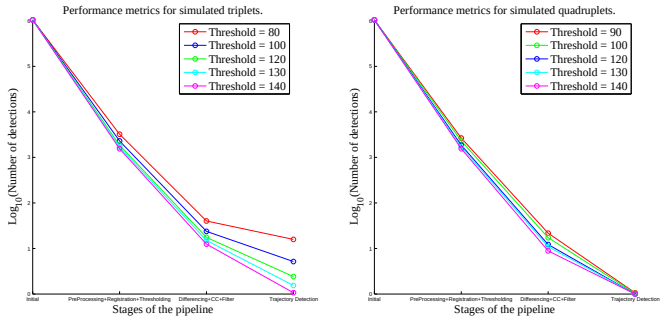

Fig. 4. Left: The number of detections at various stages of the pipeline for triplets of images. Right: The number of detections at various stages of the pipeline for quadruplets of images.

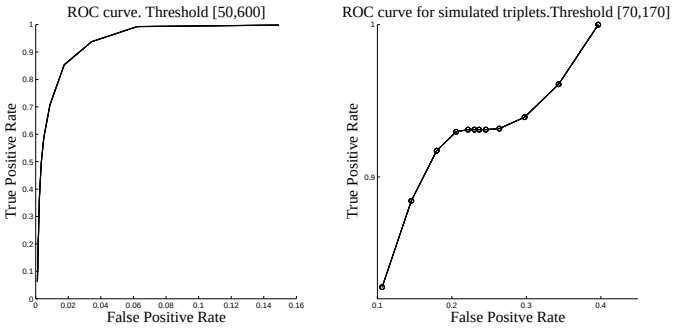

Fig. 5. Left: ROC curve for the detection of stars and asteroids after the Image thresholding stage of the pipeline. Right: ROC curve for asteroid detections at the final stage of the pipeline.

trajectory verification on a greater number of images in the sequence allows us to quickly disambiguate and reject false trajectories. In this case this trend is readily apparent when going from a triplet to a quadruplet of images.

We characterize the algorithm with regard to detections at each of the following three successive stages: the initial detection of objects, the detection of moving objects and the detection of trajectories. In Fig. 4, we plot the number of detected objects at each step of the pipeline for each selected threshold value. We can verify that the number of detections monotonically decreases when the threshold increases. Note that we have found that this not always the case, especially since a higher number of detections at the thresholding stage may induce more cancelations at the logical differencing. Additionally, we note in the right plot in Fig. 4 that the use of quadruplets allows for a single trajectory to be found irrespective of the value of the threshold used (and hence the number of detections found at the first stage), echoing the results displayed in Fig. 3. Last, we show the Receiver Operating Characteristic (ROC) for simulated space-based imagery in Fig. 5.

3.2.1. NEAT

Near Earth Asteroid Tracking(NEAT) [3] is an earth-based program run by NASA from 1995-2007 to discover NEOs. Fig. 6 and Fig. 7 show the results at all stages of the pipeline for one triplet of images of the 2002-CY46 asteroid obtained from the NEAT system archive.

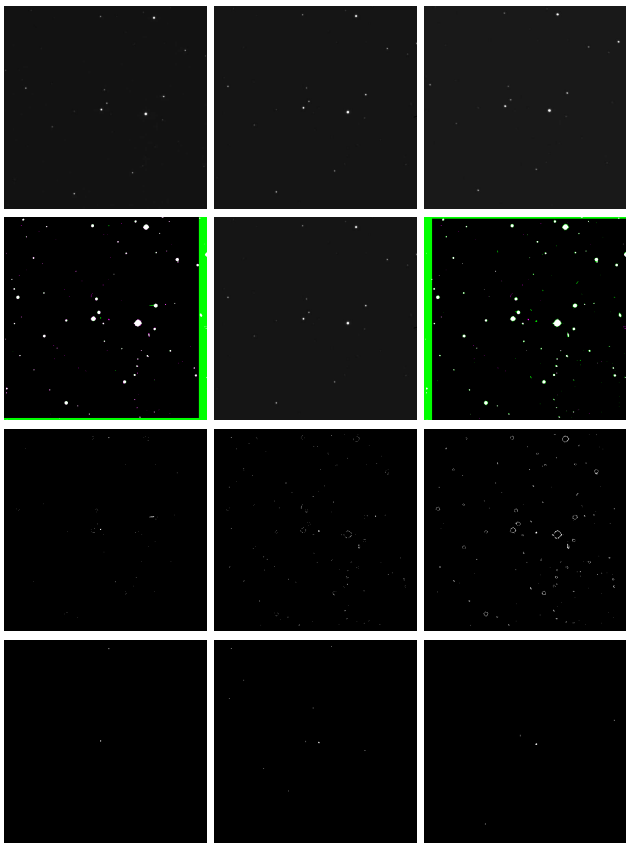

Fig. 6. Image Processing Pipeline results shown on 2002 NEAT data. (row 1): input image triplet (CY46) taken approx. 10 minutes apart. (row 2): Image Registration. Left: Image-1 registered to Image-2. Right: Image-3 registered to Image-2. (row 3): Image Differencing: Artifacts such as crater-like formations are seen in the difference images above. This is the result of some celestial bodies being over-exposed.) (row 4): Image Differencing: Filtered centroids shown in each image of the sequence.)

3.2.2. CATALINA

The Catalina Sky Survey (CSS) [14, 15] is intended to discover NEOs, specifically potentially hazardous asteroids that pose risk to earth. Fig. 8 show the results at all stages of the pipeline for one triplet of earth-based images from CSS.

3.3. Implementation

This pipeline has been developed and tested in Matlab and has also been jointly implemented in C++ on a recent Intel

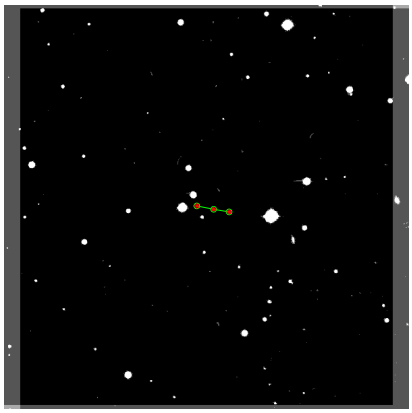

Fig. 7. Trajectory Detection for the NEAT CY46 Triplet. Asteroid trajectory detected is shown in green. True location is in red. 3 Images of the triplet are super-imposed here after registration and thresholding for ease of visualization.

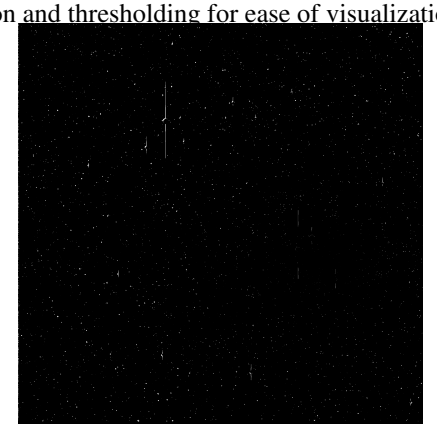

Fig. 8. Image Processing results shown on Catalina Sky Survey data.

processor. Preliminary benchmarking on this processor have shown that peak memory usage is of the order 92 Mb. Additional studies for extrapolating CPU usage and processing time figures for a proxy spacecraft target deployment processor such as the MCP750 have led to metrics that seem promising and compatible with deployment on a spacecraft architecture. More work is currently being conducted to validate these figures including activities involving the deployment of this algorithm onto a native MCP750 architecture using FPGAs.

4. CONCLUSION

We describe a pipeline for on-board NEO detection. Such algorithms are critical to allow the detection of moderate to small asteroids from space-based surveys, as these asteroids are not easily seen in earth-based surveys yet they pose a significant potential for damage if they hit the earth. We characterize the algorithmic performance on simulated space-based as well as real earth-based imagery. Performance results and preliminary computational requirements are promising with regard to the possible future deployment of this algorithm on proxy spacecraft processors such as MCP750.

5. REFERENCES

[1] Grant H Stokes, Jenifer B Evans, and Stephen M Larson, "Near-earth asteroid search programs," *Asteroids*, vol. 3, pp. 45–54, 2002.

[2] Jenifer B Evans, Frank C Shelly, and Grant H Stokes, "Detection and discovery of near-earth asteroids by the linear program," *Lincoln Laboratory Journal*, vol. 14, no. 2, pp. 199–215, 2003.

[3] NASA, "The NEAT survey," 2014.

[4] Larry Denneau, Robert Jedicke, Tommy Grav, Mikael Granvik, Jeremy Kubica, Andrea Milani, Peter Vereš, Richard Wainscoat, Daniel Chang, Francesco Pierfederici, et al., "The pan-starrs moving object processing system," *Pan*, vol. 125, no. 926, pp. 357–395, 2013.

[5] NASA, "The NEOWISE survey," 2014.

[6] Michael Shao, Bijan Nemati, Chengxing Zhai, Slava G Turyshev, Jagmit Sandhu, Gregg Hallinan, and Leon K Harding, "Finding very small near-earth asteroids using synthetic tracking," *The Astrophysical Journal*, vol. 782, no. 1, pp. 1, 2014.

[7] Toshifumi Yanagisawa, Atsushi Nakajima, Ken-ichi Kadota, Hirohisa Kurosaki, Tsuko Nakamura, Fumi Yoshida, Budi Dermawan, and Yusuke Sato, "Automatic detection algorithm for small moving objects," *Publications of the Astronomical Society of Japan*, vol. 57, no. 2, pp. 399–408, 2005.

[8] Peter S Gural, Jeffrey A Larsen, and Arianna E Gleason, "Matched filter processing for asteroid detection," *The Astronomical Journal*, vol. 130, no. 4, pp. 1951, 2005.

[9] Jeremy Kubica, Joseph Masiero, Andrew W Moore, Robert Jedicke, and Andrew Connolly, "Variable kd-tree algorithms for spatial pattern search and discovery," *Robotics Institute*, p. 253, 2005.

[10] Jeremy Kubica, Andrew Moore, Andrew Connolly, and Robert Jedicke, "A multiple tree algorithm for the efficient association of asteroid observations," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 138–146.

[11] Jeremy Kubica, Larry Denneau, Tommy Grav, James Heasley, Robert Jedicke, Joseph Masiero, Andrea Milani, Andrew Moore, David Tholen, and Richard J Wainscoat, "Efficient intra-and inter-night linking of asteroid detections using kd-trees," *Icarus*, vol. 189, no. 1, pp. 151–168, 2007.

[12] Paul Viola and William M Wells III, "Alignment by maximization of mutual information," *International journal of computer vision*, vol. 24, no. 2, pp. 137–154, 1997.

[13] NASA, "The Catalina Sky Survey," 2014.

[14] S Larson, J Brownlee, C Hergenrother, and T Spahr, "The catalina sky survey for neos," in *Bulletin of the American Astronomical Society*, 1998, vol. 30, p. 1037.

[15] AJ Drake, SG Djorgovski, A Mahabal, E Beshore, S Larson, MJ Graham, R Williams, E Christensen, M Catelan, A Boattini, et al., "First results from the catalina real-time transient survey," *The Astrophysical Journal*, vol. 696, no. 1, pp. 870, 2009.

5.1. Appendix I: details on Memory and CPU usage characterization

We detail here information regarding CPU and memory usage and their implications for deployment on a reference proxy spacecraft platform such as the MCP750.

The pipeline was developed and tested on Matlab initially and then implemented also in C++ for doing a benchmark analysis using certain performance metrics. The C++ code was run on a 8-core Intel processor (3rd generation) with clock speed of 2.3 GHz and an Intel VTune Amplifier XE 2015 profiler was used. The peak CPU usage was 12%, peak private memory and peak working set memory usages were 99,212 kB and 84,740 kB respectively. The CPU time observed was 1.267 sec (spin time = 0.465 sec, effective time = 0.802 sec). A single CPU core was used during the 0.802 sec "effective time", and the processor was idle during the spin time. Therefore "spin time" has been used in the subsequent calculations. Some additional measurements made for the Intel processor are as follows: Instructions executed = 2,868,100,000, CPI (cycles per instruction) = 0.581, CPU frequency ratio(ratio between actual and nominal CPU frequency) = 0.573. These metrics were then extrapolated for a reference processor having maximum clock speed of 466 MHz (MCP750). We estimate performance time using the relation:

$$\tau = I \times IPC^{-1} \times C^{-1} \tag{2}$$

$$IPC = 1/CPI \tag{3}$$

where τ is the performance time, I is the total number of instructions executed, IPC is the instructions per cycle, C is the clock rate of the processor, CPI is the cylces per instruction. Here we assume that the total number of instructions executed by both processors are same. The performance time on the MCP750 can be predicted as

$$\tau_{MCP} = \frac{\tau_{Intel} \times IPC_{Intel} \times C_{Intel}}{IPC_{MCP} \times C_{MCP}} \tag{4}$$

Using $\tau_{Intel} = 0.802$ sec, $IPS_{MCP,max} = 525 \times 10^6$, we get,

$$IPC_{MCP,max} = \frac{IPS_{MCP,max}}{C_{MCP}} = \frac{525 \times 10^6}{466 \times 10^6} = 1.13 \tag{5}$$

If we assume 100% processor utilization on MCP750 then, $IPC_{MCP} = 1.13$. Thus,

$$\tau_{MCP} = \frac{0.802 \times 0.12 \times 0.581^{-1} \times 2.3 \times 10^9 \times 0.573}{1.13 \times 466 \times 10^6} = 0.414 sec \tag{6}$$

This could further be extrapolated to find performance time on MCP750 for other values of CPU utilization.

In the above calculation the CPU frequency ratio measured on the Intel processor has been multiplied to make use of the nominal CPU clock frequency on that processor. We predict the memory usage to remain same on MCP750 as that in our test. This is acceptable for our application given MCP750 has up to 256 MB dynamic RAM and that we can still have sufficient memory available for additional necessary processes such as navigation, guidance, communication etc onboard. Here the specifications of the MCP750 have been referred from the processor **??** and the measurements on the Intel processor were done using 'Open Hardware Monitor v.0.7.1-beta','Speedfan v.4.5' and 'Pc-Wizard v.2014.2.13'.

% CPU Utilization	Performance time(sec)
50	0.828
12	4.968
10	4.14

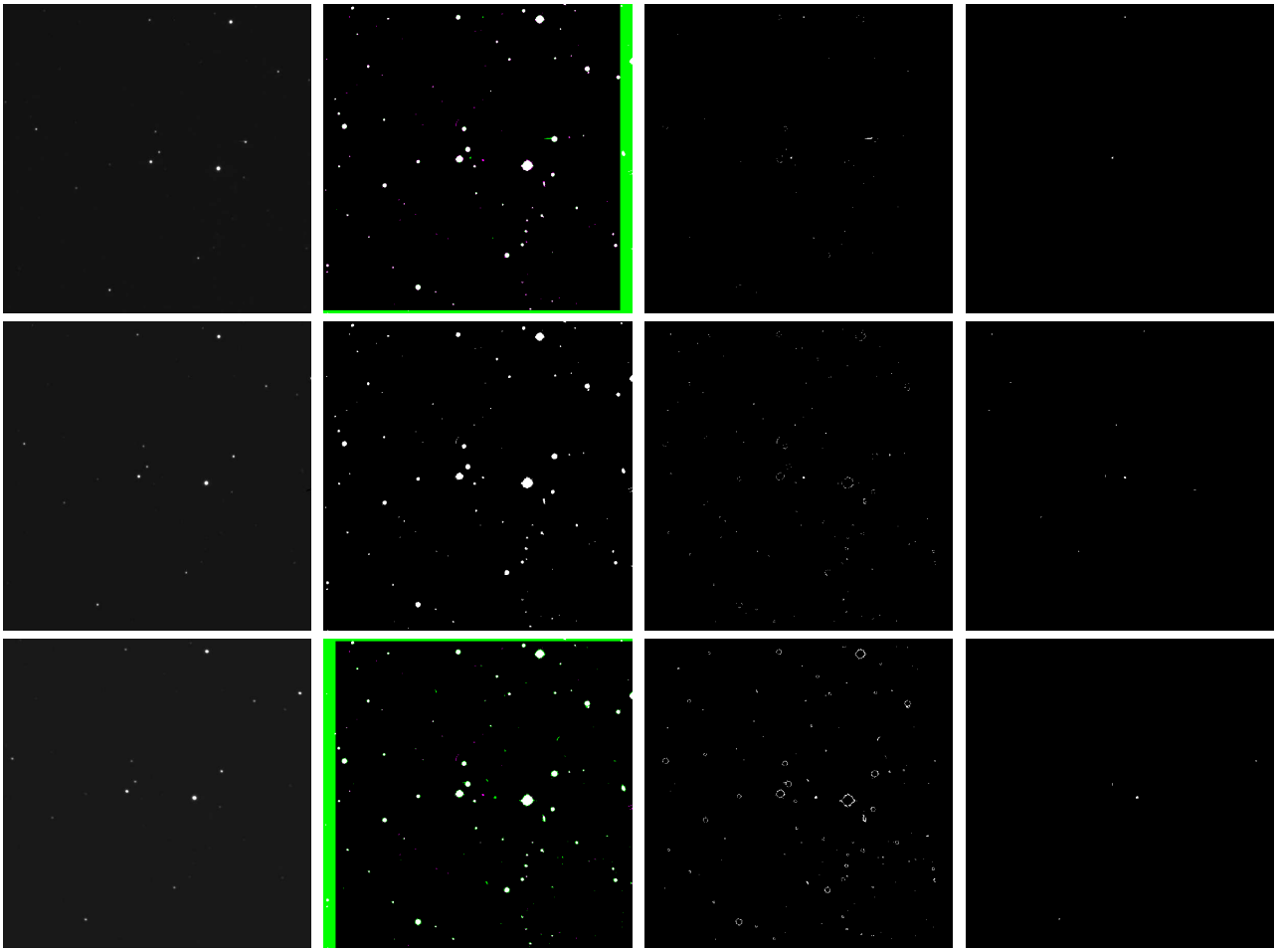

Fig. 9. Image Processing Pipeline results for the CY46 Triplet. (column 1): 2002 CY46 Triplet images taken approximately 10 minutes apart. Near Earth Asteroid Tracking (NEAT) system archive. (column 2): Image Registration results for the CY46 Triplet. Top: Image-1 registered to Image-2. Middle: Image-2 Bottom: Image-3 registered to Image-2. (column 3): Image Differencing results for the CY46 Triplet. (Artifacts such as crater-like formations are seen in the difference images above. This is the result of some celestial bodies being over-exposed.) (column 4): Image Differencing results for the CY46 Triplet. Filtered centroids in each image of the sequence.)