Comparison of Tracking-By-Detection Algorithms for Real-time Satellite Component Tracking

M. Nehru J. Attzs
Florida Institute of Technology
150 W. University Blvd.
Melbourne, FL 32901
mattzs2012@my.fit.edu

Trupti Mahendrakar
Florida Institute of Technology
150 W. University Blvd.
Melbourne, FL 32901
tmahendrakar2020@my.fit.edu

Mackenzie J. Meni
Florida Institute of Technology
150 W. University Blvd.
Melbourne, FL 32901
mmeni2021@my.fit.edu

Ryan T. White
Florida Institute of Technology
150 W. University Blvd.
Melbourne, FL 32901
rwhite@fit.edu

Isaac Silver
Energy Management Aerospace
2000 General Aviation Drive
Hangar 101
Melbourne, FL
isaac@energymanagementaero.com

Abstract—**With space becoming more and more crowded, there is a growing demand for increasing satellite lifetimes and performing on-orbit servicing (OOS) at a scale that calls for autonomous missions. Many such missions would require chaser satellites to autonomously execute safe and effective flightpath to dock with a non-cooperative target satellite on orbit. Performing this autonomously requires the chaser to be aware of hazards to route around and safe capture points through time, i.e., by first identifying and tracking key components of the target satellite. State-of-the-art object detection algorithms are effective at detecting such objects on a frame-by-frame basis. However, implementing them on a real-time video feed often results in poor performance at tracking objects over time, making errors which could be easily corrected by rejecting non-physical predictions or by exploiting temporal patterns. On the other hand, dedicated object tracking algorithms can be far too computationally expensive for spaceflight computers. Considering this, the paradigm of tracking-by-detection works by incorporating patterns of prior-frame detections and the corresponding physics in tandem with a base object detector. This paper focuses on comparing the performance of object tracking-by-detection algorithms with a YOLOv8 base object detector: namely, BoTSORT and ByteTrack. These algorithms are hardware-in-the-loop tested for autonomous spacecraft component detection for a simulated tumbling target satellite. This will emulate mission conditions, including motion and lighting, with a focus on operating under spaceflight computational and power limitations, providing an experimental comparison of performance. Results demonstrate lightweight tracking-by-detection can improve the reliability of autonomous vision-based navigation.**

TABLE OF CONTENTS

1. INTRODUCTION

The proliferation of spacecraft in orbit in recent years has led to numerous benefits to many fields: meteorology for weather forecasting and detecting forest fires, SATCOM (satellite communications) for secure communications, GPS for navigation, worldwide Internet access, and space exploration. However, this progress comes at a cost. The U.S. Department of Defense's global Space Surveillance Network tracks over 27,000 pieces of orbital space debris [1], such as decommissioned satellites, broken pieces of spacecraft, and fragments of launch vehicles from spacecraft deployments. NASA estimates there are millions of untracked pieces of orbital space debris. Most of this space debris is present in low earth orbit and can travel at speeds of thousands of miles per hour. Even small pieces of untracked debris greater than 5cm pose threats to human spaceflight safety and scientific, defense, and commercial assets.

Extending the lifetime of satellites is a major part of reducing the growth of space debris, and on-orbit servicing (OOS) of satellites can enable this. There have been several successful human-in-the-loop OOS missions by space shuttle astronauts [2] and even robotic autonomous OOS missions, such as ETS-VII [3], DART [4], XSS-10/-11 [5] [6], and MEV-1/-2 [7] [8]. Yet all these missions were performed around cooperative spacecraft. However, servicing the ever-increasing number of unfamiliar, non-cooperative assets in orbit at scale will require fully autonomous missions. An effective way to accomplish such missions is by utilizing a swarm of small chaser satellites that can autonomously identify capture points on a non-cooperative spacecraft and collaboratively navigate around hazards to dock with at the capture points. Using intelligent autonomous swarm satellites can not only minimizes the resultant forces and moments on the capture interfaces, but it can enabl fast-paced, complex,

978-1-6654-9032-0/23/$31.00 ©2023 IEEE

time-critical operations that would otherwise be infeasible due to time lag for ground-based control.

Prior work in the ORION Laboratory at Florida Tech [9] has shown state-of-the-art object detection algorithms like YOLOv5 [10] and Faster R-CNN [11] are effective at locating solar panels, antennas, and satellite bodies. This is done with test datasets of images of satellites and with hardware-in-the-loop experiments with a real-life satellite mockup. In addition, single-stage object detectors like YOLO can run in real-time under restrictive spaceflight-like computing constraints [12]. This is accomplished even for video feeds of satellites the object detectors have not observed in advance and with intraclass variation, e.g., with solar panels in various shapes and configurations [13]. Further, mapping the detections to 3D point clouds via stereographic cameras can dynamically feed artificial potential field (APF) guidance algorithms [14] that treat satellite bodies as attractive nodes and solar panels as repulsive nodes in space. These APF algorithms are successful at guiding chasers to capture points, both with simulated chasers [15] and real-world chaser drones [16].

While this end-to-end process has been shown to work to varying levels of success, the safety of the missions is not assured due to errors in the underlying computer vision algorithms. Object *detection* algorithms work by automatically locating the satellite components in video frames one-by-one from a video feed, with no analysis of the motion of the components through time. As such, implementation on a real-time video feed often results in deficient performance at tracking objects through time. These errors could be easily corrected by rejecting non-physical predictions or by exploiting temporal patterns. The present article studies the improvements that can be gained by using object *tracking* algorithms to locate hazards and safe capture points through time. We implement several low-compute object trackers that operate on top of the YOLOv8 [17] object detector in attempts improve its predictions using temporal patterns, train the trackers to work for tracking satellite components, and perform hardware-in-the-loop testing at the ORION Laboratory to measure their performance under various lighting and motion conditions.

The article is organized in the following order. Section 2 defines the object detection and object tracking tasks, reviews the relevant computer vision literature, and outlines specific failure modes trackers are expected to address. Section 3 provides information on datasets used for training the object trackers and the underlying object detector. Section 4 discusses the implementations of all algorithms used for comparison in the article. Section 5 covers hardware-in-the-loop experiments and the resulting performance metrics. Section 6 outlines the major findings of the study, their implications, and their significance.

2. Object Detection and Tracking

Recent advances in deep learning algorithms and computing hardware have led to accurate computer vision algorithms with low computational footprints. Convolutional neural networks (CNNs) accelerated by graphics processing units (GPUs) [18] on edge computers enable deployment of computer vision on spaceflight-like hardware.

Satellite Component Detection vs. Tracking

Prior work focused on the object detection task for spacecraft component detection [19]. Object detection requires an algorithm to localize objects with a tight bounding box and classify those objects—e.g., as satellite bodies, antennas, solar panels, and thrusters.

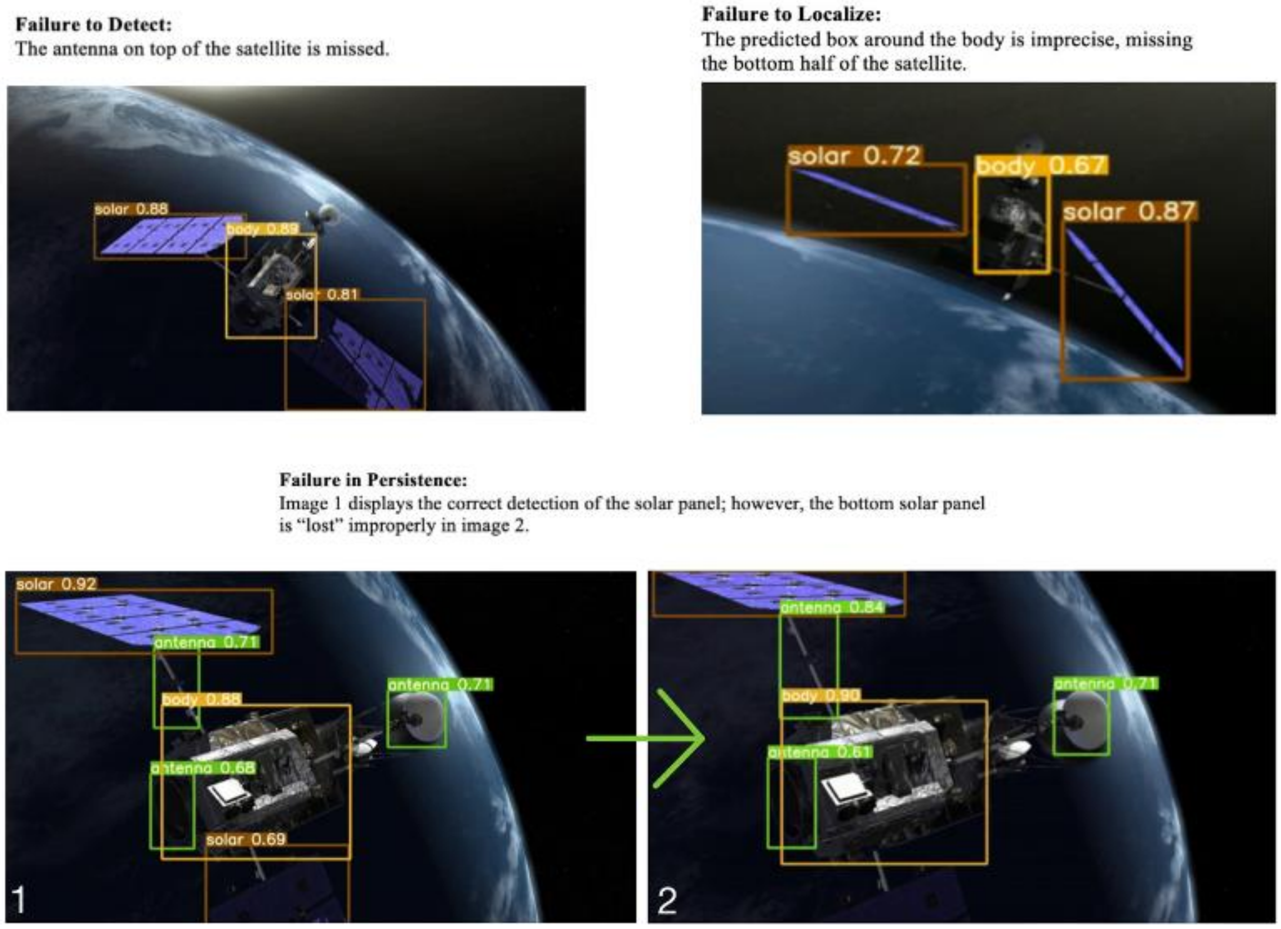
Object detection comes with many complications. Detectors may fail to detect certain components or detect components but misclassify them or localize them imprecisely. One common failure mode is "class jitter," where an object is persistent and mostly classified correctly, but occasionally the algorithm's prediction switches. In addition, detectors experience failures in persistence, where the model loses an object that is still in view or miss legitimate object disappearances. Some of these typical failure modes are depicted in Figure 1 below.

Many of these shortcomings are related to object detectors processing frames one-by-one to make decisions. No temporal patterns or associations are considered. For example, if a solar panel is detected in several consecutive frames and then, suddenly, a detection is made in the same region of the image but is labeled a body, there is a high chance it is incorrect, but detectors do not consider the past. Exploiting prior knowledge should make this "class jitter" apparent. Similarly, understanding the trajectory of bounding boxes over time for a single object should help us avoid failures to localize as it moves smoothly over time.

The computer vision task of object tracking takes object detection a step further. Object trackers are designed to make associations between the same objects in subsequent frames, thereby generating tracks followed by said objects through time. We believe object tracking shows potential to help our object detection algorithms overcome the complications discussed above.

Object Detection Algorithms

Modern object detection algorithms (or object detectors) fall into three major categories: vision transformers, multi-stage object detectors, and single-stage object detectors.



Failure to Detect:
The antenna on top of the satellite is missed.

Failure to Localize:
The predicted box around the body is imprecise, missing the bottom half of the satellite.

Failure in Persistence:
Image 1 displays the correct detection of the solar panel; however, the bottom solar panel is "lost" improperly in image 2.

Figure 1. Object Detection Failure Modes

While vision transformer-based object detectors are among the most accurate [20, 21], they require massive computation, and therefore require heavy, high-volume computers that draw a large amounts of power. In short, they are too expensive for spaceflight computers that have limiting volume, weight, and power draw constraints. Multi-stage object detectors typically operate in two stages. Region-based convolutional neural networks (R-CNNs) [22] are some of the most effective multi-stage detectors. However, they are also computationally expensive.

In contrast, single-stage object detectors use a single CNN that simultaneously maps image pixels directly to both the bounding box and class predictions. Notable early methods, such as single shot detectors [23] and the You Only Look Once (YOLO) [24], were much cheaper computationally than R-CNNs but failed to reach similar accuracies. In recent years, the successors to YOLO achieved greater accuracies and have surpassed multi-stage detector performance with much smaller computational cost, permitting much higher framerates. YOLOv8 has been shown to be effective for real-time satellite feature recognition in lab experiments as a result. It has been proven to successfully detect solar panels, bodies, antennas, and thrusters under a variety lighting and motion conditions for the observer and target RSO.

Object Tracking Algorithms

Modern object tracking algorithms follow one of two paradigms: dedicated object tracking and tracking-by-detection. Dedicated object trackers are designed to learn entire object tracks within video sequences as standalone algorithms. Tracking-by-detection trackers function in a two-stage fashion by receiving detections from an object detector and using that information to estimate the motion as a regression problem operating on those bounding boxes and their contents.

The current state of dedicated object trackers makes them not ideal for our application for several reasons. Training them requires many annotated video sequences rather than still images, which are much more difficult to obtain in large numbers. Further, dedicated neural architectures for object tracking are quite large, requiring computational resources unavailable on-board spacecraft. Third, tracking-by-detection algorithms claim state-of-the-art performance on nearly all multi-object tracking benchmarks—notably, the MOTChallenge [25, 26] and DanceTrack [27]datasets.

Since tracking-by-detection algorithms currently offer greater accuracy with less computational costs and easier-to-obtain training data, they are the optimal choice. Tracking-by-detection algorithms have three main parts: (1) an object model that estimates the frame-to-frame motion of each object being tracked, (2) an association model that assigns new object detections to existing object tracks, and (3) a module that creates and destroys object tracks as objects enter and leave the image.

An influential baseline tracking-by-detection algorithm is Simple Online and Realtime Tracking (SORT) [28]. It uses a Kalman filter for the object model. The state of each tracked object is modeled as coordinates of the center of the bounding box, area, aspect ratio, and 1-frame linearized velocity of these values. The association model uses IOU distances between detections in consecutive frames with optimal assignment by the Hungarian algorithm. SORT instantiates new object tracks if a new detection has low or undefined IOU with previous detections, and object tracks are deleted if objects are not detected for a preset number of frames. This tracker is far less computationally expensive than even very fast object detectors, so it has a negligible effect on framerate. Later, DeepSORT [29] introduced a feature extractor CNN pre-trained to map the image pixels within bounding boxes to representation vectors that could be compared. This so-called reID network allowed the model to use appearance features to associate objects through time more effectively, but it introduced computational complexity, resulting in lower effective framerates. Another notable innovation came in ByteTrack [30]. The authors found that typically discarded, low-confidence object detections often correspond to partially occluded objects, temporarily blurred objects, or objects changing size. This allowed ByteTrack to improve its constructed trajectories. Most subsequent high-quality trackers were heavily influenced by these algorithms.

BoTSORT *[31]* is similar to DeepSORT, but it adjusts the Kalman filter state vector to include width and height instead of area and aspect ratio. It allows state-dependent process and measurement noise, adds compensation for rigid camera motion, implements BYTE to enhance its trajectories, and opts for a faster ReID network. Of the two algorithms tested herein, this one is the most expensive.

In this work, two tracking-by-detection algorithms will be tested: ByteTrack and BoTSORT, each with an underlying object detector YOLOv8 trained to detect satellite bodies, antennas, solar panels, and thrusters. Each method draws from SORT, mostly with some variations of the Kalman filters, innovations to the association models, and interpolation schemes for missing frames. Focus lies in improving detection performance with trackers without reducing the YOLOv8 framerate substantially. However, it is uncertain how much performance gain will occur with the low yet effective 4-5 FPS YOLOv8 achieves on spaceflight-like hardware (e.g. Raspberry Pi). This uncertainty is because Kalman filters and association models operate frame-to-frame and relatively large motion may occur between frames that are 0.2-0.25 seconds apart. The effectiveness of object trackers in the low framerate regime is a key question addressed by this work.

3. Datasets

The YOLOv8 object detection algorithm was trained and tested on images of satellite mock-ups obtained in the ORION Facility at Florida Tech's Autonomy Lab. Two videos of satellite mockups were captured under both normal and extreme lighting conditions were taken. In each case, the mockup undergoes a 75% rotation at a constant angular velocity. Each video is approximately 3 minutes and captured at 30 frames per second. The 3692 total image frames captured were manually annotated with bounding boxes and class labels for all solar panels, satellite bodies, antennas, and thrusters. The has 7,437 solar panel, 3692 satellite body, 3652 antenna, and 758 thruster annotations altogether, as summarized in Figure 2. In addition, each component of the satellite mockup is given a unique identifier to permit measurement of the object tracker's ability to quickly re-identify components if they are temporarily non-visible or not detected.

The dataset was split into training and testing datasets. The first 80% of frames of each video were assigned to training dataset while the latter 20% is allocated to test data, specifically with 360 frames each. The rotational motion observed is the same in the two training segments of the videos to ensure the satellite positioning and the camera's viewing angle is substantially different in the test segments. These unseen test segments test the algorithm's ability to generalize to views not previously seen. This is important, as the goal is for a chaser satellite to autonomously detect and localize solar panels, antennas, satellite bodies, and solar panels of a known but non-cooperative satellite.

4. Methods

The experiments performed herein follow the tracking-by-detection paradigm for object tracking. The base object detector in all cases is the YOLOv8 algorithm implementation by Ultralytics. As such, the first step was to train this algorithm to detect solar panels, antennas, thrusters, and satellite bodies in still images via the training dataset described above. All frames of the videos are scaled to 480x480-pixels RGB color images. Various augmentation techniques including random crops, horizontal flips, translations, and random cutout boxes, were used during training. This serves to provide synthetic augmentation to the training data, which improved model generalization to the test data.

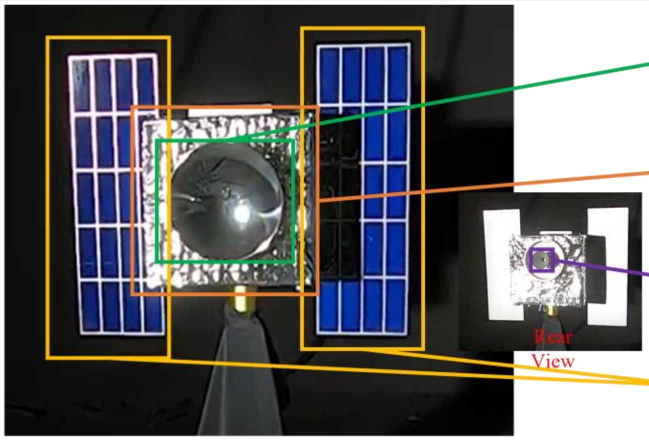
EXAMPLE IMAGE	CLASS	TOTAL ANNOTATIONS
	Antenna	3652
	Body	3692
	Thrusters	758
	Solar Panels	7437

Figure 2. Annotated Satellite Component Dataset

We used the smallest YOLOv8 nano-sized neural architecture, which has 3.2m parameters, requires only 6 MB of memory, and processes images in 8.7B floating point operations (FLOPS), making it suitable for spaceflight hardware. The YOLOv8 nano model upgrades the prior use of the YOLOv5 small model in [12], which provides some substantial improvements in accuracy, runtime/framerates, and memory requirements.

We then used the two object trackers: ByteTrack [30] and BoTSORT [32]. ByteTrack and BoTSORT are packaged with the public YOLOv8 repository by Ultralytics. Each looks at the past detections by the YOLOv8 object detector with the goal of making sense of physical patterns to track features through time, re-identify them when they are temporarily lost by the detector or occluded from the camera, help with the classification predictions.

To evaluate tracker performance for low framerate regimes, the two 360-frame final sequences of the normal and extreme lighting videos are sampled down to framerates of 1, 2, 5, 10, 15, and 30 Hz. In each video and framerate, tracking will be performed with each tracking algorithm (with YOLOv8 detections), and performance will be computed. This will address the largely unanalyzed question of low-framerate object tracking performance of these algorithms.

5. EXPERIMENTS

The performance of each object tracking algorithm will be assessed based on hardware-in-the-loop experiments. Each will be assessed based on the two test cases to determine how well each tracker will perform at tracking solar panels, the satellite body, antenna, and thruster of a satellite mockup under different lighting and motion conditions.

Each case was simulated physically, and videos were recorded at the ORION Facility at the Autonomy Lab at Florida Institute of Technology. The facility is equipped with a hardware-in-the-loop formation flight and docking simulator. It has the ability to simulate motion conditions for a satellite mockup with a body and solar panels including realistic size, configuration, shape, and reflectivity. Orbital lighting effects are simulated effectively with a Hilio Litepanel D12 350W LED light fixture with a 5,600 K color temperature within a room with non-reflective black walls.

Several evaluation metrics will be used to measure the model's ability to track objects through time. These metrics compare model predictions to human-annotated (or ground-truth) bounding boxes and labels in each frame of the test videos.

Two metrics measure the object detector's classification performance: precision is the percentage of correct positive predictions in each class, and recall is the percentage of ground-truth objects from each class correctly classified. To calculate precision and recall, we will refer to classifications as:

- True positives (TP): The correct identification of a ground-truth bounding box.
- False positives (FP): The incorrect identification of an object that is devoid or misplaced.
- False negative (FN): The missed identification of a ground-truth bounding box.

Precision (P) and recall (R) can then be calculated respectively as:

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (1)$$

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (2)$$

The qualities of a good object detector include finding all ground-truths and identifying only the relevant objects. This means the detector should have increasing recall and high precision while the confidence threshold decreases. Average precision (AP) summarizes the behavior of precision and recall that are caused by the changing confidence levels of the bounding boxes. AP is approximately the area under a smoothed precision-recall curve for a single class.

These metrics only measure classification performance and ignore the quality of the predicted bounding boxes. The similarity of two bounding boxes can be measured by the intersection of the bounding box and the ground truth it estimates over its union. This intersection over union (IOU) between overlapping predicted and ground-truth bounding boxes, will be between 0 and 1, where 1 indicates the boxes are identical and 0 indicates no relation. This idea is represented in Figure 3.

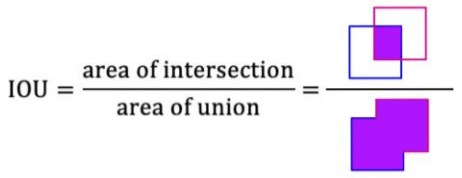


Figure 3. Intersection Over Union (IOU)

To measure the overall performance of our model, in both classification and localization, we use mean average precision (mAP). While AP is calculated for each class, the mAP is the average AP values for all classes. We measure the mAP with an IOU threshold of 0.5 (mAP@0.5). This threshold means a detection will only be considered "correct" if it makes a correct classification and the IOU between the predicted and ground-truth bounding boxes is at least 50%, to ensure good localization by our detector. A second, more comprehensive metric is mAP@0.5:0.95, where the mAP at thresholds of 0.5, 0.55, …, 0.95 is averaged to determine the ability of the detector as we increase the IOU threshold [33].

Finally, we have our metrics to measure how accurately the algorithm tracks targets. The identity switches (IDS or IDSW) metric counts the number of times the algorithm switches between objects. Multi-object tracking accuracy (MOTA) is a metric which combines the false positive rate (FP), false negative rate (FN) and mismatch rate (IDS) relative to the total number of ground truth detections (gtDet) into a single number giving an overall quantification of the tracking accuracy performance [34].

$$MOTA = 1 - \frac{|FN| + |FP| + |IDS|}{|gtDet|} \tag{3}$$

The identification metric IDF1 looks at the accuracy of associating the same object detected across all frames. It determines present trajectories by calculating the bijective mapping of ground truth trajectories (gtTrajs) to predicted trajectories (prTraj). It is computed as the ratio of identity true positives (IDTP) to the average number of IDTP, identity false negatives (IDFN) and identity false positives (IDFP). As such it can be seen as a combination of identity precision (IDP) and identity recall (IDR).

$$IDR = \frac{|IDTP|}{|IDTP| + |IDFN|} \tag{4}$$

$$IDP = \frac{|IDTP|}{|IDTP| + |IDFP|} \tag{5}$$

$$IDF1 = \frac{|IDTP|}{|IDTP| + 0.5|IDFN| + 0.5|IDFP|} \tag{6}$$

A major goal is to develop a satellite component tracking system to enable autonomous on-orbit proximity operations with non-cooperative spacecraft. As such, the computer vision algorithms are not allowed to view the satellite mockup at orientations appearing in the test set during training.

In the first experiment, we measure the baseline performance of the trained YOLOv8 model on the two test cases, wherein the detector had to identify the target satellite's body, two solar panels, antenna, and thruster. These results are shown in Table 1 below.

Case 1 corresponds to the normal lighting conditions while Case 2 corresponds to the extreme lighting condition scenario in all forthcoming comparisons.

Table 1: YOLOv8 Model Baseline Performance (All Classes)

Case	P	R	mAP @0.5	mAP@0.5:0.95
1	0.974	0.927	0.957	0.628
2	0.786	0.740	0.755	0.435

Here, we see the model performs worse but still quite well in the extreme lighting of Case 2. However, these average metrics across the classes only tell part of the story. In Table 2 below, we see the class-specific mAP@0.5.

The number of detections made by the Yolov5 model was recorded and then compared to the number of detections made when incorporating the object tracking algorithms. This is shown in Tables 2 through 5 below.

Table 2. YOLOv8 Model Baseline Performance (Class-specific mAP@0.5)

Case	Antenna	Body	Solar	Thruster
1	0.995	0.995	0.847	0.911
2	0.563	0.995	0.799	0.663

As we see, the baseline YOLOv8 object detector shows excellent performance in the scenario, although it shows some struggles with smaller features like antennas and thrusters under the extreme lighting conditions of Case 2. However, even mAP above 0.5 is generally considered to be good in the object detection literature, especially with such a small YOLOv8 nano architecture.

Next, we move on to analyzing the performance of the tracker via the MOTA and IDF1 metrics. We present performance of both ByteTrack and BoTSORT on each of the two test cases and at test video framerates 1, 2, 5, 10, 15, and 30 Hz in Table 3 as well as Figure 4 and Figure 5

Table 3: Tracker Performance Comparison for Different Framerates and Test Cases

		MOTA		IDF1	
Case	**Hz**	**Byte**	**BoT**	**Byte**	**BoT**
1	1	**0.811**	**0.811**	**0.906**	**0.906**
1	2	**0.868**	0.849	**0.934**	**0.934**
1	5	**0.864**	0.853	0.932	**0.936**
1	10	0.864	**0.866**	0.932	**0.940**
1	15	**0.880**	0.873	0.939	**0.940**
1	30	**0.884**	0.861	0.942	0.942
2	1	0.593	**0.611**	**0.778**	**0.778**
2	2	0.648	**0.704**	0.815	**0.843**
2	5	0.757	**0.784**	0.884	**0.900**
2	10	**0.756**	0.746	0.908	**0.912**
2	15	**0.739**	0.710	**0.899**	0.894
2	30	**0.730**	0.722	**0.908**	0.903

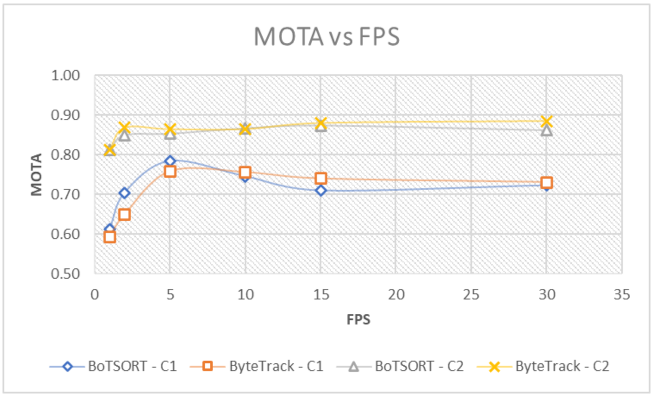


Figure 4 MOTA vs FPS comparison for both trackers and cases

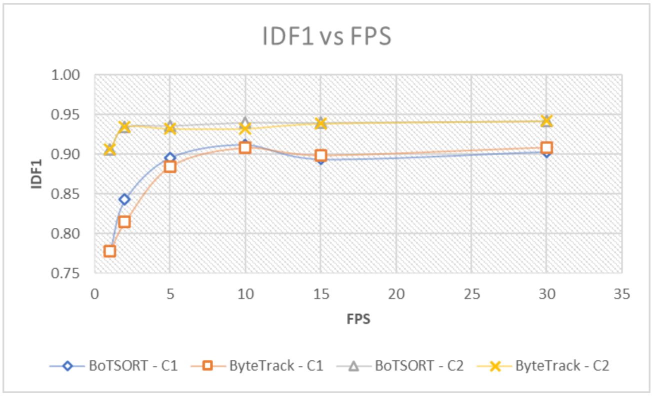


Figure 5 IDF1 vs FPS comparison for both trackers and cases

These results show that implementing the tracking algorithms alongside our base YOLOv8 object detector results in very high tracking performance. BoTSORT seems to have a slight advantage in the IDF1 metrics at framerates of 5 to 10 Hz with nearly identical performances at higher or lower framerates. BoTSORT also outperforms ByteTrack in extreme lighting with low framerates by both metrics. ByteTrack performs better under moderate lighting, generally having higher MOTA with competitive IDF1 scores. It is also noted that tracking attained optimal or near optimal performance by 5 Hz after which performance plateaued. This indicates that tracking can be implemented successfully at these lower frame rates, which is ideal for on orbit servising use case wherin there is significant computational and power limitations.

All in all, it seems the two trackers perform quite similarly by tracking quality metrics. However, ByteTrack is substantially less computationally expensive. In fact, the authors of BoTSORT report framerates of 29.6 Hz for ByteTrack and 6.6 Hz for BoTSORT. As such, we can conclude ByteTrack should be preferred for real-time satellite component tracking if we intend the algorithms to run onboard.

6. CONCLUSIONS

Overall, it was found that implementing object trackers ByteTrack and BoTSORT both provide reliable tracking of satellite components when used with a high-performance object detector YOLOv8. We also demonstrate the intuitive finding that both object detection and object tracking are substantially more challenging in the case of extreme lighting, but the models presented herein still exhibit high performance in such cases.

A unique finding of this article is that these both ByteTrack and BoTSORT trackers perform well in tandem with a high-accuracy object detector even with low inference framerates - near optimal performance from as low as 5 Hz - a surprising result largely not claimed to be a strength by the developers of the object trackers, as they tend to recommend use with object detectors running at a minimum of 30 Hz. This indicates tracking-by-detection paradigm is promising for use on resource-constrained spaceflight hardware, where 30 Hz object detection is challenging to achieve if not entirely infeasible.

While ByteTrack and BoTSORT are shown to perform similarly in tracking quality while ByteTrack has far less computational cost and is hence preferred for onboard applications, as it can be deployed on edge computers such as Raspberry Pi or other resource-limited computers feasible for use in satellite and other space applications.

ACKNOWLEDGEMENTS

The work on this study was supported by AFWERX STTR Phase II contract FA864921P1506 and the NVIDIA Applied Research Accelerator Program.

REFERENCES

[1] NASA and M. Garcia, "Space debris and human spacecraft," May 2021. [Online]. Available: https://-www.nasa.gov/mission_pages/station/news/-orbital_debris.html

[2] J. L. Goodman, "History of space shuttle rendezvous and proximity operations," *Journal of Spacecraft and Rockets*, vol. 43, no. 5, pp. 944–959, sep 2006.

[3] K. Yoshida, "Engineering test satellite VII flight experiments for space robot dynamics and control: Theories on laboratory test beds ten years ago, now in orbit," *The International Journal of Robotics Research*, vol. 22, no. 5, pp. 321–335, may 2003.

[4] R. T. Howard and T. C. Bryan, "DART AVGS flight results," in *SPIE Proceedings*, R. T. Howard and R. D. Richards, Eds. SPIE, apr 2007.

[5] T. M. Davis and D. Melanson, "XSS-10 microsatellite flight demonstration program results," in *SPIE Proceedings*, J. Peter Tchoryk and M. Wright, Eds. SPIE, aug 2004.

[6] Air Force Research Laboratory, "Xss-11 micro satellite," Jul. 2011. [Online]. Available: https://-www.kirtland.af.mil/Portals/52/documents/AFD-111103-035.pdf?ver=2016-06-28-110256-797

[7] INTELSAT. (2022, May) MEV-1 and Intelsat 901. [Online]. Available: https://www.intelsat.com/resources/-blog/mev-1-a-look-back-at-intelsats-groundbreaking-journey/

[8] J. Rainbow. (2022, Apr.) Mev-2 servicer successfully docks to live INTELSAT satellite. https://spacenews.com/mev-2-servicer-successfully-docks-to-live-intelsat-satellite/.

[9] M. Wilde, B. Kaplinger, T. Go, H. Gutierrez, and D. Kirk, "ORION: A simulation environment for spacecraft formation flight, capture, and orbital robotics," in *2016 IEEE Aerospace Conference (AERO)*. IEEE, mar 2016.

[10] G. Jocher, Ayush Chaurasia, A. Stoken, J. Borovec, NanoCode012, Yonghye Kwon, TaoXie, Kalen Michael, Jiacong Fang, Imyhxy, , Lorna, C. Wong, æ›¾é€¸å¤«(Zeng Yifu), Abhiram V, D. Montes, Zhiqiang Wang, C. Fati, Jebastin Nadar, Laughing, UnglvKitDe, Tkianai, YxNONG, P. Skalski, A. Hogan, M. Strobel, M. Jain, L. Mammana, and Xylieong, "ultralytics/yolov5: v6.2 - yolov5 classification models, apple m1, reproducibility, clearml and deci.ai integrations," 2022.

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015. [Online]. Available: https://proceedings.neurips.cc/paper/-2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf

[12] T. Mahendrakar, A. Ekblad, N. Fischer, R. White, M. Wilde, B. Kish, and I. Silver, "Performance study of YOLOv5 and faster r-CNN for autonomous navigation around non-cooperative targets," in *2022 IEEE Aerospace Conference (AERO)*. IEEE, mar 2022.

[13] T. Mahendrakar, M. N. Attzs, A. Tisaranni, J. Duarte, R. T. White, and M. Wilde, "Impact of intra-class variance on yolov5 model performance for autonomous navigation around non-cooperative targets," in *AIAA Scitech 2023 Forum (accepted)*, 2023.

[14] J. Cutler, M. Wilde, A. Rivkin, B. Kish, and I. Silver, "Artificial potential field guidance for capture of non-cooperative target objects by chaser swarms," in *2022 IEEE Aerospace Conference (AERO)*. IEEE, mar 2022.

[15] T. Mahendrakar, J. Cutler, N. Fischer, A. Rivkin, A. Ekblad, K. Watkins, M. Wilde, R. White, B. Kish, and I. Silver, "Use of artificial intelligence for feature recognition and flightpath planning around non-cooperative resident space objects," in *AIAA ASCEND 2021*. American Institute of Aeronautics and Astronautics, nov 2021.

[16] T. Mahendrakar, S. Holmberg, A. Ekblad, E. Conti, R. T. White, M. Wilde, and I. Silver, "Autonomous rendezvous with non-cooperative target objects with swarm chasers and observers," in *33rd AAS/AIAA Space Flight Mechanics Meeting (submitted)*, 2023.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/-2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[18] T. Mahendrakar, R. T. White, M. Wilde, B. Kish, and I. Silver, "Real-time satellite component recognition with YOLOv5," in *Proceedings of the Small Satellite Conference*, 2021. [Online]. Available: https://-digitalcommons.usu.edu/smallsat/2021/all2021/51/

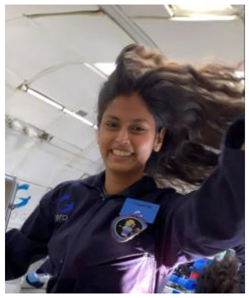
[19] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 10012–10022.

[20] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, F. Wei, and B. Guo, "Swin transformer v2: Scaling up capacity and resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 12009–12019.

[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.
[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “SSD: Single shot MultiBox detector,” in *Computer Vision ECCV 2016*. Springer International Publishing, 2016, pp. 21–37.
[23] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
[24] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, “MOT16: A benchmark for multi-object tracking,” Mar. 2016.
[25] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé, “Mot20: A benchmark for multi object tracking in crowded scenes,” Mar. 2020.
[26] P. Sun, J. Cao, Y. Jiang, Z. Yuan, S. Bai, K. Kitani, and P. Luo, “Dancetrack: Multi-object tracking in uniform appearance and diverse motion,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 20993–21002.
[27] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2016.
[28] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2017.
[29] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, “Bytetrack: Multi-object tracking by associating every detection box,” *European Conference on Computer Vision*, Oct. 2022.
[30] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, “Bot-sort: Robust associations multi-pedestrian tracking,” Jun. 2022.
[31] Y. Du, Y. Song, B. Yang, and Y. Zhao, “Strongsort: Make deepsort great again,” Feb. 2022.
[32] J. Cao, X. Weng, R. Khirodkar, J. Pang, and K. Kitani, “Observation-centric sort: Rethinking sort for robust multi-object tracking,” Mar. 2022.
[33] M. BrostrÃ¶m, “Real-time multi-camera multi-object tracker using yolov5 and strongsort with osnet,” https://github.com/mikel-brostrom/-Yolov5_StrongSORT_OSNet, 2022.
[34] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, “A comparative analysis of object detection metrics with a companion open-source toolkit,” *Electronics*, vol. 10, no. 3, p. 279, jan 2021.

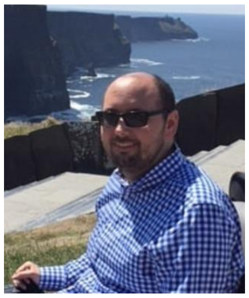
BIOGRAPHY

Monty Nehru Attzs *received a B.S. and M.S. in Aerospace Engineering from The Florida Institute of Technology, Melbourne. He is currently working towards a Ph.D. in Operations Research at the Florida Institute of Technology, Melbourne. His current research involves the development of lightweight object tracking by detection models for implementation on edge hardware.*



Trupti Mahendrakar *received a B.S. in Aerospace Engineering from Embry-Riddle Aeronautical University, Prescott, Arizona in 2019 and an M.S, in Aerospace Engineering from Florida Institute of Technology, Melbourne. She is currently a Ph.D. candidate at the Florida Institute of Technology, Melbourne. Her current research includes the implementation of machine vision algorithms to enhance on-orbit service satellite operations, optimization of cold gas thruster design, and implementation of robotic manipulators for satellite refueling.*

Mackenzie J. Meni *received a B.S. in Mathematics and is a PhD candidate at Florida Institute of Technology. She has two years of experience as a software developer with the USGS. Currently, she works with the US Army Corps of Engineers analyzing the internal workings of deep learning models to increase model interpretability, identify biases, and develop more principled neural architectures and learning techniques. In addition, she is a data science intern for Crimson Phoenix working on object detection and deployment.*



Dr. Ryan T. White *is an Assistant Professor of Mathematics at Florida Tech and Director of the NEural TransmissionS (NETS) Lab. His research focuses on computer vision, physics-inspired machine learning, synthetic data generation, and probability. His work includes multidisciplinary projects on in-space use of computer vision to support on-orbit proximity operations, medical data analytics, and glaciology. Dr. White earned a Ph.D. from Florida Tech and joined the Florida Tech faculty in 2015.*

Dr. Isaac Silver *is the CEO of Energy Management Aerospace. He earned his Ph.D. from Florida Tech in Space Sciences. He also holds a B.S. in Astronomy and Astrophysics from Florida Tech. He's an Airline Transport Pilot (Airplane Multi-Engine Land), Commercial Pilot (Airplane Single-Engine Land and Sea), Gold Seal Flight Instructor (Single and Multi-Engine), Instrument Pilot (Airplane) and Ground Instructor (Advanced). He has 17,500 hours of flight time with more than 4,000 hours as an instructor. Aircraft include DA-10, DA-20, DA-50/900, L-1329, BE-350, BE-400, IAI-1124, Learjet (24/25/31/35), and L39.*