

# A High-Stability and Resource-Efficient Centroid Extraction Method for Star Maps

Qi Zhang\*, Shugang Liu

School of Physics and Electronic Science, Hunan University of Science and Technology, Xiangtan, Hunan, 411201, China

## ABSTRACT

To address the challenges of inadequate stability and excessive resource consumption in stellar centroid extraction for spaceborne star trackers, this paper proposes a single-scan centroid extraction method. The core innovation lies in the integration of an improved Run-Length Encoding (RLE) method with a Dynamic Union-Find (DUF) mechanism, which enables the concurrent execution of connected component labeling and equivalence merging during a single raster scan, significantly minimizing memory access frequency and processing latency. Optimized specifically for stellar imaging characteristics, the method maintains sub-pixel-level accuracy while adapting to irregular star morphologies and complex noise distributions. Experimental results demonstrate that for a 2048×2048 star image, the method achieves a centroid output latency of approximately 10  $\mu$ s and a per-frame processing time of 20.97 ms. The standard deviation of centroid error is  $\leq 0.1$  pixels, and the false detection rate is below 1.5% under high-noise conditions. This work provides a highly resource-efficient FPGA solution for low-latency attitude determination in aerospace applications.

## KEYWORDS

Star sensor; Centroid extraction; Connected component labeling; Dynamic union-find; FPGA Implementation

## 1 Introduction

Star trackers, serving as high-precision autonomous attitude measurement instruments, are indispensable in modern aerospace systems, including satellites, space stations, and deep-space probes<sup>[1-3]</sup>. By identifying and matching star patterns within captured images without relying on prior attitude knowledge, they achieve superior attitude determination accuracy, often at the arcsecond level, surpassing traditional sensors like sun sensors and magnetometers<sup>[4-6]</sup>. The operational workflow of a star tracker typically consists of three key stages: star centroid extraction, star pattern recognition, and attitude determination. Among these, the performance of the centroid extraction is critical, as its processing speed and accuracy directly determine the system's attitude update rate and measurement precision. This requirement becomes even more stringent in 'lost-in-space' scenarios, where full-sky scanning imposes rigorous demands on real-time response and robustness against complex background noise<sup>[7-8]</sup>.

The trend toward high-resolution image sensors in modern star trackers, while improving measurement fidelity, exponentially increases data volume, pushing conventional centroid extraction algorithms—such as wavelet analysis<sup>[9-10]</sup>, iterative optimization<sup>[11]</sup>, frame differencing<sup>[12]</sup>, and filtering-based approaches<sup>[13]</sup>—to their performance limits. Although these methods exhibit effectiveness in noise suppression, they often involve high computational complexity and substantial resource overhead (e.g., extensive on-chip memory), failing to meet the dual constraints of low processing latency and low power consumption in spaceborne applications<sup>[14-15]</sup>.

Field-Programmable Gate Arrays (FPGAs) offer a promising hardware platform for accelerating centroid extraction, thanks to their inherent parallel computing and pipeline processing capabilities<sup>[16-17]</sup>. Nevertheless, existing FPGA-based implementations suffer from notable shortcomings: The Gaussian fitting method proposed by Jiang<sup>[18]</sup>, despite its high accuracy, consumes significant on-chip DSP resources and introduces considerable processing latency; The approach developed by Fan and Zhang<sup>[19]</sup>, while robust against single-pixel noise, is inadequate in scenarios involving multi-point interference; The Connected Component Labeling (CCL) algorithms proposed by Zhao<sup>[20]</sup> and Flatow et al<sup>[21]</sup>, require multiple image traversals, leading to high memory demands, prolonged processing time, and limited adaptability to irregular star shapes and complex backgrounds.

To overcome these limitations, this paper proposes a low-latency, resource-efficient stellar centroid extraction method optimized for FPGA implementation. The main contributions of this work are as follows:

(1) We introduce a single-scan connected component labeling approach that integrates an enhanced Run-Length Encoding (RLE) technique with a Dynamic Union-Find (DUF) structure. This design enables simultaneous label assignment and equivalence fusion within a single raster scan, eliminating the latency and memory bottlenecks associated with multi-pass methods.

(2) We propose a centroid calculation trigger mechanism based on label inheritance. This innovation enables real-time

---

\* **Corresponding Author:** Qi Zhang, qz593056@gmail.com

centroid output upon detection of a connected region's completion during the scan, significantly reducing the need for processor intervention in pixel-level operations and lowering the overall system's computational burden.

The proposed method has been successfully implemented on an FPGA platform. By leveraging hardware parallelism and pipelined processing, it achieves high-throughput processing of high-resolution star images, fully meeting the operational requirements of spaceborne star trackers.

## 2 Method Design

The proposed centroid extraction method is designed for high-speed, low-latency FPGA implementation. Its core objective is to process the input pixel stream in a single pass, performing connected component labeling, equivalence fusion, and centroid data accumulation. The method architecture comprises four modules: a filtering module, a labeling and equivalence fusion module, a data storage and merging module, and a real-time centroid output module. These modules operate synchronously on the pixel stream, directly extracting centroid coordinates and associated intermediate variables (e.g., pixel count, grayscale sum) from the raw sensor data. These modules operate in a pipelined manner on the pixel stream, enabling the direct extraction of centroid coordinates from the raw sensor data in a single pass.

### 2.1 Filtering Module

The initial processing stage aims to suppress background noise, originating from cosmic stray light, sensor dark current, and readout noise, which can cause false star detections. Leveraging the characteristic that the intensities of star pixels are significantly higher than those of the background, a global thresholding method is employed for preliminary image segmentation. The global thresholding operation is defined as Equation (1):

$$p(x,y) = \begin{cases} f(x,y), & \text{if } f(x,y) \geq T \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Where  $f(x,y)$  is the grayscale value of the pixel at coordinates  $(x,y)$  in the original image,  $p(x,y)$  is the grayscale value after thresholding, and  $T$  is a pre-defined brightness threshold (determined via offline statistical analysis of dark-field images to avoid manual parameter tuning).

For scenarios with a uniform background (e.g., low-stray-light deep space), this thresholding method effectively removes low-brightness background noise. However, it cannot eliminate isolated high-brightness noise (e.g., cosmic ray hits). As shown in Figure 1 and Figure 2, after applying thresholding to a noisy image, although most noise and background regions are suppressed, some isolated bright pixels remain. These pixels may be misidentified as small stars in subsequent processing stages.

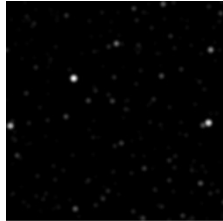


Figure 1 Noisy star image before filtering



Figure 2 Star image after threshold filtering

Additional filter techniques (e.g., Gaussian filtering, sliding-window averaging) could suppress such residual noise. However, these methods would drastically increase computational complexity and resource utilization (e.g., extra multipliers and line buffers), making them impractical for resource-constrained FPGAs. Therefore, this study does not implement additional filtering for horizontal and vertical multi-point noise at this stage. Instead, such noise is addressed in the subsequent centroid output stage: by analyzing the pixel count distribution and spatial characteristics of the stellar connected region, the system determines whether the region conforms to realistic stellar imaging features (e.g., pixel count between 4 and 30, derived from the star tracker's optical spot size and sensor pixel pitch), thereby achieving further noise filtering. This strategy strikes a balance between noise suppression accuracy and FPGA resource efficiency, thereby leveraging the advantages of simplicity, high speed, and low resource consumption.

### 2.2 Labeling and Equivalence Fusion

For threshold-filtered pixel data, this paper proposes an enhanced single-scan connected component labeling method inspired by the RLE technique. In stellar images, a star typically appears as a bright region composed of multiple adjacent pixels (exhibiting a Gaussian-like intensity distribution). The method scans the image row by row, extracting contiguous bright pixel segments (runs) in each row. Suppose the column coordinate difference between adjacent runs in consecutive rows is within one pixel (i.e., satisfying the 4-connectivity criterion). In that case, these runs are potentially connected, and cross-row connectivity is further evaluated. A DUF data structure is employed to manage run labels and

maintain equivalence relations between different runs. It supports two core operations:

**Find:** Locates the root label of a given run. Enabling the identification of the primary label representing the connected component.

**Union:** Merges two runs into the same connected component, establishing an equivalence relationship between their labels.

This design enables efficient merging of runs and unified labeling, ensuring accurate region segmentation for subsequent centroid calculation and feature extraction.

|   | 1 | 2         | 3         | 4 | 5         | 6         | 7         | y |
|---|---|-----------|-----------|---|-----------|-----------|-----------|---|
| 1 |   | <i>cl</i> | <i>cp</i> |   | <i>pl</i> | <i>pm</i> | <i>pr</i> |   |
| 2 |   |           |           |   | <i>cl</i> | <i>cp</i> |           |   |
| y |   |           |           |   |           |           |           |   |

Figure 3 Label Assignment Rules for Connected Component Labeling

After filtering, pixels are classified as either valid (foreground/star pixels) or invalid (background/noise pixels). In star images, a star typically consists of valid pixels spanning multiple rows, and its pixel segments may connect vertically, diagonally up-left, or diagonally up-right. The system assigns each valid pixel a positive integer label (referred to as 'valid label'). Invalid pixels are assigned the label 0xFF (avoiding conflicts with valid labels 0-254). As shown in Figure 3, label assignment follows a row-by-row scanning principle and depends on the pixel's position:

**First-row pixels:** Dependent on the left neighbor (*cl*). If *cl* is an invalid, a new unique label (from a pre-buffered pool) is assigned to the current pixel; otherwise, the current pixel inherits *cl*'s label to maintain continuity.

**Non-first-row pixels:** Determined by three neighboring pixels from the previous row: the upper-left pixel *pl*, the directly-above pixel *pm*, and the upper-right pixel *pr*. This 3-neighbor window strikes a balance between connectivity coverage and memory efficiency. If any upper neighbor pixels are valid, the leftmost valid label among them is assigned to the current pixel, establishing an equivalence relationship with the neighboring labels. If all upper neighbors are invalid, the first-row rule applies (dependent on the left neighbor's *cl*).

To identify cross-row star pixel clusters, DUF-based equivalence fusion records relationships between star pixel segments (e.g., merging separate branches of the same star). When a current pixel (*cp*) connects to *pl*, *pm*, or *pr*, it is deemed part of the same stellar region, triggering label equivalence establishment and merging (Figure 4 and Figure 5).

|   | 1 | 2 | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | x |
|---|---|---|----|----|----|----|----|----|----|----|----|----|---|
| 1 |   | 1 |    | 2  | 2  | 2  |    |    | 3  |    |    |    |   |
| 2 |   | 1 | p1 | p2 | p3 | p4 | p5 | p6 |    |    |    |    |   |
| 3 | 1 | 1 | 1  |    |    |    |    |    |    | 4  | 4  |    |   |
| y |   |   |    |    |    |    |    |    | p7 |    |    |    |   |

Figure 4 Schematic of Label Equivalence Merging

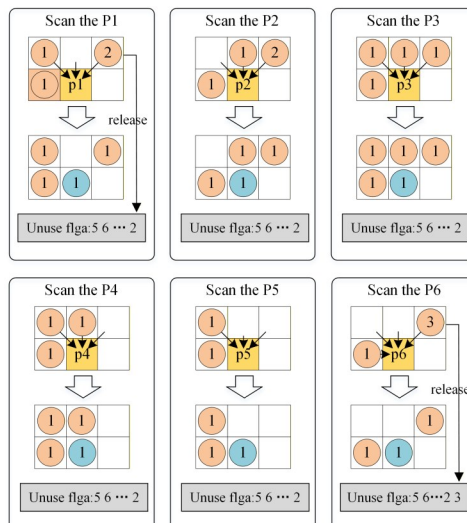


Figure 5 Steps for Labeling and Equivalence Fusion Management

As shown in Figure 5, when pixel p1 in Figure 4 is input, it connects to both p1 (label 1) and pr(label 2), causing the two segments (with labels 1 and 2) to merge. Label 2's statistics data (including pixel count and grayscale sum) are integrated into the statistical data of label 1, and the data associated with label 2 are cleared to free up memory resources. For subsequent pixel p2, which is connected to labels 1 and 2, no re-fusion occurs due to the pre-established equivalence relationship between the two labels; only pr's label updates to 1, reducing redundant operations. Special attention is required for pixel p6: It connects only to pr (label 3), with no valid pixels in pl or pm. However, according to the non-first-row rule, this situation must consider the left neighbor pixel cl (label 1). Since p6 connects to both label 3 (above) and label 1 (left), a cross-row fusion occurs between segment 3 and segment 1. When inputting pixel p7, its cl is invalid, and it connects only to pr (label 4). Hence, the current pixel is assigned to label 4, initiating a new connected component.

### 2.3 Data Storage and Merging

For each valid pixel input, the system performs pixel labeling and connected component equivalence fusion. Simultaneously, it calculates and merges the statistical data of star pixels in real-time to support subsequent centroid calculation and target identification. This study adopts a grayscale-weighted centroid method for precise star localization, as given by Equation (2):

$$X = \frac{\sum_{i=1}^n \sum_{j=1}^m p(x_i y_i) * x_i}{\sum_{i=1}^n \sum_{j=1}^m p(x_i y_i)} \quad Y = \frac{\sum_{i=1}^n \sum_{j=1}^m p(x_i y_i) * y_i}{\sum_{i=1}^n \sum_{j=1}^m p(x_i y_i)} \quad (2)$$

Where X and Y denote the grayscale-weighted centroid coordinates, and p(x, y) is the grayscale value at coordinates (x, y). For hardware-friendly computation, four accumulators are maintained for each detected connected component:

- Sum\_px: The sum of  $x * p(x, y)$  over the connected region (used to calculate the x-coordinate of the centroid);
- Sum\_py: The sum of  $y * p(x, y)$  over the connected region (used to calculate the y-coordinate of the centroid);
- Sum\_p: The sum of p(x, y) over the connected region (serves as the normalization denominator for centroid calculation);
- P\_count: The total number of pixels in the connected region (used to distinguish real stars from noise).

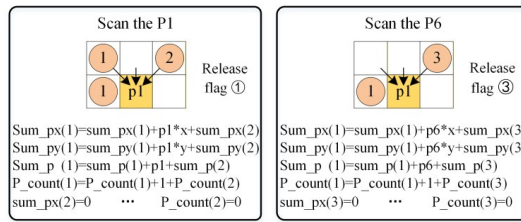


Figure 6 Schematic of Data Storage and Merging for Connected Components

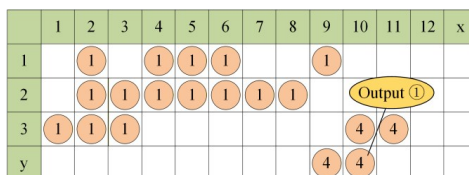
As shown in Figure 6, upon receiving new pixel data, the system updates the corresponding accumulators based on the pixel's label. When multiple labels are found to belong to the same connected component via the DUF, their accumulator data are merged. For example, when pixel p1 (associated with label 2) is merged into the connected component labeled 1, the accumulators of label 1 are updated as follows:

- Sumpx(1)=Sumpx(1)+p1\*x+Sumpx(2),
- Sumpy(1)=Sumpy(1)+p1\*y+Sumpy(2),
- Sum\_p(1)=Sum\_p(1)+p1+Sum\_p(2),
- Pcount(1)=Pcount(1)+1+Pcount(2).

After merging, the accumulator data of label 2 (e.g., Sumpx(2), Pcount(2)) are cleared to zero, releasing memory resources for reuse.

### 2.4 Centroid Output

To improve image processing efficiency and minimize processor intervention in pixel-level operations, a centroid calculation trigger mechanism based on label inheritance is proposed. The core idea is: When scanning the pixels of the current row, if the current valid pixel does not inherit the last valid label from the previous row that is adjacent to the current pixel's position, it indicates that the connected region associated with that label has ended (no new pixels will be added to it)-this triggers immediate centroid calculation and output for that region.



Once a label is no longer inherited, the system retrieves its corresponding accumulator data and computes the centroid using Equation (2). To prevent false triggers, centroid calculation and label release are restricted to the second row onward and are only permitted at mid-row or end-of-row positions. As illustrated in Figure7, when the pixel at (4,10) is input, it connects to the upper pixel pm (label 4) but fails to inherit the previous label 1. Thus, the connected region associated with label 1 is deemed complete, and its centroid is subsequently computed and output.

The resulting centroid coordinates and related data are stored in a dedicated on-chip memory (Block RAM). As configured by the system, an external processor (e.g., the ARM core in a Zynq System-on-Chip SoC) can retrieve these results in batches either after a specified number of centroids are accumulated or upon completion of full-image processing. This batch reading mechanism reduces bus communication frequency, thereby improving both bus efficiency and processor utilization.

### 3 Validation Analysis

#### 3.1 Experimental Platform and Dataset

To comprehensively evaluate the effectiveness, robustness, performance, and cross-platform adaptability of the proposed method for star map centroid extraction and connected component analysis, a multi-dimensional verification framework was established. The experiments included functional-level simulations, FPGA hardware platforms, comparative analysis with state-of-the-art methods, utilizing input data consisting of authentic images captured by spaceborne star trackers and synthetically generated images with controllable noise interference.

##### (1) Hardware Platforms

As illustrated in Figure8, two FPGA platforms were employed to evaluate the algorithm's performance under different resource constraints:

Xilinx Zynq-7020 (xc7z020clg400-2) embedded SoC: This platform supports high-speed image data transfer via an AXI DMA interface and was used to assess real-world operational efficiency in embedded scenarios.

Xilinx Kintex-7 (xc7k325tffg900-2): Equipped with 203800 Slice LUTs and 840 DSP slices, this platform provides abundant hardware resources for stress testing at 150 MHz target clock frequency. It is used to validate timing stability and throughput limits of the method during high-volume data processing.

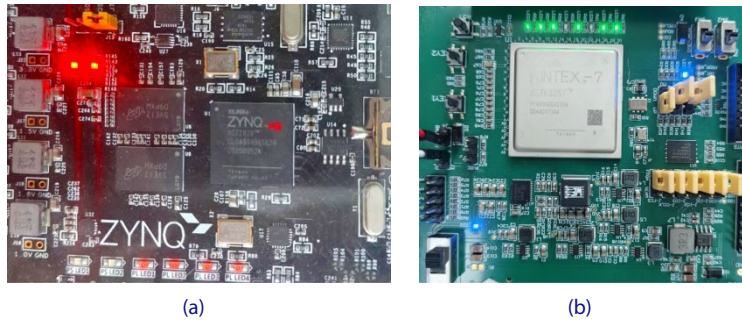


Figure 8 FPGA Hardware Implementation Platform. (a) Xilinx Zynq-7020. (b)Xilinx Kintex-k7

##### (2) Dataset Construction

The experimental dataset comprised 50 image sequences with resolutions ranging from 1024×1024 to 2048×2048 pixels, including:

- Star images densely distributed star targets, including overlapping and blurred star spots;
- High-background-noise images simulating interference from space observation environments;
- Distorted star images emulating the effects of non-ideal optical systems;
- Images overlaid with stray light and salt-and-pepper noise.

#### 3.2 Functional Correctness and Robustness Analysis

Figure9 presents typical representative images and their corresponding processing results, encompassing scenarios with stars of varying brightness levels, densely distributed star spots, and complex background noise, key challenges in practical stellar imaging.

Figs. 9(a) and 9(b) demonstrate recognition results for irregular star spots at image edges of the image, which are affected by optical distortion. The proposed method accurately labels these asymmetric regions and computes centroids with sub-pixel precision, demonstrating strong robustness to optical distortion and image blur.

Figure9(c) and 9(d) present extraction results under salt-and-pepper noise conditions with a density of 0.5%. The method successfully distinguishes valid stars from noise based on a pixel count threshold of 4 to 30 pixels, achieving 100% detection accuracy without false positives. These results validate the algorithm's excellent noise filtering capability.

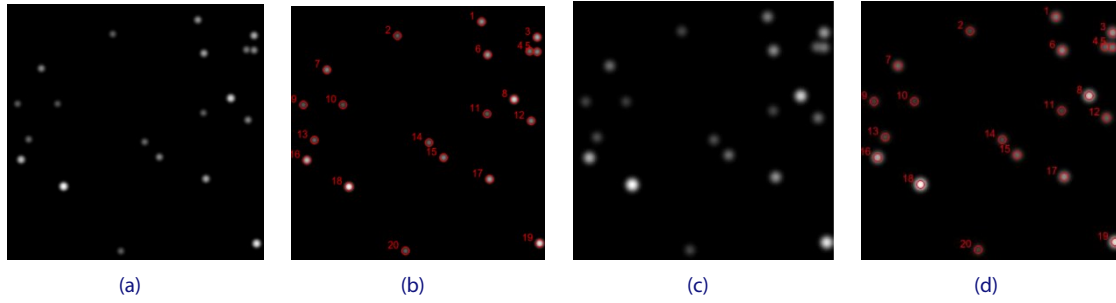


Figure 9 Star map recognition and extraction Results. (a)Original star map of irregular edge star spots. (b)Extraction result of (a). (c)Star map under Salt-and-Pepper noise. (d)Extraction result of (c)

### 3.3 Functional Correctness and Robustness Analysis

#### 3.3.1 Centroid Extraction Accuracy

To quantitatively evaluate the precision of the proposed method, a simulated dataset was used with an ideal Gaussian star spot model with known centroid coordinates as ground truth. The pipeline employed a parameterized Point Spread Function (PSF) model with varied eccentricity (0.1 to 0.5), orientation, and blur size to test the algorithm's adaptability. A Poisson-Gaussian noise model simulated realistic noise conditions. The analysis involved a test suite targeting performance attributes such as precision under varied SNR, PSF robustness, dynamic range handling, and crowded field resolution. Each image frame contained 100 stars distributed randomly in brightness (100–1000 ADU).

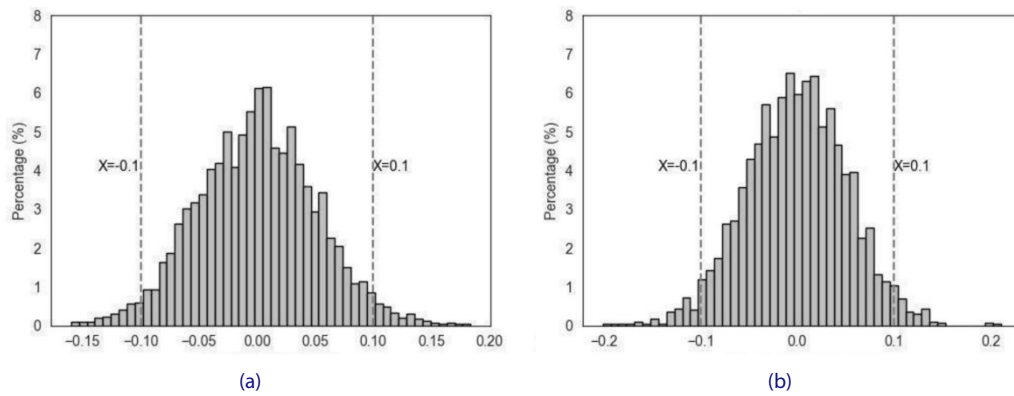


Figure 10 Distribution of centroid extraction deviations across multiple frames. (a) Row-direction deviation. (b) Column-direction deviation

The pixel-level discrepancies in both the row (x) and column (y) directions were analyzed, with the results shown in Figure 10(a) and 10(b). Statistical analysis revealed that:

- Mean deviations were 0.039 pixels (x-direction) and 0.041 pixels (y-direction);
- Maximum error: <0.2 pixels in both directions;
- Over 95% of deviations within  $\pm 0.1$  pixel.

The deviation distribution exhibited an approximately symmetrical Gaussian profile, indicating that the observed errors are predominantly caused by random noise rather than systematic bias or skewness. These results confirm that the proposed method achieves sub-pixel localization precision, thereby meeting the accuracy requirements for practical astronomical applications.

#### 3.3.2 Noise Robustness Testing

To systematically evaluate the algorithm's robustness against typical noise perturbations in stellar imaging, two sets of tests were designed to evaluate its performance under noisy data conditions. The parameter ranges of the tests cover mild to severe noise levels, and the detailed test configurations and evaluation metrics are outlined below.

(1) Stray light simulation: Gaussian noise with a mean of 0 and a standard deviation ( $\sigma$ ) ranging from 0 to 50 ADU (equivalent to a potential centroid error of 0-0.06 pixels) was added to the image background. This noise level corresponded to a potential centroid error of 0–0.06 pixels, simulating the low-frequency background interference caused by stray light in optical systems.

(2) Salt-and-pepper noise simulation: Impulse noise with density 0.1-1% was injected to simulate cosmic-ray interference. This density range also corresponds to a potential centroid error of 0-0.06 pixels, which represents the high-frequency, sparse noise typical of space-based or low-light imaging environments.

For each noise condition (i. e., each combination of noise type and intensity), 50 sequential image frames were

processed to ensure the statistical reliability of the results. Three key failure criteria were defined to evaluate the algorithm's performance:

The centroid deviation exceeded  $\pm 0.1$  pixels (from ground truth).

A false target was detected (i.e., Noise pixels were misclassified as valid star targets).

A true star was missed (i.e., actual star targets were missed, resulting in a frame detection rate of less than 95%).

The 'false detection rate' was defined as the ratio of incorrect detections to the total number of stars in the test frames, and its variation with noise intensity is presented in Figure 11.

Statistical analysis of the test results revealed three critical trends in the algorithm's noise resilience:

Under stray light noise conditions, as the noise  $\sigma$  increased, the false detection rate increased gradually. At the maximum of 50 ADU (severe stray light interference), the false detection rate reached only 2.6% (x-direction) and 3.4% (y-direction)—well below the threshold that would compromise system performance.

Under salt-and-pepper noise conditions, the false detection rate increased more slowly with noise density. Even at a noise density of 1% (equivalent to ADU), the false detection rate remains below 2.5%, demonstrating the algorithm's stronger resistance to impulse noise.

In low-noise environments ( $\sigma \leq 20$  ADU), the false detection rate is  $< 1.5\%$  for both noise types.

These results highlight the algorithm's strong noise robustness, supported by effective filtering and pixel count validation, labeling Equivalence Fusion, and grayscale-weighted centroid calculation. This combination ensures effective noise suppression without sacrificing the detection rate or centroid accuracy, confirming its suitability for real-world stellar imaging.

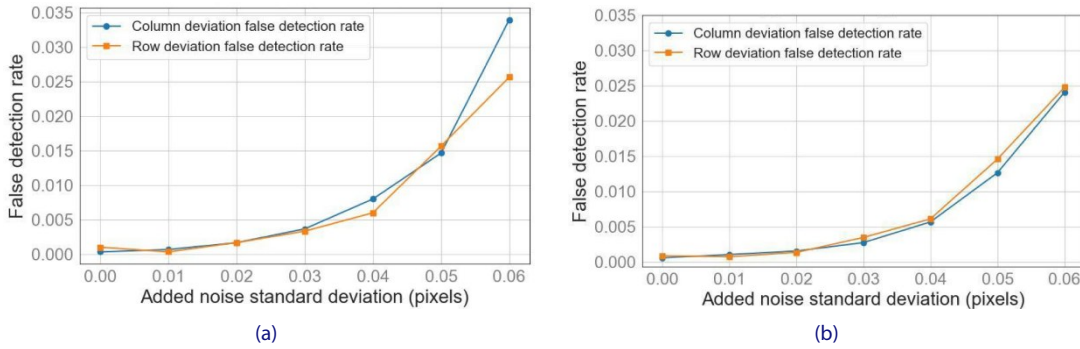


Figure 11 False Detection Rates Under Different Noise Conditions. (a) Stray light Noise (b) Salt-and-pepper noise

## 4 Conclusions

This paper presents a high-stability, resource-efficient centroid extraction method for star maps and its corresponding FPGA implementation, specifically designed to meet the performance requirements of spaceborne star trackers. The core innovation of this work lies in a single-scan connected component labeling approach that integrates an enhanced RLE technique with a dynamic DUF mechanism. This design enables the simultaneous labeling and equivalence merging during raster scanning, significantly reducing latency and memory usage.

Experimental results demonstrate that the proposed method achieves sub-pixel-level accuracy (standard deviation  $\leq 0.1$  pixels) for 2048×2048 star images, with a centroid output latency of approximately 10  $\mu$ s, and a per-frame processing time of about 20.97 ms. The false detection rate remains below 1.5% under low-noise conditions. The FPGA implementation of the method exhibits low resource utilization, making it suitable for resource-constrained embedded platforms.

The proposed method demonstrates strong robustness to irregular star shapes and complex noise backgrounds while maintaining high processing throughput. Future work will focus on enhancing the algorithm's adaptability to overlapping star fields and dynamic imaging conditions, as well as pursuing ASIC implementation for deeper space applications.

## References

- [1] Ding, J., Dai, D., Tan, W., Wang, X., & Qin, S. (2022). Implementation of a real-time star centroid extraction algorithm with high speed and superior denoising ability. *Applied Optics*, 61(11), 3115-3122.
- [2] Wei, X., Tan, W., Li, J., & Zhang, G. (2014). Exposure time optimization for highly dynamic star trackers. *Sensors*, 14(3), 4914-4931.
- [3] Yan, J., Jiang, J., & Zhang, G. (2016). Dynamic imaging model and parameter optimization for a star tracker. *Optics Express*, 24(6), 5961-5983.
- [4] Zhang, G. (2011). *Star identification*. National Defense Industry Press: Beijing, China, 109.
- [5] Liebe, C. C. (2002). Star trackers for attitude determination. *IEEE Aerospace and Electronic Systems Magazine*, 10(6), 10-16.

- [6] Agarwal, S., Hervás-Martin, E., Byrne, J., Dunne, A., Luis Espinosa-Aranda, J., & Rijlaarsdam, D. (2020). An evaluation of low-cost vision processors for efficient star identification. *Sensors*, 20(21), 6250.
- [7] Rijlaarsdam, D., Yous, H., Byrne, J., Oddenino, D., Furano, G., & Moloney, D. (2020). A survey of lost-in-space star identification algorithms since 2009. *Sensors*, 20(9), 2579.
- [8] Wang, X., Sun, C., & Sun, T. (2019). A novel two-step validation algorithm for lost-in-space star identification. *IEEE Transactions on Aerospace and Electronic Systems*, 56(3), 2272-2279.
- [9] Liu, H., Liu, C., Xie, P., Liu, S., Wang, X., Zhang, Y., ... & Zhao, Y. (2024). Stray light analysis and suppression of high-resolution camera line-of-sight variation real-time monitoring system (LoS Var RTMS). *Optics Express*, 32(14), 24184-24199.
- [10] Wang, H., & Xin, Y. (2020). Wavelet-based contourlet transform and kurtosis map for infrared small target detection in complex background. *Sensors*, 20(3), 755.
- [11] Otsu, N. (1975). A threshold selection method from gray-level histograms. *Automatica*, 11(285-296), 23-27.
- [12] Zhang, P., Zhao, Q., Liu, J., & Liu, N. (2014). A brightness-referenced star identification algorithm for aps star trackers. *Sensors*, 14(10), 18498-18514.
- [13] Liu, H. B., Yang, J. K., Wang, J. Q., Tan, J. C., & Li, X. J. (2011). Star spot location estimation using Kalman filter for star tracker. *Applied optics*, 50(12), 1735-1744.
- [14] Wei, M. S., Xing, F., & You, Z. (2018). A real-time detection and positioning method for small and weak targets using a 1D morphology-based approach in 2D images. *Light: Science & Applications*, 7(5), 18006-18006.
- [15] Bai, X., Zhang, S., Du, B., Liu, Z., Jin, T., Xue, B., & Zhou, F. (2011). Survey on dim small target detection in clutter background: wavelet, inter-frame and filter based algorithms. *Procedia Engineering*, 15, 479-483.
- [16] Marin, M., & Bang, H. (2020). Design and simulation of a high-speed star tracker for direct optical feedback control in ADCS. *Sensors*, 20(8), 2388.
- [17] Chianelli, G., Kunalakantha, P., Myhre, M., & Lee, R. S. (2023). A dual-purpose camera for attitude determination and resident space object detection on a stratospheric balloon. *Sensors*, 24(1), 71.
- [18] Jiang, H., & Fan, X. Y. (2012). Centroid locating for star image object by FPGA. *Advanced Materials Research*, 403, 1379-1383.
- [19] Fan, Q. Y., & Zhang, G. J. (2011). Spot centroiding algorithm for discrete noise image and its hardware implementation. *Guangxue Jingmi Gongcheng (Optics and Precision Engineering)*, 19(12), 2992-2998.
- [20] Zhao, F., Zhang, L., Zhang, Z. Y., & Lu, H. Z. (2011). A hardware acceleration based algorithm for real-time binary image connected-component labeling. *Journal of Electronics & Information Technology*, 33(5), 1069-1075.
- [21] Flatt, H., Blume, S., Hesselbarth, S., Schunemann, T., & Pirsch, P. (2008, July). A parallel hardware architecture for connected component labeling based on fast label merging. In *2008 International Conference on Application-Specific Systems, Architectures and Processors* (pp. 144-149). IEEE.