Seeing Stars: Learned Star Localization for Narrow-Field Astrometry

Violet Felt
United States Space Force
violet.felt.1@spaceforce.mil

Justin Fletcher
United States Space Force
550 Lipoa Parkway, Kihei, HI
justin.fletcher.14.ctr@spaceforce.mil

Abstract

Star localization in astronomical imagery is a computer vision task that underpins satellite tracking. Astronomical star extraction techniques often struggle to detect stars when applied to satellite tracking imagery due to the narrower fields of view and rate track observational modes of satellite tracking telescopes. We present a large dataset of real narrow-field rate-tracked imagery with ground truth stars, created using a combination of existing star detection techniques, an astrometric engine, and a star catalog. We train three state of the art object detection, instance segmentation, and line segment detection models on this dataset and evaluate them with object-wise, pixel-wise, and astrometric metrics. Our proposed approaches require no metadata; when paired with a lost-in-space astrometric engine, they find astrometric fits based solely on uncorrected image pixels. Experimental results on real data indicate the effectiveness of learned star detection: we report astrometric fit rates over double that of classical star detection algorithms, improved dim star recall, and comparable star localization residuals.

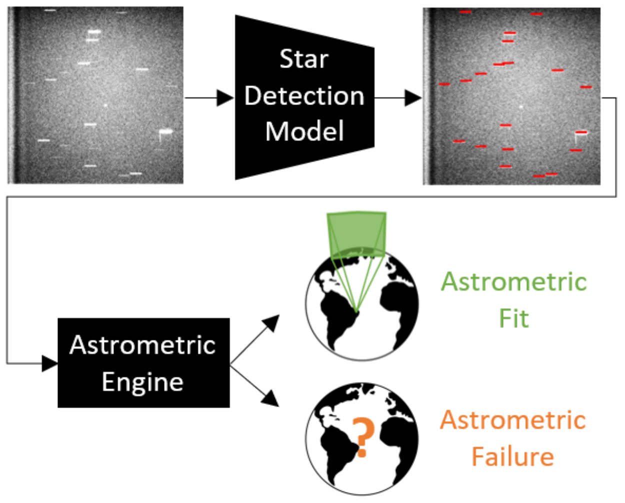


Figure 1. The astrometric image processing pipeline. A star detection model accepts an astronomical image as input and produces detected star positions. The detected star positions are passed to an astrometric engine, which uses the ratios between the star positions to match detected stars to real world catalog stars. The output of this pipeline is either an astrometric fit (if the detected stars could be matched to the celestial sphere) or an astrometric failure (if the detected stars could not).

1. Introduction

Human activity in space is rapidly accelerating [31]. With this orbital object population growth comes heightened challenges for human activity in space [2]; among these are collision avoidance, debris tracking, and safety of human space flight. The burgeoning ranks of anthropogenic Earth satellites (e.g., commercial megaconstellations [28]) have driven a commensurate increase in the number of ground-based astronomical telescopes with which to track them [3, 15]. However, the computer vision applications through which the operators of these telescopes perceive the space environment are largely adapted from astronomical data processing. These algorithms require expert adaptation to each telescope, and are prone to error when improperly calibrated.

The adaptation of astronomical techniques to satellite tracking yields sub-optimal object detection performance due to misalignment between the content of astronomical optical images and those collected for satellite tracking [15]. Both the narrower fields of view and rate track observational modes reduce precision and recall for star source extraction, resulting in failed astronomical alignment and, ultimately, satellite tracking failure. Prior work has improved source extraction for satellite tracking [16], but astrometric localization of the detected satellites remains a limiting factor for end-to-end performance.

In this work, we propose a learned approach to star detection in narrow-field rate-track imagery. We show that this approach outperforms two baseline star source extraction methods across all metrics. Our star source extraction models can be paired with a lost-in-space algorithm to produce an astrometric fit for an image with no input other than the uncorrected image pixels, as shown in Figure 1. This differs from traditional astrometric image processing pipelines, which require dark and flat field corrections in the star detection step as well as scale and pointing information in the astrometric engine step [37]. We contribute:

- StarNet: a real dataset of 64k narrow-field rate-tracked images and their corresponding 4.5 million stars, extracted from a star catalog.
- Three novel learned approaches to the star source extraction task leveraging object detection, instance segmentation, and line segment detection in which star locations are predicted as bounding boxes, pixel-wise masks, and line segments, respectively.
- A framework for evaluating star detection solutions including object-wise metrics, pixel-wise metrics, and astrometric fit rate.

2. Related Work

Astrometric processing comprises the extraction (i.e., detection) of relevant light sources from astronomical imagery and the the positioning of those sources with respect to the fixed celestial background (i.e., astrometric fitting), as shown in Figure 1. In this section, we review source extraction methods for stars, various approaches to astrometry, learned computer vision techniques that are applicable to our source extraction problem, and conclude with a discussion of recent work in deep learning for Space Domain Awareness (SDA).

2.1. Star Source Extraction Methods

Prior works in source extraction employ hand-crafted features and rule-based systems, and are often optimized for one specific sensor or telescope [13, 29]. Some rely on image metadata such as telescope track rate and exposure time, limiting their generalization capabilities [24, 37]. Recent forays into learned computer vision techniques have revealed the applicability of deep learning to source extraction, and include object detection neural networks for stars [5, 22], galaxies [5, 20], and satellites [16, 22, 38], as well as semantic segmentation neural networks for stars [11, 42], galaxies [33], and satellites [39].

For star extraction, these studies have relied upon small, often simulated, training datasets [5, 11] consisting of wide-field-of-view (WFOV), sidereal (i.e., the sensor is slewed to match the apparent movement of the stars) images. In the cases where real data is used [22, 42], ground truth star annotations are produced by traditional computer vision techniques or human annotation, both of which are imperfect approximations for ground truth star locations.

In order for a star extraction method to be a robust solution for the satellite tracking community, it must be able to detect stars when telescope track rates deviate from sidereal and stars appear as star streaks instead of point sources. Few classical source extraction methods have this capacity [13, 24], and no published deep learning methods do [22, 42]. In this work, we propose a method that addresses this gap. The foundation of our work is a large, real dataset of narrow-field, rate-track images. We pair an astrometric engine with a star catalog to ensure that every real star in an image is labeled in the ground truth.

2.2. Astrometry Methods

Astrometry is an astronomical data processing task in which the positions of celestial bodies (extracted by an image processing algorithm or deep learning model) are matched to a portion of the celestial sphere. Astrometry can be broadly divided into lost-in-space algorithms and recursive algorithms, both of which produce astrometric metadata for astronomical images [36]. Lost-in-space algorithms autonomously identify the stars present in an image without any initial image metadata [23], while recursive astrometric algorithms leverage image metadata (an initial pointing guess and an approximate scale) for the same task [37].

A typical astrometric pipeline consists of a pattern-based feature extraction algorithm (which transforms the extracted stars in an image into a unique object such as a geometric shape, matrix, or string), a catalog search step (typically of $O(n)$ complexity where n is the number of star patterns being searched through), and occasionally a validation step (such as a voting mechanism or a Bayesian decision process) [36].

One publicly-available lost-in-space algorithm is astrometry.net, which was designed by an interdisciplinary team of astronomers and software developers to be robust to false extracted stars and shifted star centroids, both of which are common failure modes for astrometric engines [23, 36]. With rigorously documented scalability and fidelity, astrometry.net is the astrometric engine of choice for this work and is discussed further in Section 4.2.

2.3. Deep Learning Computer Vision Techniques

Three established deep learning problem formulations are well-suited to the task of star localization: object detection, instance segmentation, and line segment detection.

Recent developments in object detection demonstrate the effectiveness of one-stage object detectors such as YOLOX [19, 34] and transformer-based object detectors such as Deformable DETR [6, 43]. While architectures such as Faster RCNN [35] and RetinaNet [26] continue to perform well, these newer architectures eliminate the need for hand-designed components such as non-maximum suppression and anchor generation, increasing their applicability to new problem domains. Deformable DETR is particularly well-suited for detecting small objects [43], like stars in astronomical imagery.

Instance segmentation developments tend to follow advances in object detection, with Mask RCNN evolving Faster RCNN [21] and Mask2Former extending DETR [9] to predict masks and bounding boxes in parallel. Other models such as HTC [7] and QueryInst [12] interleave the bounding box regression and mask prediction tasks in a self-attention manner, enabling communication between the tasks to integrate complementary features. HTC incorporates a semantic segmentation branch to distinguish faint objects from noisy background [7], useful for dim stars in astronomical imagery.

Recent developments in line segment detection (an offshoot of wireframe parsing) shift deep learning approaches from detecting line segment proposals and junctions separately to treating line segments as objects [10, 41]. LETR is the current state of the art architecture, modifying DETR to predict line segment endpoints and utilizing a direct endpoint L_1 loss to avoid IoU issues when line segments are horizontal or vertical [41], a common occurrence for stars in astronomical imagery.

2.4. Deep Learning for Space Domain Awareness

SDA entails the detection, astrometric localization, identification, and characterization of artificial satellites. Deep learning approaches to satellite detection [14, 16, 17], identification [18, 30], and characterization [32] have been explored, but astrometric localization still relies on physics based methods.

Astrometric localization is a process in which the stars in an image are extracted, then matched to a set of real world stars in a process called astrometric fitting, as shown in Figure 1. This allows for accurate localization of any satellites in the image, down to arcseconds of precision. A satellite can only be localized, and subsequently tracked, if the astrometric fitting process is successful.

The zoo of learned star localization models proposed in this work are a first step towards a deep learning solution for satellite localization, as high-precision and high-sensitivity star detection enables astrometric fitting, and by extension high-accuracy satellite localization.

3. Approach

3.1. A Task Formulation for Star Detection

The objective of astrometric image processing is to produce an astrometric fit for an image. This binary measurement of astrometric fit/failure is not conducive to learning, so we substitute the astrometric processing task with the star localization task. If every ground truth star in a given image is extracted, then that image will have an astrometric fit.

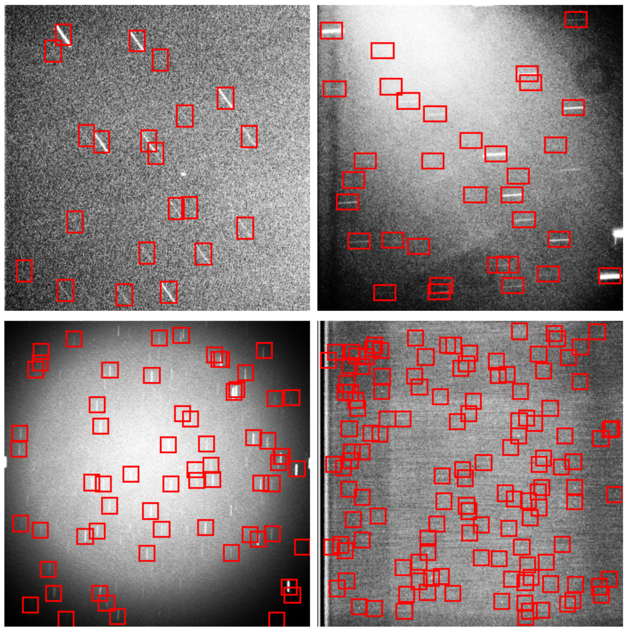


Figure 2. Sample StarNet images and their corresponding ground truth stars represented as bounding boxes, demonstrating a range of star streak characteristics and background noise patterns.

The task of localizing an unknown number of stars with unknown streak lengths and angles in noisy electro-optical imagery naturally lends itself to a deep learning computer vision solution. Stars can intuitively be represented as bounding boxes for object detection models, pixel-wise masks for instance segmentation models, and line segments for line segment detection models.

3.2. Creating Training Data

The process of creating our StarNet training dataset is detailed in Section 4. Satellite tracking images collected over a period of four years are run through the astrometric imaging pipeline shown in Figure 1. Four classical star detection methods detailed in Section 4.1 are used to extract stars: SExtractor, AstroGraph, human annotation, and model ensemble bootstrapping. These extracted stars are submitted for astrometric fitting using the astrometry.net algorithm described in Section 4.2. If astrometric fitting is successful, the astrometric fit is paired with the SSTRC7 star catalog defined in Section 4.3 to extract all real stars in the image. Finally, the ground truth stars are represented in one of three formats suitable for deep learning (bounding boxes, pixel-wise masks, and line segments) in Section 4.4.

3.3. Training Models

Nine deep learning models are trained on the StarNet dataset. We select the object detection models Deformable DETR, Faster RCNN, RetinaNet, YOLOX and the instance segmentation models Mask2Former, Mask RCNN, QueryInst, HTC as they span a wide variety of model architectures and loss functions. Additionally, we select the line segment detection model LETR as a more specialized model that aligns well with the task of star streak detection. All models are described in Section 2.3. The top-performing models (Deformable DETR, HTC, and LETR) undergo rigorous comparison to baseline star detection methods in Section 5.

Every model architecture is created in pytorch [27] and modified to detect one class (star) and a maximum of 1000 objects (stars), then trained on the train partition of StarNet using an NVIDIA DGX system with 4 GPUs. Models are instantiated with a pre-trained R-50 backbone and trained for 500 epochs with a total batch size of 64 [8]. The number of model parameters and floating-point operations (FLOPs) for the top-performing models is detailed in Table 2.

3.4. Evaluating Models

To frame the star detection task within the SDA community, we prioritize sensitivity (recall) and precision when evaluating our deep learning models on the test partition of StarNet. High-sensitivity star detection is critical for detecting enough stars per narrow-field image to find an astrometric fit. High-precision star detection is essential for achieving small star localization residuals, and by extension small satellite localization residuals.

We compare the trained deep learning models to SExtractor [4] (a widely-used classical star extraction method within the astronomy community) and AstroGraph (a newer classical star extraction method designed for narrow-field rate-tracked imagery [37]). The metrics precision, recall, and F_1 score are used to quantify deep learning model performance against SExtractor and AstroGraph on both object-wise and pixel-wise scales in Sections 5.1 and 5.2, aligning with previous work in astrometric source detection [16]. We compare astrometric fit rates in Section 5.3, using the astrometric engine astrometry.net. To further understand star recall, we analyze star recall in relation to the number, length, and magnitudes of stars in Section 5.4. To further understand star precision, we decompose star residuals into RA residuals and Dec residuals in world space in Section 5.5. Finally, we address model and pipeline computational complexity in Section 5.6.

4. StarNet Dataset

The StarNet dataset consists of 64,668 images and their corresponding 4.5 million stars. StarNet images are captured in rate-track mode against low earth orbit (LEO), medium earth orbit (MEO), and geosynchronous equatorial orbit (GEO) targets by four sensors at three geographic locations. The images range in size from 512×512 pixels to 1024×1024 pixels, in field of view (FOV) from 0.3 degrees to 0.9 degrees, and in instantaneous field of view (IFOV) from 2.0-4.2 arcseconds per pixel. The star streaks in these images range in angle from 0 to 360 degrees, in length from 0 to 270 pixels, and in quantity from 9-500 stars per image. Sample StarNet images are shown in Figure 2.

4.1. Detecting Stars

SExtractor is a well-known algorithm for detecting stars in astronomical images. It uses a series of image processing steps such as background subtraction, filtering, segmentation, de-blending, and pruning to localize stars, each represented by a barycenter in pixel space and an isophotal ellipse enclosing the detected star. SExtractor is optimized for detecting point sources in sidereal imagery and can struggle detecting long star streaks in rate-track imagery, producing multiple detections per star streak [4].

AstroGraph is a comparatively recent star detection approach that entails a series of steps including dark and flat field corrections, background subtraction, thresholding, clustering, template matching, and filtering to generate a list of detected stars, defined by barycenters in pixel space. AstroGraph leverages image metadata to estimate star streak length and angle, enabling it to reliably detect long star streaks. Its ability to detect dim stars can be impacted by the absence of high-quality dark and flat frames, as is the case for the observations in our dataset [37].

Human annotation is accomplished by a team of trained analysts that label stars in astronomical images. Due to the high number of stars per image, we introduce an iterative process in which an annotator labels 10 stars, the image is submitted for astrometric fitting, and if the image cannot be fit the annotator labels 10 more stars. This cycle is repeated until the image is fit, or there are no more visible stars to label. Labeling a star consists of annotating both endpoints of a star streak. This process yields a throughput of approximately 30 images per annotator per hour.

Model ensemble bootstrapping is used to augment StarNet. The four object detection models detailed in this work (Deformable DETR, Faster RCNN, RetinaNet, YOLOX) are trained on 10k images of StarNet. If all trained models predict a star within 4 pixels of each other, the average pixel location of the predictions is included in the set of stars submitted for astrometric fitting. In this approach, we are using the astrometric fit process as an oracle. We acknowledge the potential bias introduced when using model predictions to populate the training dataset. However, by including the astrometric fitting and star catalog steps in the data production pipeline we keep astronomy “in the loop,” thereby ensuring that every star included in the ground truth is a bona fide star.

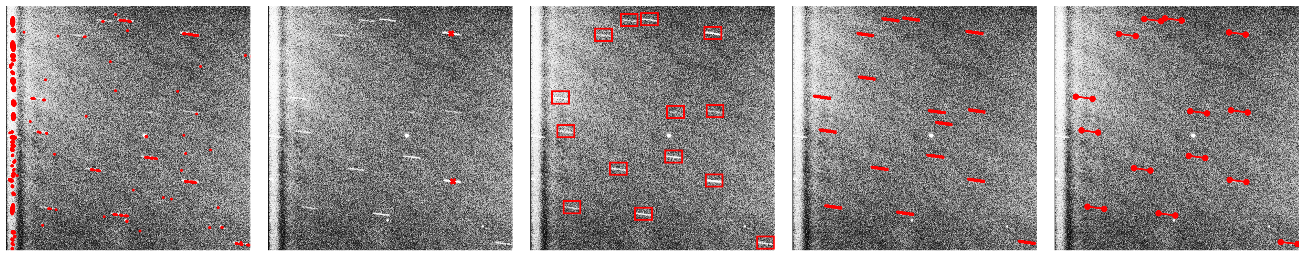


Figure 3. A random StarNet image with star detections. From left to right: SExtractor isophotal ellipses, AstroGraph centroids, Deformable DETR bounding boxes, HTC instance segmentations, and LETR line segments.

4.2. Astrometric Fitting: Astrometry.net

Astrometry.net is an astrometric engine that uses the pixel locations of stars to estimate the pointing, scale, and orientation of an image. It encodes that information as a World Coordinate System (WCS), which describes the geometric transformation between pixel space and world space using a reference point and a matrix. The astrometry.net algorithm operates by geometrically hashing sets of four stars (known as a "quad") and comparing them to pre-indexed hashes built from star catalogs. Matching quads are verified through a Bayesian decision process, weighted to avoid false positive astrometric fits. We use the provided 4200 star index, constructed from the 2MASS catalog [23].

4.3. Star Extraction from Catalog

The WCS generated by astrometry.net is paired with the SSTRC7 star catalog to extract all of the real stars in an image. SSTRC7 merges many catalogs including GAIA-DR2, Tycho-2, and 2MASS to form a comprehensive catalog with over 1.6 billion stars with a maximum visual magnitude of 18 [37]. Each extracted star is represented by a star centroid in pixel space and a visual magnitude. We include all stars with a visual magnitude of less than 15 in the ground truth, yielding an average of 73 stars per image. We ensure every image in StarNet can be astrometrically fit using only the stars in the ground truth.

4.4. Representing Stars for Deep Learning

Bounding boxes are created for object detection models. Star streak length and angle information are extracted from image metadata to construct a bounding box that encompasses the star streak. Bounding boxes are then padded to ensure they are at least 5% of the size of the image. A representation without padding (with bounding boxes as small as 1 pixel x 1 pixel) was explored but resulted in a significant drop in model recall.

Pixel-wise masks are created for instance segmentation models. Star streak length and angle information are extracted from image metadata to construct a 1-pixel-wide star streak connecting the endpoints of the streak, then this streak is dilated three times to account for the point-spread function [24]. A semantic segmentation representation was evaluated but resulted in a marked decrease in astrometric fit rate due to overlapping star streaks.

Line segments are created for line segment detection models. Star streak length and angle information are extracted from image metadata to construct a line segment connecting the top-left endpoint of the streak to the bottom-right endpoint of the streak. An alternative representation connecting the "start time" streak endpoint to the "end time" streak endpoint was investigated but resulted in a substantial decline in model precision.

5. Experiments

5.1. Object-Based Metrics

We match ground truth star centers to detected star centers by computing the Euclidean distance and thresholding at 8 pixels, aligning with previous work in deep learning for SDA [16]. Although intersection-over-union (IoU) is a widely-used metric for matching, it is less relevant in the SDA domain where the objective is to find star centroids, and it is not applicable to line segment detection. A star centroid is defined as the position of a source for SExtractor and AstroGraph, the center of a bounding box in object detection, the geometric center of a segmentation mask in instance segmentation, and the midpoint of a line segment in line segment detection. The object-based metrics are shown in Table 1. HTC has the highest precision and LETR has the highest recall, while Deformable DETR represents a balance between both with the highest F_1 score. SExtractor noticeably struggles with precision (identifying multiple stars along the same streak) and AstroGraph noticeably struggles with recall (only identifying the brightest/least-noisy stars), as shown in Figure 3.

5.2. Pixel-Based Metrics

For star detection methods that make pixel-wise predictions, we match ground truth star pixels to detected star

Table 1. Comparing performance metrics on the StarNet test partition. Best metrics are bolded.

Star Det. Method	Object-Based Metrics			Pixel-Based Metrics			Astrometric Metrics	
	Precision	*Recall*	*F1*	*Precision*	*Recall*	*F1*	*Fit Rate*	*False Fit Rate*
SExtractor (Baseline)	0.29	0.46	0.35	0.28	0.20	0.23	0.26	0.05
AstroGraph (Baseline)	0.89	0.27	0.41	-	-	-	0.38	0.00
Deformable DETR **(ours)**	0.89	0.70	**0.78**	-	-	-	**0.96**	0.00
HTC **(ours)**	**0.90**	0.59	0.71	**0.84**	**0.56**	**0.67**	0.96	0.00
LETR **(ours)**	0.80	**0.75**	0.77	0.15	0.48	0.22	0.96	0.00

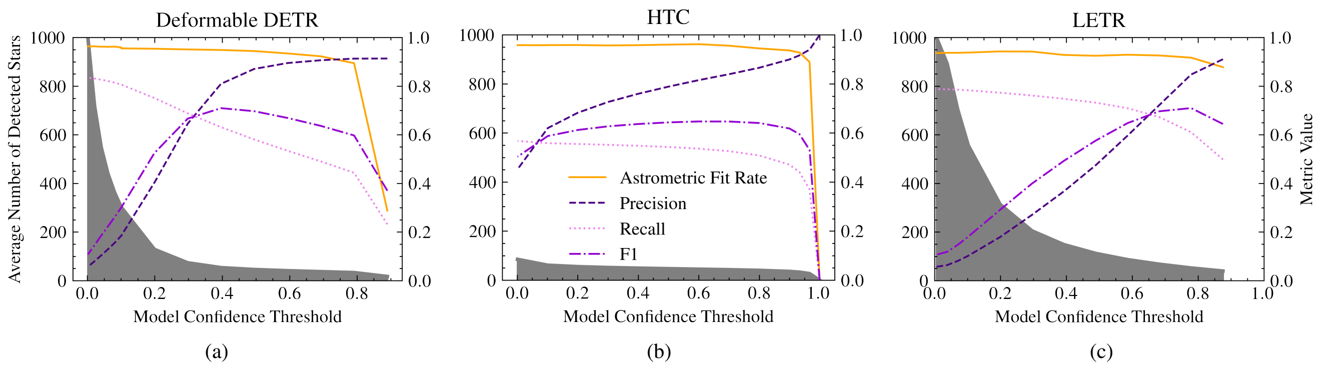


Figure 4. Comparing astrometric fit rate to model performance metrics for (a) Deformable DETR (b) HTC (c) LETR. The average number of detected stars at a given confidence threshold is shown in gray.

pixels by comparing heatmaps. For SExtractor, we construct isophotal ellipses and rasterize them into a heatmap. For instance segmentation, we collapse individual segmentation maps into a semantic segmentation map. For line segment detection, we rasterize line segments and dilate them to match segmentation ground truth, aligning with previous line segment detection work [41]. The pixel-based metrics are shown in Table 1. The instance segmentation model HTC exhibits the highest pixel-wise performance across precision, recall, and F_1; a logical result as this model was the only one trained to make pixel-accurate predictions.

5.3. Astrometric Fit Rate

An astrometric fit is the ultimate goal of the astrometric image processing pipeline. Achieving an astrometric fit means a star detection method has detected enough stars (high recall) precisely enough (high precision) to match said stars to a portion of the celestial sphere. The astrometric fit rate of each star detection method is displayed in Table 1. All deep learning models have similarly high astrometric fit rates, while SExtractor's fit rate suffers due to low precision and AstroGraph's fit rate suffers due to low recall.

A false astrometric fit occurs when the WCS reference point created by astrometry.net differs by more than 1 degree from the pointing information in the image metadata. In other words, when the detected stars are so poor quality that they resemble a different part of the celestial sphere. The false fit rate of each star detection method is displayed in Table 1. Only SExtractor has a significant false fit rate, highlighting the consequences of predicting false stars.

In order to get a more complete understanding of the relationship between deep learning metrics and astrometric fit rate, we plot precision, recall, F_1, and astrometric fit rate as functions of model confidence threshold in Figure 4. Astrometric fit rate is highly correlated to star recall. This is intuitive; detecting more stars provides astrometry.net with more information for astrometric fitting (particularly significant for narrow-field imagery). The consistency of astrometric fit rate over a wide range of predicted star quantities highlights astrometry.net's robustness to missing stars.

5.4. Exploring Star Recall

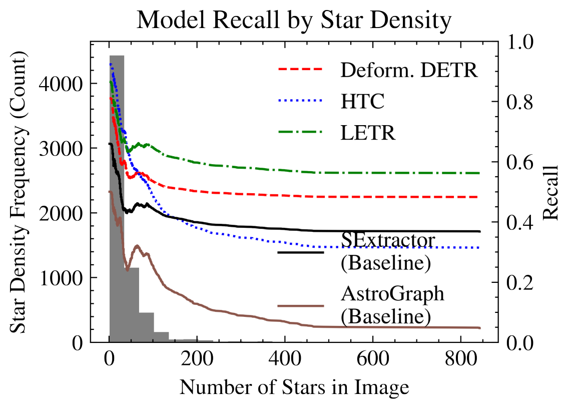


Figure 5. Natural data distribution of number of stars in image (gray), and star source extraction recall by number of stars in image (colors).

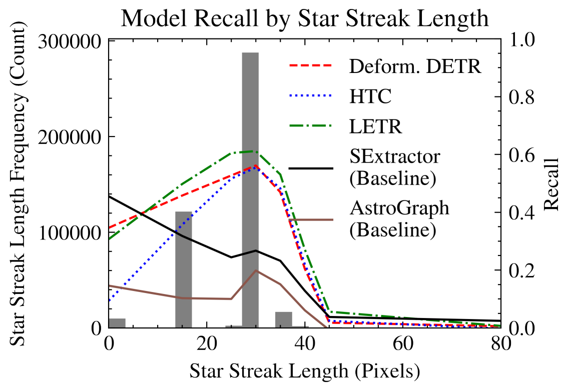


Figure 6. Natural data distribution of star streak length (gray), and star source extraction recall by star streak length (colors).

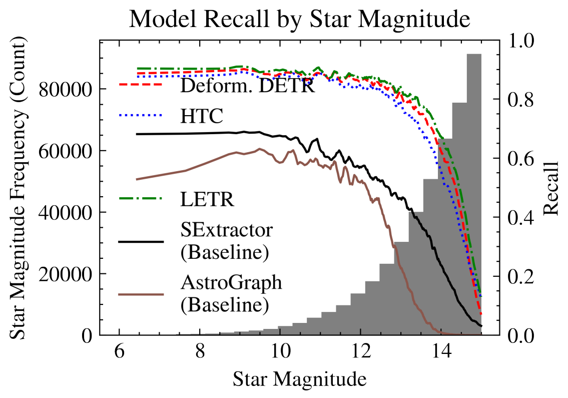


Figure 7. Natural data distribution of star magnitude (gray), and star source extraction recall by star magnitude (colors).

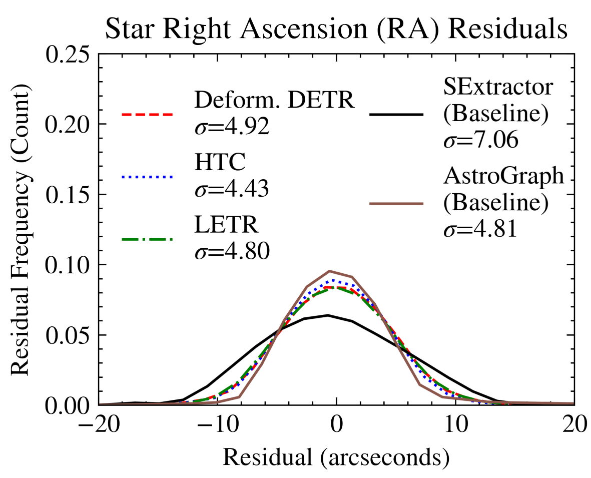
Star density has an inverse relationship with recall for every star detection method, as shown in Figure 5. The decrease in SExtractor and AstroGraph recall at higher star densities may be due to the image noise introduced by a dense star field. The decrease in model recall at higher star densities may be due to lack of representation in the training data or model architecture. We scale the number of queries in Deformable DETR and LETR to align with our data (the original Deformable DETR model uses 100 queries to detect an average of 7 objects per image [25], our models use 1000 queries to detect an average of 73 objects per image [41]). Further query scaling may increase model recall in densely populated star fields, but should be considered carefully due to the polynomial relationship between the number of object queries and the number of model parameters. HTC uses non-maximum suppression (NMS) with a threshold of 0.5, which can be problematic in dense star fields. Increasing the IoU threshold required for NMS could improve model recall in densely populated star fields, but may lead to inferior predictions in less densely populated star fields.

Star streak length has a direct relationship with model recall and an inverse relationship with SExtractor and AstroGraph recall, as shown in Figure 6. SExtractor outperforms the deep learning models for zero-pixel-long star streaks (i.e., stars in sidereal imagery). The poor performance in model recall for point source stars can be attributed to the lack of representation in the training data (only 1% of StarNet is sidereal), as well as common knowledge that deep learning models struggle with small objects [1, 43]. Although we can (and did) pad bounding boxes to facilitate detection, we could not pad segmentation masks (a point source segmentation mask only occupies 0.02% of a StarNet image), contributing to HTC's notably poor performance detecting point source stars.

Star magnitude has an inverse relationship with recall for every star detection method, as shown in Figure 7. SExtractor and AstroGraph have lower recall than the models across all star magnitudes. The drop in model recall around a visual magnitude of 14 aligns with results from previous deep learning work in astronomical source detection [16], and confirms that our ground truth star visual magnitude cutoff of 15 was a reasonable choice for this application.

5.5. Star Localization Residuals

Figure 8 displays predicted star residuals in world space for the top-performing deep learning models. Right Ascension (RA) and Declination (Dec) are coordinates used to specify the position of an object in the sky, similar to how latitude and longitude are coordinates used to specify the



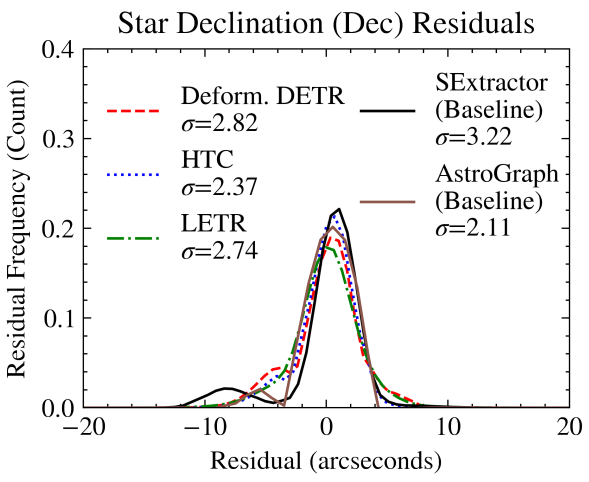


Figure 8. Star centroid residual distributions in world space. Standard deviations are included in the legend.

position of an object on the Earth's surface. RA is the angular distance between the object and the vernal equinox, while Dec is the angular distance between the object and the celestial equator. Both can be expressed in units of arcseconds, where one arcsecond is approximately equal to 1/3 of a pixel in StarNet imagery.

For the StarNet dataset, RA generally corresponds to the axis along the star streak while Dec corresponds to the axis perpendicular to the star streak. The Dec residuals are tighter than the RA residuals for every star detection method, illustrating that localizing a star center "along the streak" is more difficult than localizing a star center perpendicular to the streak. AstroGraph has a tighter residual spread than SExtractor, emphasizing the distinction between a baseline method optimized to detect point source stars (SExtractor) and a baseline method optimized to detect streak-like stars (AstroGraph). All models have similar residual spreads to AstroGraph.

5.6. Computational Complexity

Table 2. Model architecture computational complexity details and training times (in GPU hours).

Model	Params	FLOPs	GPU Hours
Deform. DETR	120.5M	1486.2G	975
HTC	76.9M	248.1G	80
LETR	177.6M	2737.25G	1460

The computational complexity of each model is detailed in Table 2. The large number of queries in Deformable DETR and LETR lead to a significant increase in the number of FLOPs, contributing to longer training times. Inference time is negligible, as rate-track telescopes take $n < 1$ images per second.

The complexity of the classical star detection methods SExtractor and AstroGraph is $O(n)$ due to their image filtering operations. In contrast, the trained star detection models make predictions in $O(1)$, enabling stars to be detected in real-time.

The entire astrometric processing pipeline still has $O(n)$ complexity, as the astrometric engine takes $O(n)$ time to match predicted stars to real-world stars. The adoption of a deep learning approach for the astrometric engine has been impeded by the significant architecture required for inference [40], but should be explored in future work to render the entire astrometric image processing pipeline $O(1)$.

6. Conclusion

We have explored the efficacy of learned approaches to star streak extraction in narrow field of view rate-track astronomical imagery. Our methods establish a new state of the art in star source extraction and bridge the last gap in the learned astrometric processing pipeline from raw telescope imagery to usable satellite tracking and identification information. By introducing a benchmark dataset and establishing quantitative performance across several task formulations we place the study of learned star source extraction on firmer footing. Future work may build on this foundation to increase recall performance in dense star fields, increase localization precision to sub-pixel levels, and may also explore end-to-end learned astrometric processing.

7. Acknowledgements

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government.

References

[1] Mohamed Abdou and Abdelkarim Erradi. Crowd counting: A survey of machine learning approaches. 2020. 7

[2] Defense Intelligence Agency. Challenges to Security in Space. Technical report, Mar. 2022. 1

[3] Jeff Aristoff, Neil Dhingra, Alex Ferris, Austin Hariri, Joshua Horwood, Ari Larson, Tyler Lyons, Jeff Shaddix, Navraj Singh, and Kevin Wilson. Non-traditional data collection and exploitation for improved GEO SSA via a global network of heterogeneous sensors. In *Proceedings of AMOS Tech Conference*, 2018. 1

[4] E. Bertin and S. Arnouts. SExtractor: Software for source extraction. *Astronomy and Astrophysics Supplement Series*, 1996. 4

[5] Colin Burke et al. Deblending and classifying astronomical sources with mask r-cnn deep learning. 2019. 2

[6] Nicolas Carion et al. End-to-end object detection with transformers. *ArXiv*, 2020. 2

[7] Kai Chen et al. Hybrid task cascade for instance segmentation. *CVPR*, 2019. 3

[8] Kai Chen et al. Mmdetection: Openmmlab detection toolbox and benchmark. *arXiv*, 2019. 4

[9] Bowen Cheng et al. Masked-attention mask transformer for universal image segmentation. *CVPR*, 2022. 3

[10] Xili Dai et al. Fully Convolutional Line Parsing, 2021. 3

[11] Maria Drozdova et al. A study of Neural networks point source extraction on simulated Fermi/LAT Telescope images. *Astronomische Nachrichten*, 2020. 2

[12] Yuxin Fang et al. Instances as queries. *ICCV*, 2021. 3

[13] Garrett Fitzgerald et al. Toward deep-space object detection in persistent wide field of view camera arrays. *AMOS*, 2021. 2

[14] Garrett Fitzgerald, Ruixu Liu, and Vijayan Asari. Geosynchronous satellite detection and tracking with wfov camera arrays using spatiotemporal neural networks (geo-spann). *SPIE*, 2022. 3

[15] Justin Fletcher et al. The dynamic optical telescope system: Collaborative autonomous sensing for space domain awareness. *Journal of Defense Research & Engineering*, 2021. 1

[16] Justin Fletcher, Ian McQuaid, and Peter Thomas. Feature-Based Satellite Detection using Convolutional Neural Networks. 2019. 1, 2, 3, 4, 5, 7

[17] J. Zachary Gazak et al. Exploiting spatial information in raw spectroscopic imagery using convolutional neural networks. *AMOS*, 2020. 3

[18] J. Zachary Gazak et al. Spectranet: Learned recognition of artificial satellites from high contrast spectroscopic imagery. *WACV*, 2022. 3

[19] Zheng Ge et al. Yolox: Exceeding yolo series in 2021. *arXiv*, 2021. 2

[20] R. Gonzalez, R. Munoz, and C Hernandez. Galaxy detection and identification using deep learning and data augmentation. 2018. 2

[21] Kaiming He et al. Mask r-cnn. *IEEE*, 2018. 3

[22] Peng Jia, Qiang Liu, and Yongyang Sun. Detection and Classification of Astronomical Targets with Deep Neural Networks in Wide-field Small Aperture Telescopes. *The Astronomical Journal*, 2020. 2

[23] Dustin Lang et al. Astrometry.net: Blind astrometric calibration of arbitrary astronomical images. *The Astronomical Journal*, 2010. 2, 5

[24] Martin Levesque. Detection of artificial satellites in images acquired in track rate mode. *AMOS*, 2011. 2, 5

[25] Tsung-Yi Lin et al. Microsoft coco: Common objects in context. *arXiv*, 2014. 7

[26] Tsung-Yi Lin et al. Focal loss for dense object detection. *IEEE*, 2017. 2

[27] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. *GitHub repository*, 2016. 4

[28] Robert Massey, Sara Lucatello, and Piero Benvenuti. The challenge of satellite megaconstellations. *Nature Astronomy*, 4(11):1022–1023, Nov. 2020. 1

[29] Guy Nir, Barak Zackay, and Eran O. Ofek. Optimal and Efficient Streak Detection in Astronomical Images. *The Astronomical Journal*, 2018. 2

[30] Klaus Okkelberg et al. Self-supervised auxiliary task learning for estimating satellite orientation. *AMOS*, 2021. 3

[31] J. Olson, S. Butow, E. Felt, and T. Cooley. State of the Space Industrial Base. Technical report, Aug. 2022. 1

[32] Matthew Phelps, J. Zachary Gazak, Thomas Swindle, Justin Fletcher, and Ian McQuaid. Inferring Space Object Orientation with Spectroscopy and Convolutional Networks. *AMOS (Sept. 2021)*, 2021. 3

[33] Carmelo Pino et al. Semantic segmentation of radio-astronomical images. 2021. 2

[34] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018. 2

[35] Shaoqing Ren et al. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 2015. 2

[36] David Rijlaarsdam et al. A Survey of Lost-in-Space Star Identification Algorithms Since 2009. *Sensors*, 2020. 2

[37] Paul Sydney and Charles Wetterer. Improvements in space surveillance processing for wide field of view optical sensors. 2014. 2, 4, 5

[38] Luis Varela et al. Streak detection in wide field of view images using Convolutional Neural Networks (CNNs). 2019. 2

[39] Douglas Woodward et al. Pixelwise image segmentation with convolutional neural networks for detection of resident space objects. 2021. 2

[40] Likai Xu, Jie Jiang, and Lei Liu. RPNet: A Representation Learning-Based Star Identification Algorithm. *IEEE Access*, 2019. 8

[41] Yifan Xu et al. Line Segment Detection Using Transformers without Edges, 2021. 3, 6, 7

[42] Danna Xue et al. Dim small target detection based on convolutional neural network in star image. *Multimedia Tools and Applications*, 2020. 2

[43] Xizhou Zhu et al. Deformable detr: Deformable transformers for end-to-end object detection. *ICLR*, 2021. 2, 3, 7