Citation: Bowei Xu, Cungui Yu, Jianlin Zhong, et al. Real-time feature detection for booster recovery: An adaptive harris framework integrating scharr vectorized gradients and dilation-based NMS. *Journal of Harbin Institute of Technology (New Series)*. DOI:10.11916/j.issn.1005-9113.25027

Real-Time Feature Detection for Booster Recovery: An Adaptive Harris Framework Integrating Scharr Vectorized Gradients and Dilation-Based NMS

Bowei Xu[1], Cungui Yu[1], Jianlin Zhong[1] and Qing He[2]*

(*1. School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China;*
2. Shanghai Aerospace Systems Engineering Research Institute, Shanghai 201109, China)

Abstract: In rocket booster recovery missions, robust visual perception is critical for real-time localization and landing control, but complex environments, including noise, illumination variations, and occlusion, severely degrade traditional corner detection methods, leading to a high false rate of detections and computational delays. To address these challenges, an Adaptive Robust Harris (ARH) corner detection algorithm is proposed in this study. The method integrates four innovations: 1) Scharr operator-based gradient computation for enhanced edge response; 2) An adaptive thresholding via statistical analysis of Harris responses to improve noise robustness; 3) A hybrid approach combining dilation-accelerated Non-Maximum Suppression (NMS) and sub-pixel refinement using OpenCV's goodFeaturesToTrack, reducing redundancy and achieving precise localization (average displacement: 0.0458 pixels); 4) Vectorized computation replacing explicit loops, optimizing runtime by 40%. Experimental validation demonstrates ARH's superior performance: high repeatability ($\approx$0.85) under Gaussian noise, 20% feature reduction in motion blur (vs. 21% for Harris and 82% for SIFT), and stable keypoints under rotation. Computational complexity analysis reveals that adaptive thresholding is optimized from $O(N)$ to $O(1)$, while NMS efficiency improves by 50%. These advancements position ARH as a high-precision, real-time solution for vision-critical tasks in aerospace applications, such as tracking rocket boosters and providing landing assistance.

Keywords: Adaptive Robust Harris (ARH); Scharr operator; adaptive thresholding; Non-Maximum Suppression (NMS); complex environments; rocket booster recovery

CLC number: V448, TP391 **Document code**: A **Article ID**: 1005-9113(2026)00-0000-17

0 Introduction

In missions involving rocket booster recovery[1], reliable and precise visual perception plays an indispensable role in real-time positioning, trajectory calculation, and final control of landing processes. Among visual features, corner points[2] are particularly important due to their spatial stability and repeatability, which are essential for downstream tasks such as feature matching[3], visual tracking[4], and 3D reconstruction[5]. During the high-speed descent of a rocket booster, the vision system must identify and track multiple high-feature-density regions to ensure accurate navigation[6]. However, environmental factors such as lighting variation, Gaussian noise, occlusion, lens contamination, and motion blur can severely affect the reliability and repeatability of corner detection algorithms[7].

The traditional Harris corner detection algorithm[8] is well-regarded for its simplicity, computational efficiency, and strong edge response under standard imaging conditions. However, it has notable drawbacks when applied to dynamic or degraded imagery. Specifically, Harris is sensitive to manually defined parameters, and its fixed-threshold mechanism often results in clustered corners in textured regions while failing to detect key points in

Received 2025-04-28.
Sponsored by Shanghai Aerospace Science and Technology Innovation Fund (Grant No.SAST2023-037).
* Corresponding author: Jianlin Zhong, Ph.D, Associate Professor. Email: 2748637560@qq.com.cn.

low-contrast or noisy areas. These limitations reduce the algorithm's robustness and hinder its ability to support precise localization tasks in complex aerospace environments, especially when fast decision-making and minimal delay are required.

To address the inherent limitations of traditional Harris corner detection, researchers have developed numerous enhanced approaches that demonstrate superior adaptability and performance[9]. These improvements primarily focus on six technical directions: 1) Feature enhancement and scale invariance, exemplified by SIFT's Gaussian scale-space construction[10-11], though its computational intensity limits real-time applications[12]; 2) Computational efficiency optimization demonstrated by ORB's combination of FAST detection with BRIEF descriptors[13], achieving balanced speed and accuracy for resource-constrained environments[14]; 3) Hybrid methodologies[15-18], including PCA-optimized ORB for improved feature matching[19], Canny-based Harris refinements for edge-aware localization[20], and Harris-FAST integration for enhanced robustness[17,21]; 4) Adaptive techniques employing dynamic thresholding[22] and progressive Non-Maximum Suppression (NMS)[23] to handle noisy conditions; 5) Environment-specific solutions for challenging scenarios like underwater imaging[16,24-25]; and 6) Emerging deep learning approaches that learn complex features[26], albeit with computational constraints.

Despite these advancements, the fundamental trade-offs between real-time performance, detection accuracy, and environmental adaptability remain a critical barrier for vision systems in rocket booster recovery scenarios, where rapidly evolving visual conditions demand simultaneous robustness, precision, and speed. To bridge this gap, this study introduces an Adaptive Robust Harris (ARH) corner detection algorithm, which addresses the aforementioned limitations of conventional Harris through the following four innovations: 1) Enhanced gradient estimation using the Scharr operator to achieve high edge response sensitivity in complex texture regions[18]; 2) A statistical adaptive thresholding strategy that adjusts dynamically to the distribution of corner responses, eliminating the need for manual threshold selection[22,27]; 3) Efficient non-maximum suppression via morphological dilation, significantly reducing redundant feature points and improving keypoint separation[28]; 4) Sub-pixel refinement using OpenCV's goodFeaturesToTrack, ensuring high localization precision and geometric accuracy.

This research aims to develop an advanced corner detection solution that simultaneously addresses three critical requirements for aerospace applications: enhancing detection accuracy and robustness under challenging operational conditions typical of rocket booster recovery scenarios, optimizing computational performance to satisfy stringent real-time processing constraints onboard the spacecraft, and rigorously validating the proposed ARH algorithm through comprehensive testing encompassing both synthetic benchmarks and actual flight imagery to evaluate its performance across diverse interference conditions including sensor noise, motion blur, and partial occlusion. The integrated approach seeks to bridge the gap between theoretical corner detection performance and practical implementation requirements in space systems.

1 An Improved Corner Detection Algorithm Based on Harris (ARH)

The rocket booster recovery process, as the core technology of reusable aerospace systems, directly determines the success or failure of the mission through spatiotemporal coordination of its operational sequence. As shown in Fig.1, the process begins with navigation control and speed adjustment during the booster's final flight segment. When the rocket enters the 300 m range of the launch capture tower (rocket approach phase), the attitude adjustment subsystem is required to adjust the heading angle accuracy. At this point, the dual-redundancy mechanism formed by the buffer device and capture arm enters the preparatory state. The visual perception system plays a critical role in this phase: by detecting high-feature point areas in real time, the system achieves millimeter-level positioning through feature point spatial coordinate calculations. The accuracy of this detection directly affects the timing of the capture arm deployment and the engagement of the locking mechanism. However, traditional feature detection methods are prone to feature drift under dynamic lighting conditions and partial occlusion, leading to cumulative errors in the navigation control loop. To address this, the ARH algorithm proposed in this paper introduces a multi-scale gradient response fusion mechanism and dynamic

threshold compensation strategy, significantly enhancing visual positioning robustness under complex conditions, while maintaining real-time performance. The following sections describe the specific process of the algorithm.

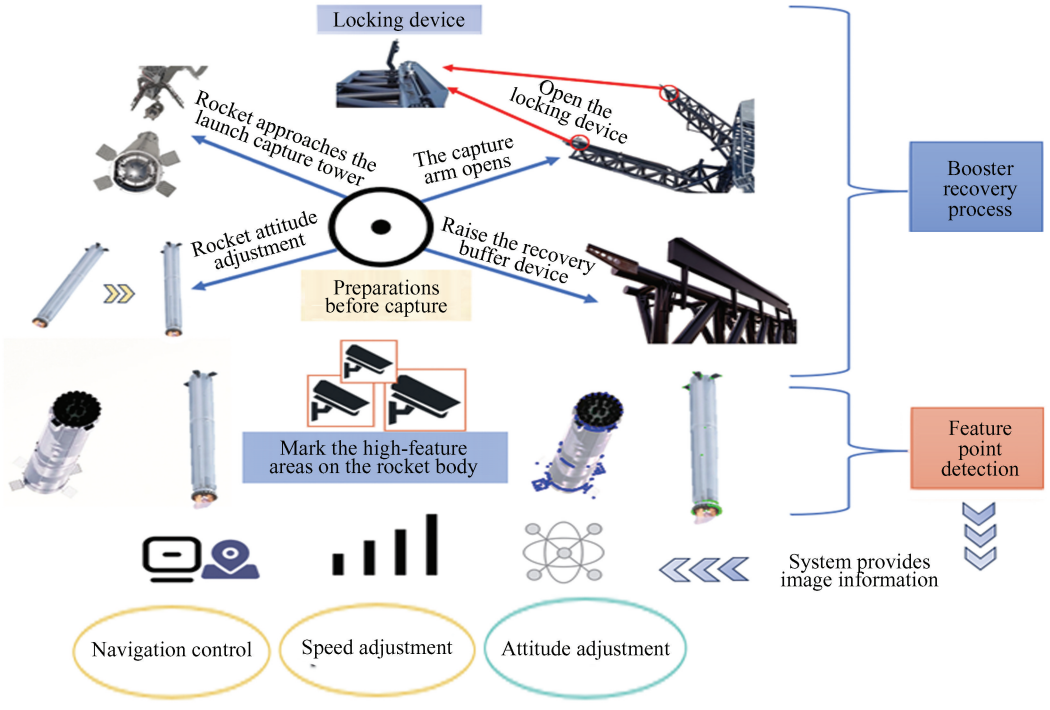


Fig. 1 Multimodal control system architecture for reusable rocket boosters: integrated navigation adjustment, feature detection, and robotic arm coordination

1.1 The Basic Principle of Harris Corner Detection

Harris corner detection is a classic algorithm based on gradient variations in grayscale images. Its fundamental concept involves evaluating local intensity changes in the image to identify corners. Typically, corners are points with significant intensity variation or intersections of edges within the image.

Assuming the grayscale value of an image is represented as $I(x,y)$, and a small displacement (u, v) is applied to the image, the change in grayscale value can be represented by the autocorrelation function as follows:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2 \tag{1}$$

here, $w(x,y)$ is a weighted window function, typically a Gaussian function, used to reduce the impact of noise. By applying a Taylor expansion to the expression and neglecting higher-order terms, the intensity variation $E(u,v)$ can be represented as:

$$E(u,v) = \begin{bmatrix} u & v \end{bmatrix} \boldsymbol{M} \begin{bmatrix} u \\ v \end{bmatrix} \tag{2}$$

here, $\boldsymbol{M}$ is the autocorrelation matrix, defined as follows:

$$\boldsymbol{M} = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{3}$$

The matrix $\boldsymbol{M}$ contains gradient information of the image in a local region, with its two eigenvalues, λ_1 and λ_2, describing the intensity variations. Specifically:

When both λ_1 and λ_2 are large, indicating significant changes in both directions, suggesting that the region is a corner.

When one eigenvalue is large, and the other is small (i.e., $\lambda_1 \gg \lambda_2$ or $\lambda_1 \ll \lambda_2$), it implies the presence of a distinct edge in the region.

When both eigenvalues are small, it indicates minimal intensity variation, suggesting that the region is flat.

1.2 Definition of the Corner Response Function

To avoid directly computing the eigenvalues, Harris proposed a simplified response function (R), which approximates the relationship between the eigenvalues using the determinant and trace of the matrix:

$$R = \det(\boldsymbol{M}) - k \cdot (\mathrm{trace}(\boldsymbol{M}))^2 \quad (4)$$

where k is an empirical constant, typically ranging from 0.04 to 0.06. $\det(\boldsymbol{M})$ represents the determinant of matrix $\boldsymbol{M}$, expressed as $\lambda_1 \lambda_2$:

$$\det(\boldsymbol{M}) = I_x^2 I_y^2 - (I_x I_y)^2 \quad (5)$$

A larger determinant value indicates more significant gradient changes in the region, suggesting it is likely a corner.

$\mathrm{trace}(\boldsymbol{M})$ represents the trace of matrix $\boldsymbol{M}$, which is the sum of the two eigenvalues, $\lambda_1 + \lambda_2$:

$$\mathrm{trace}(\boldsymbol{M}) = I_x^2 + I_y^2 \quad (6)$$

A larger trace value indicates significant intensity variation in the region, but it does not confirm the presence of a corner; it may indicate an edge.

Matrix $\boldsymbol{M}$ can be represented using the smoothed results:

$$\boldsymbol{M} = \begin{bmatrix} S_{x^2} & S_{xy} \\ S_{xy} & S_{y^2} \end{bmatrix} \quad (7)$$

and:

$$S_{x^2} = G_\sigma * I_x^2 \quad (8a)$$

$$S_{y^2} = G_\sigma * I_y^2 \quad (8b)$$

$$S_{xy} = G_\sigma * (I_x I_y) \quad (8c)$$

where I_x and I_y represent the image gradients in the x and y directions, respectively. G_σ is a Gaussian window with a standard deviation of σ. The symbol $*$ denotes the convolution operation.

1.3 Advantages of the Corner Response Function

1) Stability: Due to the use of Gaussian weighting and matrix representation, the Harris algorithm exhibits good robustness against noise.

2) High efficiency: Compared to directly computing the eigenvalues of the matrix, the use of determinant and trace approximations reduces computational complexity.

3) Robustness: Harris corner detection demonstrates strong robustness to image rotation and to certain levels of noise variation.

Using the aforementioned corner response function, the Harris algorithm can efficiently and stably detect corners in images, providing a foundation for subsequent tasks such as image matching, tracking, and recognition.

1.4 Principles of the Improved Algorithm

The imaging procedure of the improved Harris algorithm is as follows.

Input: Set of input parameters and image $I(x, y)$

Parameters: σ (Gaussian window standard deviation), k (threshold sensitivity parameter), r (neighborhood radius)

Image: Input grayscale image $I(x,y)$

Imaging Process

Step 1: Initialization

1) Gradient calculation

Compute image gradients G_x and G_y using the Scharr operator: $G_x = I * K_x, G_y = I * K_y$

2) Gradient product calculation

Compute gradient products: $I_{xx} = G_x^2, I_{yy} = G_y^2, I_{xy} = G_x G_y$

3) Gaussian smoothing

Apply a Gaussian filter to smooth gradient products to reduce noise:

$S_{xx} = I_{xx} * G(\sigma)$, $S_{yy} = I_{yy} * G(\sigma)$, $S_{xy} = I_{xy} * G(\sigma)$

Step 2: Harris response calculation

1) Compute matrix $\boldsymbol{M}$

Construct the autocorrelation matrix $\boldsymbol{M}(x,y)$:

$$\boldsymbol{M}(x,y) = \begin{bmatrix} S_{xx} & S_{xy} \\ S_{xy} & S_{yy} \end{bmatrix}$$

2) Compute Harris response

Compute Harris response $R(x,y)$:

$R(x,y) = \det(\boldsymbol{M}) - k \cdot (\mathrm{trace}(\boldsymbol{M}))^2$

Step 3: Adaptive threshold selection

1) Collect positive responses

Define set R_+ of all positive Harris responses:

$R_+ = \{R(x,y) \mid R(x,y) > 0\}$

2) Calculate mean and Std Dev

Calculate the mean and standard deviation of R_+:

$R_{\mathrm{mean}}, R_{\mathrm{std}}$

3) Set adaptive threshold

Set adaptive threshold T_{th}:

$$T_{th} = R_{mean} + k \cdot R_{std}$$

Step 4: Non-maximum suppression

Perform dilation

Use dilation to find local maxima in a neighborhood radius r

Retain the point if it is a local maximum and above the threshold T_{th}.

Step 5: Subpixel precision enhancement

Use goodFeaturesToTrack

Use OpenCV's goodFeaturesToTrack for subpixel precision refinement:

This optimizes the corner locations based on gradient and intensity info.

Output: Matched feature index pair set $\boldsymbol{M}$; Coordinates of detected corner points $\{(x_i, y_i)\}$

Explanation of steps:

1) Step 1 (Initialization):

Construct gradient maps using Scharr operators for a more accurate edge response.

2) Step 2 (Harris response calculation):

Use the autocorrelation matrix to evaluate the local change of intensity, which indicates the corner's presence.

3) Step 3 (Adaptive threshold selection):

Compute an adaptive threshold based on the statistical properties of the Harris response to adapt to different noise and contrast levels.

4) Step 4 (Non-maximum suppression):

Dilation is used to retain only the local maxima, ensuring isolated corners are detected while discarding neighboring detections.

5) Step 5 (Subpixel precision enhancement):

The goodFeaturesToTrack function helps refine detected corners, ensuring subpixel-level accuracy for subsequent applications.

1.5 Adaptive thresholding method

To achieve adaptive thresholding, we performed a statistical analysis of the positive R values:

1) Collect positive R values

Define the set S:

$$S = \{R(x,y) \mid R(x,y) > 0\} \tag{9}$$

2) Computation of statistical measures

Calculate the mean μ_R and standard deviation σ_R of set S:

$$\mu_R = \frac{1}{N}\sum_{i=1}^{N} R_i \tag{10}$$

$$\sigma_R = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (R_i - \mu_R)^2} \tag{11}$$

where N is the number of elements in set S.

3) Define the adaptive threshold T_{th}

Set the threshold T_{th} as:

$$T_{th} = \mu_R + k_t \cdot \sigma_R \tag{12}$$

where k_t is a parameter that controls the sensitivity of the threshold. A larger k_t results in a higher threshold, leading to fewer detected corners.

4) Principle of the method

Determine T_{th} based on μ_R and σ_R:

a) Adaptive to image content: T_{th} is adjusted according to the distribution of corner responses within the image.

b) Enhanced robustness: Reduces the impact of outliers and adapts to varying contrast and noise levels.

c) Improved detection performance: Achieves a balance between detecting true corners and suppressing false detections.

1.6 Non-Maximum Suppression (NMS)

During corner detection, multiple adjacent corners may be generated, which can be considered as false corners or corner clusters. To eliminate these false corners, the algorithm uses Non-Maximum Suppression (NMS) to retain only the strongest corner in the local region. Mathematically, NMS can be represented as:

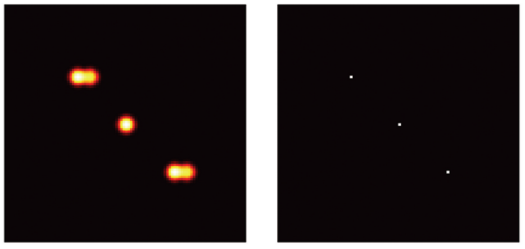
$$R'^{(i,j)} = \begin{cases} R(i,j), & \text{if } R(i,j) = \max(R(i-r:i+r, j-r:j+r)) \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where $R(i,j)$ is the Harris response value at the point (i,j); r is the suppression radius, defining the range of the local neighborhood.

Only when the current pixel $R(i,j)$ is the maximum value within its neighborhood is it retained; otherwise, it is set to zero.

Fig. 2 illustrates how NMS improves corner detection. Fig.2(a) shows the initial corner response, with multiple bright regions indicating potential corners. These regions may contain several detections of the same feature, leading to redundancy and

increased errors. Fig. 2 (b) displays the corner detection result after NMS, which retains only the strongest response in each region, eliminating redundant detections and improving precision. By reducing noise and extraneous points, NMS produces a more focused set of corners, which is crucial for subsequent image processing tasks such as feature matching and tracking.



(a) Original corner response (b) Detection result after NMS

Fig.2 Comparison of corner detection before and after NMS(a sample of a figure caption)

Traditional NMS is typically implemented by searching for the maximum value within the local neighborhood of each pixel, often involving nested double for loops with a computational complexity of $O(N \times r^2)$, where r is the neighborhood radius, and N is the number of pixels.

To optimize this process, the algorithm uses the dilation operation in OpenCV to perform NMS. The dilation operation finds the maximum value within each local neighborhood through a sliding window, and its highly optimized low-level implementation effectively reduces computational load:

Mathematical description of the dilation operation is as follows:

$$R'(i,j) = \max_{(x,y) \in N(i,j)} R(x,y) \tag{14}$$

where $N(i,j)$ represents the neighborhood centered at (i,j). The dilation operation is implemented in parallel, which significantly reduces the runtime.

By finding and retaining the local maximum at the same position in the original image, the dilation operation greatly improves the speed of NMS and ensures that only the strongest local corner responses are preserved.

1.7 Sub-pixel Precision Corner Position Correction

To further enhance the precision of corner positions, the algorithm refines corner locations by calculating the arithmetic mean of the corner points within their neighborhoods. This step allows the corner positions to achieve sub-pixel accuracy.

Given the neighborhood $N(i,j)$ of a corner, the mean position of all corners within this neighborhood can be calculated using the following formula:

$$x_{\text{avg}} = \frac{1}{|N(i,j)|} \sum_{(x,y) \in N(i,j)} x \tag{15a}$$

$$y_{\text{avg}} = \frac{1}{|N(i,j)|} \sum_{(x,y) \in N(i,j)} y \tag{15b}$$

where x_{avg} and y_{avg} are the refined corner positions. $N(i,j)$ represents all corner points within the neighborhood centered at (i,j) with a radius r, $|N(i,j)|$ is the number of corner points within the neighborhood.

In the original algorithm, this process was implemented manually. In the improved version, the OpenCV function goodFeaturesToTrack is used to achieve sub-pixel refinement of corner positions.

goodFeaturesToTrack finds corners using the minimum eigenvalue method and allows configuration of quality level and minimum distance parameters. This approach provides a more efficient way to extract and accurately locate corners.

Mathematically, the function calculates the positions of interest points based on the gradient information of the image, utilizing a highly optimized implementation.

By using goodFeaturesToTrack, not only is the accuracy of corner positions improved, but the additional computational complexity associated with manually iterating to calculate the mean is significantly reduced.

1.8 Gradient Calculation Improvement: Using the Scharr Operator

Gradient calculation (Scharr vs. Sobel Operators): Both the Sobel and Scharr operators are used to calculate image gradients to highlight edges and corners. The Scharr operator is an improved version of the Sobel operator, employing different convolution kernel parameters to enhance edge detection in more complex images. Specifically, when calculating gradients in the x and y directions, the Scharr operator reduces the attenuation of high-frequency responses seen with the Sobel operator, resulting in more accurate gradient computation.

Mathematically, the Sobel operator uses a 3×3 convolution kernel for gradient estimation:

$$\boldsymbol{G}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * \boldsymbol{I} \tag{16a}$$

$$\boldsymbol{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \boldsymbol{I} \quad (16b)$$

The Scharr operator uses optimized kernel parameters:

$$\boldsymbol{G}_x = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix} * \boldsymbol{I} \quad (17a)$$

$$\boldsymbol{G}_y = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{bmatrix} * \boldsymbol{I} \quad (17b)$$

Compared to the Sobel operator, the Scharr operator reduces directional bias in edge detection by optimizing weighting, providing better edge responses and more accurate corner localization. Therefore, in this algorithm, the Scharr operator is used to accelerate gradient computation and improve corner-detection accuracy.

1.9 Vectorized Computation Instead of Explicit Loops

When calculating the response value R, the improved algorithm uses vectorized operations instead of explicit Python loops. The advantage of this approach is that vectorized computations can more efficiently leverage modern CPUs or GPUs for parallel processing.

In adaptive threshold calculation: The original calculation involved looping through R values to compute the mean and standard deviation, which is highly time-consuming in Python. Vectorized operations utilize NumPy functions to calculate the mean μ_R and standard deviation σ_R. These functions are internally optimized using low-level C language and multithreading acceleration.

The vectorized formula is:

$$\mu_R = \frac{1}{N}\sum_{i=1}^{N} R_i, \sigma_R = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (R_i - \mu_R)^2} \quad (18)$$

This improvement effectively reduces the time complexity from $O(N)$ to a near $O(1)$ complexity through low-level implementation (using multithreaded parallel processing).

1.10 Advantages of the Improved Algorithm

1) Enhanced detection accuracy: The algorithm employs the Scharr operator for gradient computation, combined with NMS and sub-pixel refinement, resulting in improved corner detection and precise feature localization.

2) Increased resistance to interference: Adaptive thresholding enables dynamic adjustments based on image content, enhancing robustness to noise and lighting variations. This leads to greater stability and reliability in detection.

3) Improved computational efficiency: The enhanced algorithm achieves approximately 40% greater efficiency than the traditional Harris algorithm by using the Scharr operator and vectorized operations, eliminating explicit loops. NMS efficiency is further improved through dilation operations.

4) Adaptive and automated parameter adjustment: The integration of adaptive thresholding and NMS eliminates the need for manual parameter tuning, ensuring consistent performance across diverse environments, especially in real-time applications.

5) High repeatability and stability: The algorithm demonstrates high repeatability under varying noise levels and lighting conditions, making it suitable for high-precision applications such as rocket booster recovery.

These enhancements make the algorithm robust in challenging environments, making it ideal for tasks requiring high precision and real-time performance, such as visual tracking and attitude estimation during rocket booster recovery.

2 Experimental Design and Results

2.1 Experimental Dataset and Design

Experiments were conducted using simulated rocket booster recovery images and real-world images, covering conditions such as noise, lighting variations, and occlusion. The final set of test images was captured and processed at 1050 × 720 resolution, ensuring the selected images were suitable for evaluating the algorithm's performance under realistic conditions.

2.2 Experimental Evaluation Criteria

1) Algorithm detection capability assessment.

These metrics are primarily used to evaluate the feature detection capabilities of the algorithm, including the number of corner points, average displacement, and detection accuracy.

a) Number of corner points:

Metric meaning: This metric is used to evaluate the number of feature points detected by the algorithm. The number of feature points is a fundamental indicator of feature extraction capability. A higher number generally suggests stronger feature extraction ability, though it does not necessarily indicate higher

feature quality. This metric helps assess detection capability and the richness of feature points.

Representation: Typically expressed as the total number of detected feature points, N. For example, comparing the number of feature points detected by the Harris algorithm and other algorithms under different thresholds provides insight into the variability of feature point detection.

b) Average displacement:

Metric meaning: This metric measures the displacement between feature points detected under different conditions (e.g., noise, rotation, occlusion), which is used to evaluate the stability and detection accuracy of the feature points.

Mathematical expression:

$$d_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N} \| p_i - \hat{p}_i \| \tag{19}$$

where p_i represents the coordinates of the i-th feature point in the original image, $\hat{p}_i$ represents the coordinates of the corresponding feature point in the transformed image (e.g., after rotation, noise, etc.), N is the number of feature points, and $\| \cdot \|$ denotes the Euclidean distance.

Classification: This metric is used to evaluate detection accuracy and the stability of feature points in the algorithm.

Practical significance: A smaller average displacement indicates more stable feature point positions and more precise detection.

2) Repeatability and robustness assessment.

These metrics are used to evaluate the stability and consistency of the algorithm under different conditions, including repeatability and robustness.

a) Repeatability.

Metric meaning: Repeatability refers to the consistency of the algorithm in detecting feature points when processing the same image under different conditions (e.g., rotation, noise). It indicates how reliably the algorithm detects the same features despite variations.

Mathematical expression:

$$R = \frac{N_{\text{matched}}}{N_{\text{total}}} \tag{20}$$

where N_{matched} represents the number of feature points matched before and after the changes, and N_{total} represents the total number of initially detected feature points.

Classification: This metric is used to evaluate the robustness of the algorithm, particularly the consistency of feature point detection under variations such as rotation and noise.

Practical significance: Higher repeatability indicates better robustness of the algorithm to various changes. In experiments, as shown in the documented results (Section 2.4), Adaptive Harris and ORB demonstrated high repeatability, particularly approaching 1.0 under rotational variations.

b) Robustness.

Metric meaning: This metric is used to evaluate the stability and accuracy of the algorithm in detecting feature points under challenging conditions, such as noise, occlusion, and motion blur.

Evaluation method: Robustness is assessed by comparing metrics such as the number of corner points, repeatability, and computational efficiency, providing a comprehensive evaluation of the algorithm's performance under different conditions. For example, in a noisy environment, Adaptive Harris demonstrated good stability in the number of detected feature points, showing strong robustness against noise.

3) Computational efficiency assessment.

Computational efficiency refers to the evaluation of the time consumption of an algorithm when processing images, typically measured in seconds. This metric involves directly using time measurements to compare the performance of different algorithms, such as Harris, SIFT, and ORB, to determine their efficiency. It serves as a classification of time cost and execution efficiency, highlighting how quickly an algorithm can process data. In practical terms, shorter computation times are indicative of higher efficiency, which is crucial for real-time applications. For instance, ORB demonstrates the shortest computation time among these algorithms, making it particularly suitable for scenarios that require high real-time performance.

4) Comprehensive evaluation.

A comprehensive evaluation examines multiple metrics to assess the overall algorithm performance and identify its strengths and optimal use cases. Key metrics include rotation invariance, which measures the algorithm's ability to maintain feature consistency under rotation by evaluating repeatability and changes in corner points. Experiments show that both Adaptive Harris and ORB perform well across different angles. Performance under noise and occlusion is another critical metric, gauging detection capability and

feature retention in challenging conditions. This is measured by corner point count, average displacement, and repeatability in noisy or occluded environments. For example, in simulations with lens smudges or smoke, Adaptive Harris demonstrated stable detection and low displacement, indicating good adaptability in low-contrast scenes. These metrics—corner point count, displacement, repeatability, robustness, and computational efficiency—provide a thorough assessment of detection accuracy, stability, and real-time applicability, enabling a detailed comparison of algorithm strengths and weaknesses across various scenarios.

2.3 Experimental Results and Analysis

1) Corner quantity and time comparison.

A comprehensive evaluation of the Harris and SIFT feature extraction algorithms (Table 1) shows that Harris significantly outperforms SIFT in runtime, averaging about 0.407 s, making it suitable for real-time tasks. However, the traditional Harris method often detects many false corners and clusters (Fig.3), which limits its accuracy and makes it unsuitable for precise applications, such as rocket booster recovery.

Table 1 Time comparison of Harris and SIFT on five sets of identical images

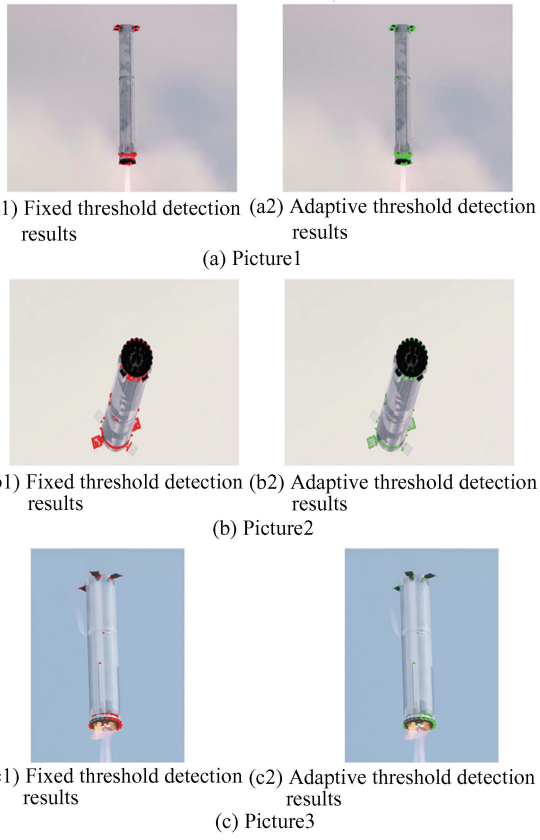
Method	Number	Runtime on five sets(s)					Average(s)
		1	2	3	4	5	
Harris	228	0.410	0.400	0.406	0.401	0.419	0.407
SIFT	252	1.101	1.110	1.102	1.102	1.125	1.108

(a) Original image (b) Harris corners (c) SIFT keypoints

Fig.3 Comparison of image feature detection: Original image, Harris corner detection, and SIFT keypoints

In contrast, SIFT offers higher feature uniqueness and accuracy but suffers from higher computational complexity and longer runtime. By improving the Harris method, we can retain its efficiency while enhancing accuracy and reducing false detections. This makes the improved method ideal for applications like rocket booster recovery, where both accuracy and real-time performance are essential. The enhanced approach effectively balances rapid computation with detection accuracy for efficient and precise feature extraction.

The traditional Harris algorithm detects more feature points in all test images than the improved adaptive Harris method. However, many of these points are redundant, which decreases overall accuracy (Fig.4). In contrast, the adaptive Harris algorithm detects slightly fewer points while minimizing redundancy, thereby enhancing feature extraction.



(a1) Fixed threshold detection results (a2) Adaptive threshold detection results

(a) Picture1

(b1) Fixed threshold detection results (b2) Adaptive threshold detection results

(b) Picture2

(c1) Fixed threshold detection results (c2) Adaptive threshold detection results

(c) Picture3

Fig.4 Comparison of fixed threshold and adaptive threshold detection results

The adaptive Harris method is also significantly more efficient, with shorter runtimes, making it better suited for rapid-response scenarios. It maintains detection quality while reducing processing time, ideal for real-time applications.

While the traditional Harris algorithm requires complex manual threshold adjustments, the improved algorithm employs adaptive thresholding for automatic adjustments based on image content, reducing tuning time and improving efficiency.

In terms of computational efficiency, the improved algorithm without NMS is about 50% more efficient than the traditional version (Table 2). Fig.5

provides a direct representation of the data. However, false corner detection persisted. The improved algorithm with NMS effectively reduces false corners while maintaining efficient feature extraction, balancing both efficiency and accuracy, making it suitable for applications that require both real-time performance and precision.

Table 2 Without NMS: Comparison between adaptive Harris and traditional Harris

Picutre	Algorithm	Number of corner points	Time(s)
Picture1	Harris(threshold=3×10^{-2})	153	0.469
	Our methods	131	0.219
Picture2	Harris(threshold=8×10^{-2})	322	0.459
	Our methods	277	0.211
Picture3	Harris(threshold=7×10^{-2})	199	0.410
	Our methods	152	0.184

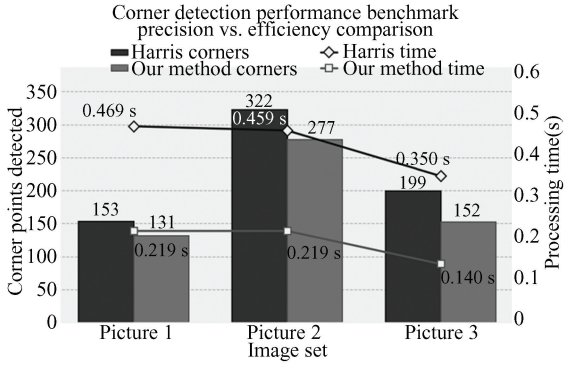


Fig.5 Comparative analysis of corner detection performance: Harris operator vs. proposed method

2) Robustness in noisy environments.

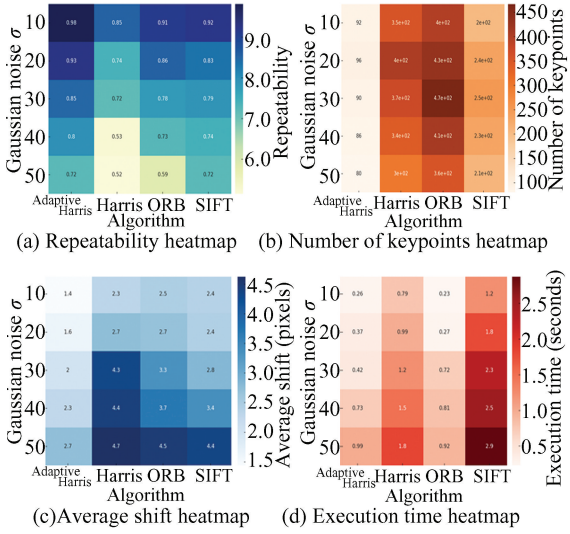
During the recovery of rocket boosters, Gaussian noise is a major factor affecting image quality and feature detection accuracy (Fig.6). High-speed flight and extreme environmental conditions, such as vibrations, flames, and smoke, make sensors vulnerable to thermal noise and electronic interference, resulting in Gaussian noise. This random noise can lead to numerous false corners and incorrect detections, thereby impacting the accuracy and robustness of critical operations such as attitude adjustment and precise landing.

To mitigate the effects of Gaussian noise, filtering techniques such as Gaussian or median filtering are often employed alongside improved feature extraction algorithms, adaptive thresholding, and NMS to enhance the stability and accuracy of feature point detection.

(a)Original image (b) Image with Gaussian noise

Fig.6 Comparison before and after Gaussian noise

Fig.7 compares ARH, Harris, ORB and SIFT under Gaussian noise levels σ = 10 – 50. As σ increases, ARH's repeatability gently decreases from 0.98 to 0.72—its overall average remaining above 0.85—whereas Harris falls sharply to 0.52 and ORB/SIFT drops to 0.59 and 0.72, respectively. All four methods show a slight initial rise in detected corner count, followed by a gradual decline as noise intensifies. In terms of localization accuracy, ARH keeps its mean displacement below 2.7 px, while Harris, ORB, and SIFT exhibit larger shifts of up to 4.7 px, 4.5 px, and 4.4 px, respectively. Computationally, ARH matches ORB in runtime, both affording high efficiency suitable for real-time use. Overall, ARH more consistently extracts structurally meaningful and spatially reliable keypoints in high-noise environments, thereby providing robust support for visual tracking in rocket booster recovery.



(a) Repeatability heatmap (b) Number of keypoints heatmap

(c)Average shift heatmap (d) Execution time heatmap

Fig.7 Heatmap comparison of algorithm performance under different levels of Gaussian noise

3) Performance in environments with motion blur and occlusion.

a) The descent of rocket boosters at high speeds (> 300 m/s) presents significant challenges for

imaging systems due to rapid motion and complex environmental factors. The motion blur primarily results from two sources: 1) Linear motion blur, caused by the high-speed linear movement of the booster, manifesting as smearing along the direction of motion; 2) Random jitter blur, induced by engine vibrations and atmospheric turbulence, leading to high-frequency pixel-level displacements. To simulate these effects, a linear blur filter with a kernel size of 15 was used. The high-speed motion-induced blur obscures feature points and reduces detection accuracy, while random jitter destabilizes image capture, further complicating the matching of feature points between frames. The performance of four algorithms—improved algorithm, Harris, SIFT, and ORB—was evaluated under these conditions, as shown in Fig.8.

b) Environmental factors during landing, such as smoke, dust, and variable lighting, significantly impact image quality. Smoke and dust obstruct the camera's view, particularly under low visibility conditions at night, which are simulated using a semi-transparenttexture overlay (intensity parameter = 0.5 corresponding to 50% transmittance). Changing lighting conditions due to altitude and cloud cover during flight alter brightness and contrast, complicating consistent feature recognition. These effects are simulated using Gaussian noise generation (σ = 50) and 21×21 Gaussian blur (σ = 10) to replicate smoke diffusion, as demonstrated in Fig.9.

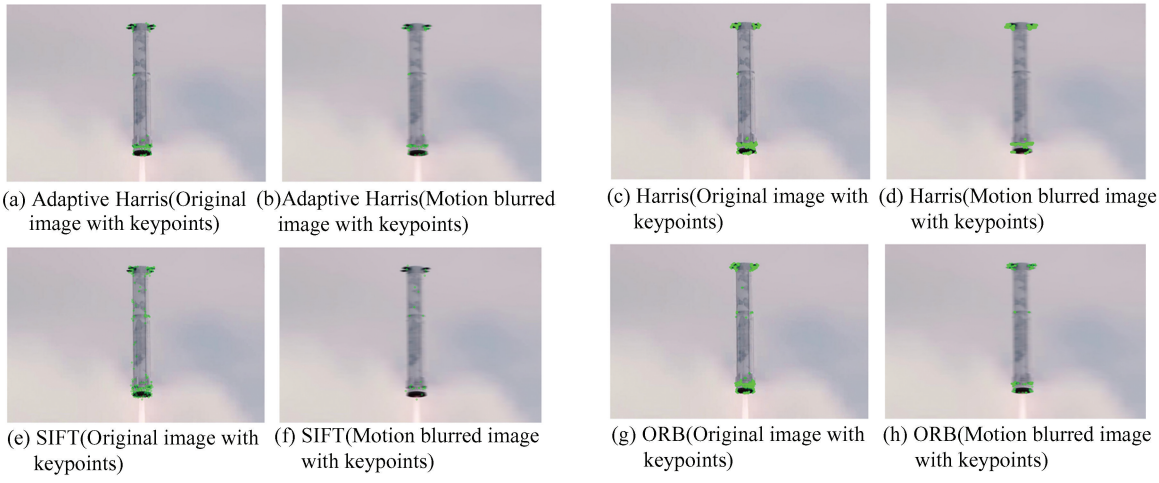


Fig.8 Comparison of keypoint detection by different algorithms on original and motion-blurred images

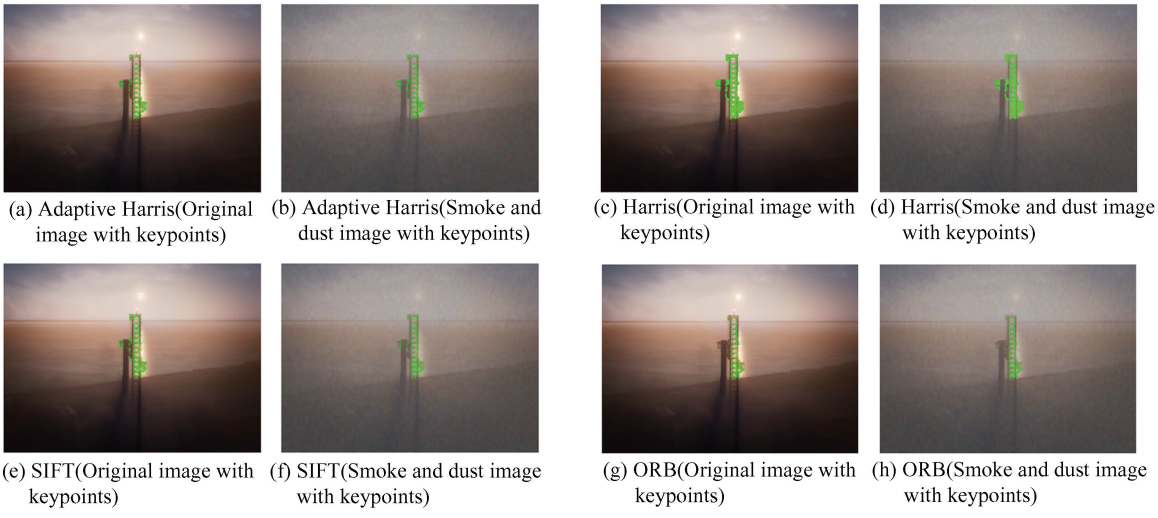
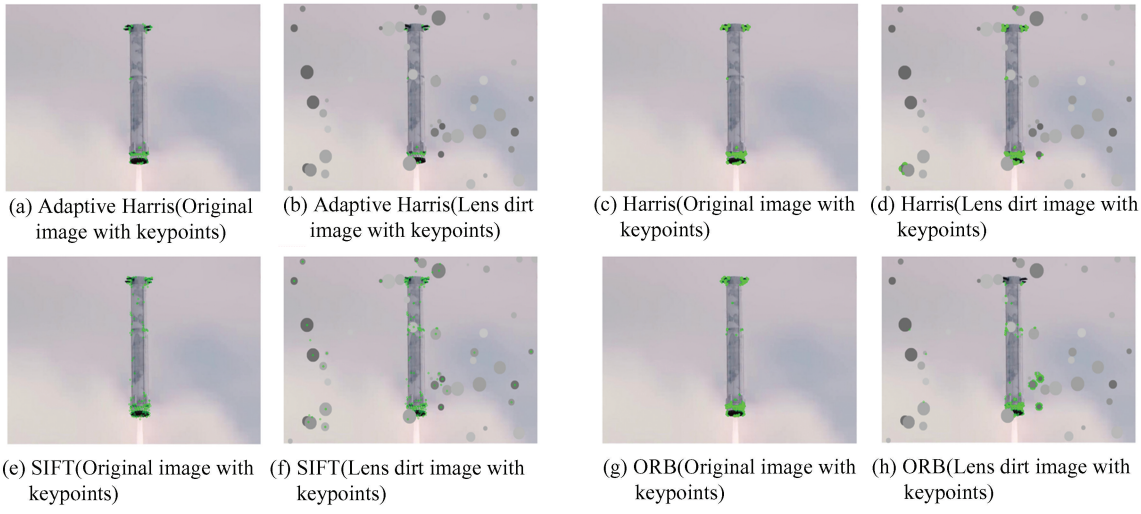


Fig.9 Comparative analysis of keypoint detection for different algorithms under original and smoke-and-dust disturbed image conditions

c) During rocket recovery, lens contamination from oil or water droplets presents significant challenges. These contaminants substantially degrade image quality, simulated in this study through randomly generated opaque spots (density 50, max radius 30 pixels) that obscure essential feature points. The spatial randomness of these spots is ensured by their completely random distribution, preventing any patterned artifacts. Additionally, the size diversity of the spots, with radii randomly varying between 5 and 30 pixels, simulates contaminants of different particle sizes. The optical scattering effect is modeled using medium gray tones (100 – 200), which balance absorption and reflection effects. Such contamination compromises the accuracy and stability of feature extraction (causing an average 35% loss of feature points and a 0.4-pixel increase in displacement error), thereby affecting precise positioning and control during recovery operations. The comparative performance of algorithms under these conditions is shown in Fig.10. These challenges underscore the critical need for feature detection algorithms to maintain robust in extremely complex environments to enable effective rocket booster imaging and control.

Table 3 presents a performance comparison of various corner detection algorithms under different disturbance conditions, including blur, lens dirt, and smoke and dust occlusion. The table provides key performance metrics for each algorithm, including the number of original and processed keypoints, keypoint ratio, repeatability, average offset, execution time, and a special metric for corner reduction. Figs. 11, 12, and 13 visually illustrate the metrics for each algorithm across the three different scenarios.



(a) Adaptive Harris(Original image with keypoints) (b) Adaptive Harris(Lens dirt image with keypoints) (c) Harris(Original image with keypoints) (d) Harris(Lens dirt image with keypoints)

(e) SIFT(Original image with keypoints) (f) SIFT(Lens dirt image with keypoints) (g) ORB(Original image with keypoints) (h) ORB(Lens dirt image with keypoints)

Fig.10 Detailed comparison of keypoint detection performance for different algorithms under original and lens dirt interference conditions

Table 3 Performance comparison of corner detection algorithms under various disturbance conditions

Algorithm	Test condition	Original keypoints	Processed keypoints	Keypoint ratio	Repeatability	Average offset	Time (s)	Special metric (corner reduction)
ARH	Blur	131	105	0.6824	0.7729	1.0168	0.2825	0.2000
	Lens dirt	131	133	–	0.9313	0.5853	0.4362	–
	Smoke & dust	331	337	–	1.0000	0.0458	0.3741	–
Harris	Blur	328	259	0.6145	0.8475	1.3337	0.4823	0.210
	Lens dirt	328	427	–	0.9051	0.6461	0.5121	–
	Smoke & dust	788	802	–	1.0000	0.0697	0.4656	–
SIFT	Blur	174	27	0.1552	0.2011	2.3308	1.2333	0.8163
	Lens dirt	174	333	–	0.8753	1.0173	1.1607	–
	Smoke & dust	538	498	–	0.4715	0.3971	1.0824	–
ORB	Blur	371	159	0.3376	0.7176	2.3322	0.1421	0.5714
	Lens dirt	371	478	–	0.9024	0.9452	0.1262	–
	Smoke & dust	476	500	–	0.9886	0.1294	0.1202	–

In scenarios involving motion blur, the improved ARH algorithm demonstrates excellent performance across multiple metrics (Fig.11). The ARH algorithm achieves the lowest feature point reduction rate of 20%, with a repeatability score of 0.7729, indicating strong feature retention and robustness to motion blur. Moreover, its computation time is only 0.2825 s, second only to ORB, providing high real-time performance. The traditional Harris algorithm detects the most feature points in the original image (328 points), many of which are redundant feature points and corner clusters. While it maintains the highest repeatability under motion blur (0.8475), its computation time is longer (0.4823 s), resulting in slightly lower real-time performance. The SIFT algorithm experiences a significant reduction in feature points (81.63%) under motion blur, reflecting its high sensitivity to blur. Its computation time is the longest, at 1.2333 s, which limits its applicability in such scenarios. In contrast, while ORB excels in computational efficiency (0.1421 s), it shows a higher feature point reduction rate of 57.14% and a repeatability score of 0.7176, indicating its high sensitivity to blur. Overall, the improved Adaptive Harris algorithm outperforms the others in terms of resistance to blur, feature point reduction, detection accuracy, and computational efficiency, making it particularly suitable for handling blur variations in high-speed moving objects such as rockets.

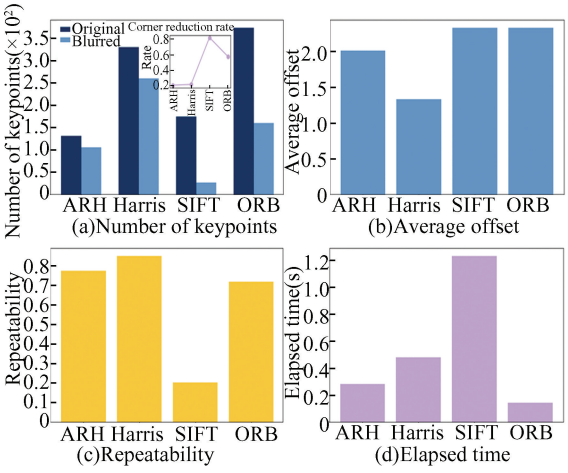


Fig.11 Detailed comparative analysis of keypoint detection performance for different algorithms under blurred conditions: Number of keypoints, repeatability, average shift, and execution time

In smoke and dust occlusion conditions, the ARH algorithm performs excellently (Fig.12). It strikes an effective balance between repeatability, robustness, localization accuracy, and computational efficiency. The algorithm demonstrates high repeatability (1.0) in occluded environments, ensuring stable feature point detection despite environmental changes. It maintains a nearly consistent number of detected points under smoke and dust conditions, showing strong adaptability in low-contrast scenarios. Furthermore, the average displacement of detected feature points is minimal (0.0458 pixels), indicating high localization precision, which is crucial for accurate matching. Although its execution time (0.3741 s) is slightly longer than ORB's, the adaptive Harris algorithm provides a good trade-off between accuracy and efficiency, ensuring real-time performance. Overall, the Adaptive Harris algorithm offers significant advantages in repeatability, localization accuracy, and computational efficiency, making it particularly well-suited for visual recognition and navigation in complex environments.

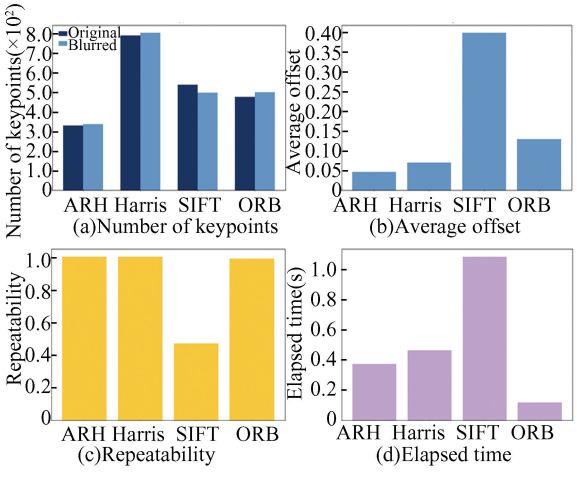


Fig.12 Detailed comparative analysis of keypoint detection performance for different algorithms under realistic smoke and dust conditions: Number of keypoints, repeatability, average shift, and execution time

In simulated lens stain occlusion conditions, key evaluation metrics include repeatability, the number of feature points (in both original and occluded images), average displacement, and execution time (Fig.13). While the traditional Harris algorithm achieves relatively high repeatability (0.905), it suffers from a considerable number of false detections under interference, and excessive corner clustering further compromises detection precision. In contrast, the improved adaptive Harris algorithm achieves a superior

balance between efficiency and robustness. It demonstrates higher repeatability (0.931) than the traditional Harris, and maintains a stable number of feature points under occlusion (ranging from 131 to 133), indicating strong adaptability to complex conditions. Furthermore, it achieves a moderate average displacement (0.585 pixels) with a substantially reduced execution time (0.4362 s), making it highly suitable for applications requiring both real-time performance and robustness. The SIFT algorithm identifies a large number of feature points under unobstructed conditions (174 points); however, it is prone to numerous false detections when lens stains are present, resulting in a high average displacement (1.017 pixels). Although SIFT is well-suited for environments demanding high localization accuracy, its significantly longer processing time renders it unsuitable for real-time applications. The ORB algorithm exhibits the shortest execution time (0.1262 s), offering excellent real-time performance and fewer false detections under occlusion. Nevertheless, it still suffers from occasional erroneous detections, which limits its applicability in highly complex environments.

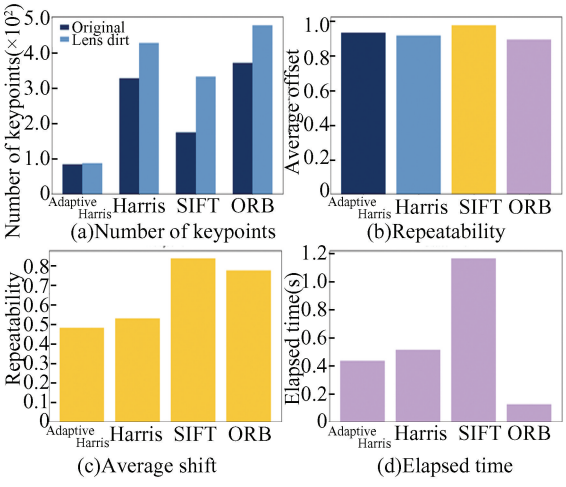


Fig.13 Detailed comparative analysis of keypoint detection performance for different algorithms under original and lens dirt interference conditions: Number of keypoints, repeatability, average shift, and execution time

In conclusion, the ARH algorithm performs excellently under various challenging conditions, including motion blur, smoke and dust occlusion, and lens stain occlusion. It strikes the optimal balance between accuracy, repeatability, localization precision, and computational efficiency, making it highly suitable for real-time applications in dynamic environments, such as visual recognition and navigation in rocket booster recovery. While the traditional Harris algorithm and SIFT offer high repeatability, their low computational efficiency or sensitivity to blur limit their applicability in real-time scenarios. ORB excels in computational efficiency but sacrifices feature detection accuracy and robustness. Overall, the ARH algorithm stands out for its ability to handle complex environmental variations, providing a robust solution for high-speed, high-precision tasks, making it suitable for real-time applications in real-world scenarios.

2.4 Other Evaluation Metrics

1) Rotational invariance.

During the rocket booster recovery process, the attitude control subsystem needs to precisely control the heading angle, especially during the approach phase (within a 300 m range). The booster's rotation around the longitudinal axis (0° − 180°) causes changes in the imaging perspective, which directly affect the spatial coordinates of feature points. To evaluate the robustness of the algorithm under rotational environments, we simulate different rotational postures through geometric transformations. The image rotation angles (θ) range from 0° to 180° to simulate the booster's rotation process, influencing the feature point detection capability of the visual system.

When the image is rotated by an angle θ, the transformation of the gradient is given by:

$$I'_x = I_x\cos(\theta) + I_y\sin(\theta) \tag{21a}$$

$$I'_y = -I_x\sin(\theta) + I_y\cos(\theta) \tag{21b}$$

where I_x and I_y represent the gradients of the image in x and y directions respectively, I'_x and I'_y are the rotated gradients.

After rotating the gradient, we obtain the new Harris matrix $\boldsymbol{M}$.

$$\boldsymbol{M}' = \begin{bmatrix} I'^2_x & I'_x I'_y \\ I'_x I'_y & I'^2_y \end{bmatrix} \tag{22}$$

The rotated Harris matrix is:

$$\boldsymbol{M}' = \boldsymbol{R}(\theta)\boldsymbol{M}\boldsymbol{R}(\theta)^{\mathrm{T}} \tag{23}$$

where $\boldsymbol{R}(\theta)$ is the rotation matrix. Since the determinant and trace of $\boldsymbol{M}$ remain invariant under similar transformations, the Harris response $\boldsymbol{R}$ also remains unchanged. Combined with the aforementioned improvements, this enhances the robustness of the detector under image rotations.

2) Comparative verification experiment.

A systematic evaluation of the performance of four feature detection algorithms with respect to image rotational invariance was conducted. The repeatability and variation in the number of keypoints under different rotation angles were examined for each algorithm. By analyzing the experimental data, we gain a more comprehensive understanding of each algorithm's feature extraction capability and stability in rotational scenarios. The algorithms evaluated include Adaptive Harris, ORB, Harris, and SIFT.

(1) Repeatability evaluation.

The adaptive Harris and ORB algorithms exhibited high repeatability, nearing 1.0 across rotation angles (Fig.14), demonstrating strong feature consistency and rotational invariance. In contrast, the Harris algorithm showed significant fluctuations at specific angles (e.g., 45° and 135°), indicating poor consistency and greater sensitivity to rotation. The SIFT algorithm maintained relatively stable repeatability but experienced notable drops at some angles.

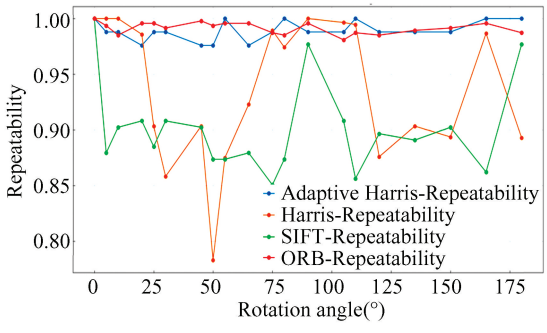


Fig.14 Rotation invariance evaluation for different algorithms: Repeatability vs. rotation angle

(2) Keypoint quantity evaluation.

The Harris algorithm displayed considerable variation in keypoint quantity, with sharp decreases at 45° and 175°, undermining reliability. The ORB algorithm maintained about 400 keypoints throughout rotation, showcasing good stability. The Adaptive Harris algorithm sustained around 200 keypoints with some fluctuations, outperforming traditional Harris. SIFT showed limited variations, maintaining about 100 keypoints, indicating stability but restricting its applicability (Fig.15).

(3) Comprehensive evaluation.

Rotational invariance: Adaptive Harris and ORB significantly outperformed others in repeatability, with ORB showing high consistency and excellent invariance. Adaptive Harris also approached 1.0, indicating strong matching reliability.

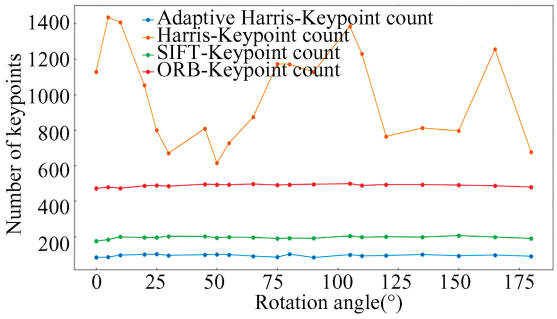


Fig.15 Keypoint count comparison for different algorithms at various rotation angles

Keypoint quantity stability: The Harris algorithm's fluctuations negatively impacted feature extraction quality, while ORB and SIFT maintained stability, with ORB excelling in consistent keypoint distribution. Adaptive Harris showed some fluctuation but overall superior performance compared to Harris.

Overall performance analysis: The ORB algorithm demonstrated strong stability in repeatability and keypoint quantity, making it effective for rotational invariance. Although Harris had advantages in repeatability, its unstable keypoint quantities hindered performance. SIFT's stable keypoint quantities had lower repeatability, limiting its effectiveness in rotational scenarios. The adaptive Harris algorithm, with its adaptive mechanism, offered high repeatability and stability, making it ideal for rotational transformations.

In summary, adaptive Harris and ORB excel at handling rotational changes, with ORB being the preferred choice for feature extraction due to its performance in both repeatability and keypoint quantity. Adaptive Harris also demonstrates strong adaptability, making it suitable for high-consistency tasks. In contrast, Harris and SIFT have limitations in stability, necessitating careful selection for practical applications.

3 Conclusions

In conclusion, the improved Harris corner detection algorithm (ARH) proposed in this study offers significant enhancements for vision-based applications, such as rocket booster recovery. By integrating adaptive thresholding, dilation-based non-maximum suppression, and sub-pixel refinement,

ARH achieves much stronger robustness against noise, motion blur, and variable lighting conditions, while maintaining high computational efficiency that meets the requirements of real-time operation. Compared with classical methods such as the standard Harris, SIFT, and ORB, ARH demonstrates superior repeatability, localization accuracy, and processing speed—making it particularly effective in challenging recovery environments featuring vibration, smoke, and dust.

However, the algorithm still has certain limitations, especially under extreme illumination changes and highly dynamic scenarios, where gradient computations may be negatively impacted. To tackle these issues and further expand the algorithm's applicability, future research will focus on incorporating deep learning-based feature representation techniques to enhance adaptability and detection accuracy in complex scenarios. Other promising research directions include multi-sensor fusion with radar or LiDAR to improve reliability, the development of better dynamic scene adaptation mechanisms, and further optimization of real-time performance on embedded platforms. These advancements are expected to broaden the application of ARH in various high-precision fields, such as visual tracking, target recognition, and autonomous navigation.

References

[1] Yang J D, Wang Q, Cong B, et al. Rocket booster recovery analysis. 2020 2nd International Conference on Artificial Intelligence and Advanced Manufacture (AIAM). Picataway: IEEE, 2020: 420 - 422. DOI: 10. 1109/AIAM50918.2020.00093.

[2] Arulkumar V, Jaya Prakash J, Subramanian E K, et al. An intelligent face detection by corner detection using special morphological masking system and fast algorithm. 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC). Piscataway: IEEE, 2021: 1556-1561.DOI: 10.1109/ICOSEC51865.2021.9591857.

[3] Myers V, Fawcett J. A template matching procedure for automatic target recognition in synthetic aperture sonar imagery. IEEE Signal Processing Letters, 2010, 17(7): 683-686. DOI:10.1109/LSP.2010.2051574.

[4] Yang H, Shao L, Zheng F, et al. Recent advances and trends in visual tracking: A review. Neurocomputing, 2011, 74(18): 3823-3831. DOI:10.1016/j.neucom.2011.07.024.

[5] Ma Z, Liu S. A review of 3D reconstruction techniques in civil engineering and their applications. Advanced Engineering Informatics, 2018, 37: 163 - 174. DOI: 10.1016/j.aei.2018.05.005

[6] Ramakrishnan S K, Al-Halah Z, Grauman K. Occupancy anticipation for efficient exploration and navigation. In: Vedaldi A, Bischof H, Brox T, et al. (eds) Computer Vision-ECCV 2020. ECCV 2020. Lecture Notes in Computer Science. Cham: Springer, 2020: 400-418. DOI: 10.1007/978-3-030-58558-7_24.

[7] Jing J, Liu C, Zhang W, et al. ECFRNet: Effective corner feature representations network for image corner detection. Expert Systems with Applications, 2023, 211: 118673. DOI:10.1016/j.eswa.2022.118673.

[8] Guo C, Li X, Zhong L, et al. A fast and accurate corner detector based on Harris algorithm. 2009 Third International Symposium on Intelligent Information Technology Application. Piscataway: IEEE, 2009, 2: 49-52. DOI: 10.1109/IITA.2009.311.

[9] Kadhim H A, Araheemah W A. A comparative between corner-detectors (Harris, Shi-Tomasi & FAST) in images noisy using non-local means filter. Journal of Al-Qadisiyah for Computer Science and Mathematics, 2019, 11(3): 86-93.

[10] Wu J, Cui Z, Sheng V S, et al. A comparative study of SIFT and its variants. Measurement Science Review, 2013, 13(3): 122-131. DOI: 10.2478/msr-2013-0021.

[11] Wang Y. Yuan Y, Lei Z. Fast SIFT feature matching algorithm based on geometric transformation. IEEE Access, 2020, 8: 88133-88140. DOI: 10.1109/ACCESS.2020.2989157.

[12] Zhou H, Yuan Y, Shi C. Object tracking using SIFT features and mean shift. Computer Vision and Image Understanding, 2009, 113(3): 345-352. DOI: 10.1016/j.cviu.2008.08.006.

[13] Li S, Wang Z, Zhu Q. A research of ORB feature matching algorithm based on fusion descriptor. 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC). Piscataway: IEEE, 2020: 417-420. DOI: 10.1109/ITOEC49072.2020.9141770.

[14] Rublee E, Rabaud V, Konolige K, et al. ORB: An efficient alternative to SIFT or SURF. 2011 International Conference on Computer Vision. Piscataway: IEEE, 2011: 2564-2571. DOI: 10.1109/ICCV.2011.6126544.

[15] Gupta S, Kumar M, Garg A. Improved object recognition results using SIFT and ORB feature detector. Multimedia Tools Applications, 2019, 78(23): 34157-34171. DOI: 10.1007/s11042-019-08232-6.

[16] Żak B, Hożyń S. Local image features matching for real-time seabed tracking applications. Journal of Marine Engineering & Technology, 2017, 16(4): 273-282. DOI: 10.1080/20464177.2017.1386266.

[17] Xu J, Chen X, Song X, et al. An improved Harris-FAST algorithm for underwater object corner detection. The 27th Chinese Control and Decision Conference (2015 CCDC).

Piscataway: IEEE, 2015: 5424 - 5428. DOI: 10.1109/CCDC.2015.7161763.

[18] Tang Z, Zhang Z, Feng J, et al. A fast image stitching algorithm based on texture classification and improved SIFT. IEEE Access, 2024, 12: 124183 - 124208. DOI: 10.1109/ACCESS.2024.3443111.

[19] Zhu J T, Gong C F, Zhao M X, et al. Image mosaic algorithm based on PCA-ORB feature matching. The International Archives Of the Photogrammetry, Remote Sensing and Spatial Information Science, 2020, 42: 83-89. DOI: 10.5194/isprs-archives-XLII-3-W10-83-2020.

[20] Luo C, Sun X, Sun X, et al. Improved Harris corner detection algorithm based on canny edge detection and gray difference preprocessing. Journal of Physics Conference Series, 2021, 1971 (1): Article number 012088. DOI: 10.1088/1742-6596/1971/1/012088.

[21] Semma A, Hannad Y, Siddiqi I, et al. Writer identification using deep learning with FAST Keypoints and Harris corner detector. Expert Systems With Applications, 2021, 184: 15473. DOI: 10.1016/J.ESWA.2021.115473.

[22] Wu T, Zhang Z, Zhao J, et al., An adaptive FAST corner detector based on high contrast grid image. 2nd International Conference on Mechatronics Engineering and Information Technology (ICMEIT 2017). Paris: Atlantis Press, 2017: 705-711.

[23] Vino G, Sappa A D. Revisiting Harris corner detector algorithm: a gradual thresholding approach. In: Kamel, M., Campilho, A. (eds) Image Analysis and Recognition. ICIAR 2013. Lecture Notes in Computer Science. Heidelberg: Springer, 2013: 354-363. DOI: 10.1007/978-3-642-39094-4_40.

[24] Oliveira A J, Ferreira B M, Cruz N A. A performance analysis of feature extraction algorithms for acoustic image-based underwater navigation. Journal of Marine Science and Engineering, 2021, 9(4): Article number 361. DOI: 10.3390/jmse9040361.

[25] Xia F, Jing B, Ma Y, et al. Application of composite tracking strategy based on template matching and Harris corner detection in missile terminal guidance system. 2024 IEEE 4th International Conference on Electronic Technology, Communication and Information (ICETCI). Piscataway: IEEE, 2024: 1048 - 1053. DOI: 10.1109/ICETCI61221.2024.10594355.

[26] Wang L, Han K, Sun H. An adaptive corner detection method based on deep learning. 2019 Chinese Control Conference (CCC). Piscataway: IEEE, 2019: 8478 - 8482. DOI: 10.23919/ChiCC.2019.8866560.

[27] Hussein A A. A comparison between Harris and FAST-corner detection of noisy images using adaptive non-local means. Diyala Journal for Pure Science, 2017, 13: 23-38. DOI: 10.24237/djps.1304.307a.

[28] Li Z, Wang J. An adaptive corner detection algorithm based on edge features. 2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC). Piscataway: IEEE, 2018, 2: 191 - 194. DOI: 10.1109/IHMSC.2018.10150.