



Article

Implementation of Real-Time Space Target Detection and Tracking Algorithm for Space-Based Surveillance

Yueqi Su ^{1,2,3} , Xin Chen ^{1,2}, Gaorui Liu ^{1,2}, Chen Cang ^{1,2} and Peng Rao ^{1,2,*}

¹ Shanghai Institute of Technical Physics, Chinese Academy of Sciences, Shanghai 200083, China; suyueqi@mail.sitp.ac.cn (Y.S.); chenxin@mail.sitp.ac.cn (X.C.); liugaorui@mail.sitp.ac.cn (G.L.); cangchen@mail.sitp.ac.cn (C.C.)

² Key Laboratory of Intelligent Infrared Perception, Chinese Academy of Sciences, Shanghai 200083, China

³ University of Chinese Academy of Sciences, Beijing 100049, China

* Correspondence: peng_rao@mail.sitp.ac.cn

Abstract: Space-based target surveillance is important for aerospace safety. However, with the increasing complexity of the space environment, the stellar target and strong noise interference pose difficulties for space target detection. Simultaneously, it is hard to balance real-time processing with computational performance for the onboard processing platform owing to resource limitations. The heterogeneous multi-core architecture has corresponding processing capabilities, providing a hardware implementation platform with real-time and computational performance for space-based applications. This paper first developed a multi-stage joint detection and tracking model (MJDTM) for space targets in optical image sequences. This model combined an improved local contrast method and the Kalman filter to detect and track the potential targets and use differences in movement status to suppress the stellar targets. Then, a heterogeneous multi-core processing system based on a field-programmable gate array (FPGA) and digital signal processor (DSP) was established as the space-based image processing system. Finally, MJDTM was optimized and implemented on the above image processing system. The experiments conducted with simulated and actual image sequences examine the accuracy and efficiency of the MJDTM, which has a 95% detection probability while the false alarm rate is 10^{-4} . According to the experimental results, the algorithm hardware implementation can detect targets in an image with 1024×1024 pixels in just 22.064 ms, which satisfies the real-time requirements of space-based surveillance.



Citation: Su, Y.; Chen, X.; Liu, G.; Cang, C.; Rao, P. Implementation of Real-Time Space Target Detection and Tracking Algorithm for Space-Based Surveillance. *Remote Sens.* **2023**, *15*, 3156. <https://doi.org/10.3390/rs15123156>

Academic Editor: Paolo Tripicchio

Received: 10 May 2023

Revised: 3 June 2023

Accepted: 14 June 2023

Published: 16 June 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: space target; heterogeneous multi-core system; detection and tracking; MJDTM; FPGA; DSP

1. Introduction

The term space target refers to all outer space objects, including nonfunctional spacecraft, spent upper stages, and space debris [1]. With the development of human activities, the amount of space debris is multiplying. The collision between different space targets such as space debris and spacecraft may lead to equipment damage and mission failure and even produce more space debris, which poses a significant threat to aerospace safety. Therefore, space target detection and tracking are essential to avoid space collision and ensure the operation safety of on-orbit spacecraft. Space situational awareness technology is important to guarantee on-orbit safety, monitoring space targets, evaluating space events and providing space situational information for on-orbit spacecraft using the space-based or ground-based detection equipment. Its primary mission is to accurately detect and track space targets and calculate important characteristic parameters such as the size and shape of space targets that may pose a threat to the on-orbit spacecraft [2].

Compared to ground-based observation, a space-based photoelectric detection system has the advantages of high maturity, high precision, and low energy consumption, making it possible to realize all-weather space target detection and on-orbit spacecraft protection [3]. The subject of the detection and tracking of space targets using optical

detection equipment consists of a set of problems that are central to the disciplines of space-based space target awareness. However, as space-based optical detection technology is upgraded, the detection field of view is gradually expanding, and space-based images are including increasingly complex information about the space environment. The existing methods have limited ability to suppress background noise in space-based images and are insufficient for space target perception, which generally concentrates on a single task such as detection or tracking. Therefore, developing a target detection algorithm with improved detection precision and a low rate of false alarms to separate space targets from the background is the critical problem of space target detection and tracking algorithms. Meanwhile, there are few space-based image processing system solutions presented by the current researchers, which have not been able to solve the problems of small space target detection and stellar target suppression, and the detection performance of the systems cannot keep up with the demand for the real-time processing of high-resolution space target images. The development of miniaturized, dedicated, high-speed processing systems for real-time space target detection with constrained on-orbit hardware resources remains a difficult task.

To achieve real-time space target surveillance, we proposed a high-precision detection and tracking architecture for space targets and developed a high-speed image processing platform to fulfill the algorithm implementation while maintaining real-time processing requirements. The main contributions of our paper are as follows. First, inspired by the human visual contrast mechanism, we improved the local feature contrast and energy concentration degree method to extract the potential small space targets in the optical image sequences. The local subtraction of the target detection algorithm suppresses background noise, and the accumulation of the target area boosts the target to achieve high-precision detection for space targets. Second, to eliminate false alarms of stellar targets with similar imaging features to the real space target, we proposed a stellar target suppression method that uses differences in motion relative to the Earth and real-time satellite attitude data to distinguish between the space and stellar targets. The algorithm is based on the historical coordinate data of the tracking trajectory and uses the platform parameters to determine the target type accurately. Finally, a comprehensive and lightweight space target perception architecture, called the multi-stage joint detection and tracking model (MJDTM), is given. It combines the space target detection method based on the LFC, the Kalman filter algorithm, and the proposed stellar target suppression method to accurately detect and track space targets. The architecture is implemented on a specialized heterogeneous multi-core processing platform based on FPGA and DSP. Additionally, the performance measures of the architecture and its implementation are evaluated using the simulated and real image sequences including computation time, resource usage, and detection capability.

The remainder of this paper is divided into the following sections. Section 2 summarizes related works on space target detection and tracking algorithms and existing hardware implementation schemes. In Section 3, the proposed multi-stage joint detection and tracking model is elaborated. In Section 4, we present the proposed hardware architecture and the algorithm implementation. Section 5 validates the performance and effectiveness of the proposed implementation. Section 6 gives the discussion of this paper. The conclusions are provided in Section 7.

2. Related Work

Various algorithms and relevant hardware implementations have been exploited for faint and tiny moving space target detection and tracking in space-based optical image sequences. Track-before-detect (TBD) and detect-before-track (DBT) are the dominating solutions to the difficulty of moving target detection and tracking. A dynamic programming approach was developed by BARNIV [4,5] that utilizes the velocity and shape information to detect linear moving objects with a low signal to noise ratio (SNR). The particle filter method [6] is a nonlinear dynamic filter based on the Monte Carlo method. A TBD algorithm has been realized using a Bayesian particle filter to approximate the posterior

probability distribution of the target state [7]. Reed et al. [8] established a dim and small target detection method based on three-dimensional matching filtering that matched and filtered the feature information of moving targets in the Fourier domain. These three methods described above could be defined as the TBD method. In actual scenarios, since the energy distribution and pattern of stellar and space targets are similar, it is challenging for the TBD approach to distinguish between them. Moreover, the variety of the target motion state will enhance the computational burden of the algorithm, making it hard for the TBD method to satisfy real-time application requirements. Accordingly, the DBT method is more suitable for space target surveillance in the space-based scenario.

The detection stage of the DBT method needs to extract the possible target and acquire the target region. The star map registration algorithm [9–12] is a common method for space target detection. In contrast, the satellite platform attitude variation increases the image background uncertainty and complexity, which makes the star map registration method unsuitable for space-based scenarios. Some threshold segmentation methods based on target enhancement have been studied, including the wavelet filtering method [13,14], local contrast method [15,16], and morphology filtering. Boccignone et al. [13] presented a small target detection method using wavelets. Jiang et al. [14] improved this method and developed an automatic space debris extraction algorithm. It utilized wavelet transform and variational hybrid filtering algorithms to suppress noise and detected candidate debris targets using the Hough transform. Mathematical morphology-based algorithms usually use image filters to eliminate background noise and enhance small targets, such as median filters [17], max-mean and max-median filters [18], and top-hat [19]. The local contrast method [15] is a powerful small target detection algorithm that was inspired by the human visual system contrast mechanism. It can enhance the target by calculating the local contrast map of the infrared image. Chen et al. [16] combined the local contrast method with energy concentration degree and proposed an infrared dim and small target detection algorithm. Lv et al. [20] developed a novel algorithm called neighborhood saliency map (NSM) based on the contrast mechanism of the human visual system. Han et al. [21] improved the local contrast method (LCM) and designed a detection architecture named multiscale tri-layer local contrast measure (TLLCM). The image filter algorithm based on the LCM method, which has been applied to the problem of infrared (IR) small target detection, could effectively boost the dim target and increase the detection accuracy. In recent years, researchers have also proposed deep learning-based solutions to the problem of dim and small target detection [22]. However, the network structure of these algorithms is frequently complex, and they frequently require a large quantity of experimental data to learn, making their implementation and application challenging.

Once the target has been extracted, it requires a tracker to predict the target motion state and update the trajectories in the subsequent frames. Fan Shi et al. [23] tracked a moving target using a primary scale invariant feature transform (P-SIFT) keypoint matching algorithm. In this way, the deviation of feature extraction will also have an impact on tracking. K. Fujita et al. [24] described a computer vision technique called an optical flow algorithm to detect and track GEO debris. However, its computational complexity makes it challenging to meet the requirements of real-time applications. The Kalman filter [25] is a classic target tracking algorithm used in dynamic procedures where the measured process is linear and Gaussian. Scala and Bitmeand [26] proposed the extended Kalman filter for solving the tracking problem when both the dynamic and measurement processes are nonlinear. Tao et al. [27] presented a space target surveillance algorithm that contains a variance detector and uses a Markov-based dynamic model to forecast the potential target position.

In addition to noise interference, hundreds of thousands of stars are the primary interference sources for space target detection in space-based detection scenarios. The image difference method [28] directly differentiates adjacent image frames, but when the platform moves, the imaging position of the stars changes, which will cause a false alarm for detection. The star mask frame method [29] employs multi-frame image accumulation

to calculate the position of a star and generates a star mask frame to filter out the stars in the image. However, detection fails when the target is near the imaging distance of the star. The star image recognition method [30] matches the image with the star map to extract the matching star point, but it is hard to implement it in hardware due to the heavy calculation burden. In this paper, a stellar target suppression method that uses differences in motion and real-time satellite attitude data is provided to distinguish between the space and stellar targets.

The target only takes up a small portion of the image pixels due to the large separation between space targets and detector, and the contrast between targets and background may not be strong enough for the detection method to utilize the texture feature data effectively. The space target detection method mentioned above can only solve the problem of target detection in some specific scenes, and it is challenging to overcome the problem of background stellar false alarms and strong noise in space-based scenes. In addition, these works have not been implemented by the hardware platform, and its real-time processing capability needs to be evaluated.

Moreover, the research community has designed some image processing systems using the limited hardware system resources that could implement the related target detection and tracking algorithms in the space-based scenario. A high-performance embedded processing platform based on a graphics processing unit (GPU), DSP, and FPGA has become the potential solution for onboard image processing [31–34]. As the specialized image processor, the embedded GPU processing platform [35] has been widely used in unmanned driving technology, AI computation, and video image processing. Its parallel processing capability supports it in handling complex data and geometry computing [36]. However, the disadvantages of poor independence and high power consumption hinder the application of GPUs in onboard applications. In the meantime, DSP and ARM processors with computing capability, flexibility, and large-scale integration have been adopted to implement the vision and image processing algorithms [37]. Sun et al. [38] described an onboard space debris detection approach on a multi-core DSP platform that can process a 2048×2048 image in 600 ms. The parallelism of this system constrains the throughput of processing data streams, making it challenging to process intensive computing with large data volumes. Over other embedded systems, the use of FPGAs in high-speed parallel data processing has become more prevalent due to their parallel processing capability. The FPGA platform is suitable for onboard image processing because of its flexibility, reconfigurability, and high energy efficiency [39]. Han et al. [40] proposed a high-speed tracking and measurement method for non-cooperative space targets and applied it to an FPGA-based space-embedded system. However, their scheme does not consider the situation of small space targets and establishes an overly ideal stellar interference model that may malfunction in practical space-based scenarios. Yang et al. [41] implemented the ATGP algorithm on FPGAs to achieve real-time target and anomaly detection in hyperspectral image sequences. Nevertheless, it is not feasible for FPGAs to implement high-precision data operations, and their programming development is complex. A heterogeneous processing platform based on FPGA and DSP is one of the most commonly used embedded image processing systems, and has been relatively maturely applied to the field of space-based image processing [42,43]. The high-speed parallel processing capability of FPGA has considerable advantages in large-scale image data processing. The DSP processor has the characteristics of large-scale integration and stability, which can realize high-precision digital signal processing.

For space-based surveillance, an integrated processing system with high flexibility, powerful processing performance, and low power consumption can quickly complete image processing. The current research in space target detection and tracking and other image processing performance needs to be improved, as it has failed to give a high-performance space target sensing method and its hardware implementation. However, a high-performance space target perception method and its hardware implementation have not been provided by the present research.

In this paper, we provide a complete space target perception architecture that realizes the accurate detection and tracking of small space targets. A space-based image processing system platform based on FPGA + DSP has been constructed to implement this architecture.

3. Methodology

A flow diagram of the MJDTM architecture is outlined in Figure 1. With the detection range extension of the space-based optical detector, there are more stellar targets and noise points in images, making it challenging to accurately identify space targets only occupying one to several pixels in the image plane. To ensure target detection accuracy and reduce the false alarm rate, the interference of the stellar targets and background noise points needs to be suppressed. The proposed space target perception architecture contains three main parts: space target detection and tracking, stellar target suppression, and target feature calculation. As shown in Figure 1, we first adopt an improved local contrast method to extract the potential space point target during the target detection and tracking stage. Then, the classical Kalman filter algorithm and the Hungarian matching algorithm are combined to predict the target state and correlate tracking trajectories. The sidereal targets with similar imaging properties to space targets are suppressed during the stellar target suppression stage. A schematic diagram of the image sequence after target detection tracking and stellar suppression is given in Figure 1. After that, the feature information of the space target that is confirmed as the real target is calculated. Details are as follows.

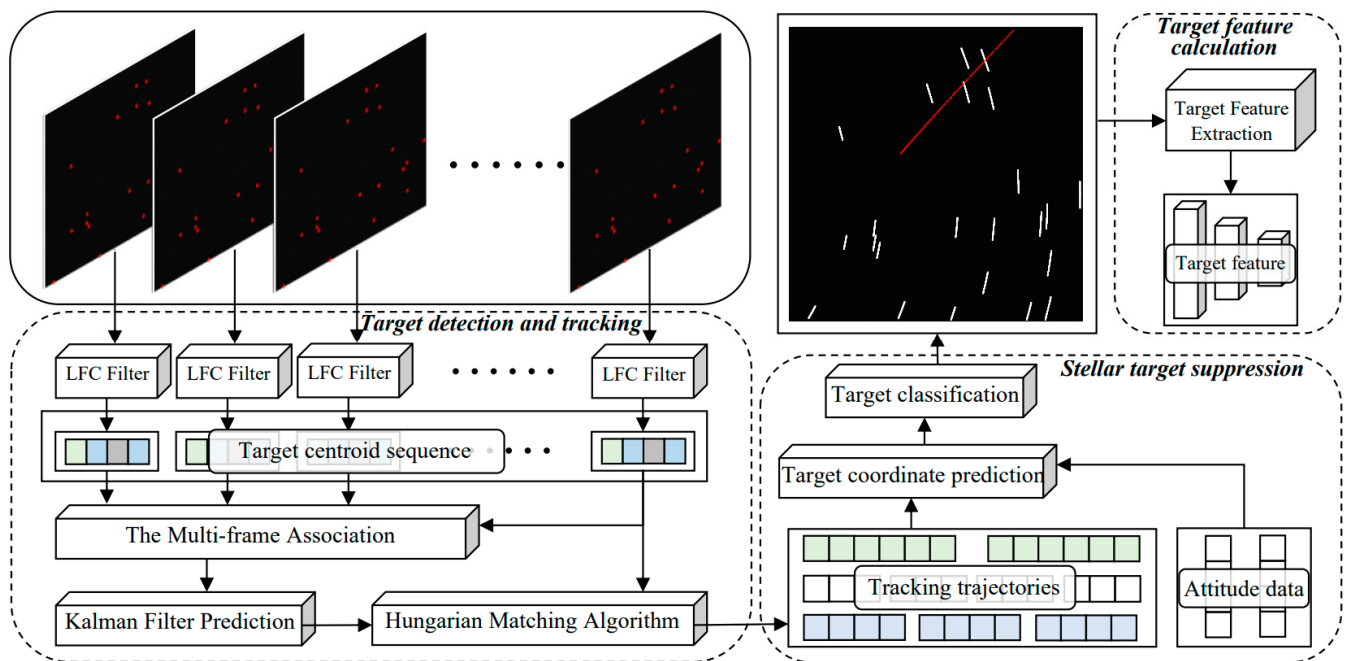


Figure 1. Workflow of the proposed architecture.

3.1. Target Detection and Tracking

3.1.1. Target Detection Algorithm

The space-based optical image of space can be modeled as follows:

$$F(i, j) = T(i, j) + S(i, j) + B(i, j) + N(i, j) \quad (1)$$

where (i, j) represents the pixel coordinates of the image and $F(i, j)$ denotes the grayscale value of the pixel coordinate (i, j) in the image. $T(i, j)$ and $S(i, j)$, respectively, denote the space target and the stellar targets, which obey the Gaussian distribution model. $B(i, j)$ represents the background of the deep space environment and $N(i, j)$ denotes the noise generated by the internal noise of the imaging system and external environment interference.

It can be seen from the above model that most of the image information obtained by the space-based space target detection equipment is from the deep space background environment. The space targets and stellar targets only account for a small part of the image, and various noise disturbances are randomly distributed throughout the image. Due to the limitation of the detection distance and the short exposure time, the space targets and stellar targets occupy only one pixel in the image and the energy of the target is weak. In order to accurately detect real space targets, the detection algorithm should enhance the target region for better target segmentation and extraction, and reduce the independent noise points on the image to lower the false alarm rate. All possible targets in the image must be segmented during the target detection phase to avoid missing actual space targets since the space target and stellar target have very similar imaging characteristics.

In this paper, the target detection algorithm uses the target energy feature and the local standard deviation feature to establish the LFC model and employs this model to realize the image filtering and the detection of the space and stellar targets. The local contrast method [15] is an image-filtering method based on the contrast mechanism of the human visual system and is commonly used to solve IR dim target detection problems. Similar to IR faint targets, space targets in space-based optical image sequences have weak energy and occupy only a few pixels without shape and texture features. Therefore, the detection rate could be guaranteed by using this technique to locate space targets in the deep space background. Chen et al. [15] proposed a local feature contrast and energy concentration degree method (LFC-ECD) that combines the LCM algorithm with the energy concentration algorithm to detect small infrared targets. This algorithm suppresses the neighboring regions of the target through local subtraction and performs energy accumulation to enhance the faint target. In the target detection stage, we remove the energy accumulation progress of the LFC-ECD and use the local feature contrast filter to extract space targets. The specific steps are as follows.

By sliding the local window on the whole image, the local feature contrast value of each point in the image is calculated. Firstly, the image slice larger than the target region is selected as the local window. Additionally, the slice of coordinate (x, y) in the image is divided into the target region S_0 and the neighboring region S , which is shown in Figure 2.

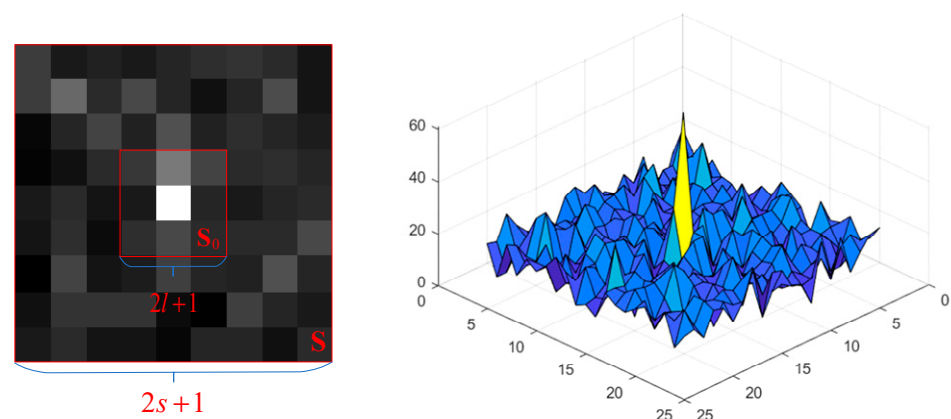


Figure 2. The target and its neighboring region.

The region of S_0 and S are represented as follows:

$$R_S = \{(p, q) | \max(|p - x|, |q - y| \leq s)\}, \quad s = 4, 7, 10, 13 \quad (2)$$

$$R_{S_0} = \{(i, j) | \max(|i - x|, |j - y| \leq l)\}, \quad l = 1, 2, 3, 4 \quad (3)$$

where s and l represent the radius of S_0 and S , respectively, and (i, j) and (p, q) denote the pixel coordinates of S_0 and S in the image.

The pixel grayscale average value and standard deviation in the region S are computed to represent the background and noise of the target region, and the formula is denoted as follows:

$$G_m(x, y) = \sum_{(p,q) \in R_s} \frac{G(p, q)}{(2s + 1)^2} \quad (4)$$

$$G_s(x, y) = \sqrt{\frac{\sum_{(p,q) \in R_s} [G(p, q) - G_m(x, y)]^2}{(2s + 1)^2}} \quad (5)$$

where $G(p, q)$ is the pixel grayscale at (p, q) in the region S and $G_m(x, y)$ and $G_s(x, y)$ are the pixel grayscale average value and standard deviation at (x, y) in the region S , respectively.

Then, the background should be inhibited by the regional background subtraction because of the solid local continuity, and the formula is represented as follows:

$$G_t(x, y) = G(x, y) - G_m(x, y), (i, j) \in S_0 \quad (6)$$

where $G(x, y)$ denotes the grayscale of the pixel at (i, j) in the region S_0 and $G_t(x, y)$ represents the grayscale of the pixel at (i, j) in the region S_0 after the background suppression.

When the target is weak, the pixel grayscale of the region S_0 will be low after the background subtraction. Therefore, the target component should be magnified by the energy accumulation to ensure the target is detected correctly. The formula of energy accumulation is denoted as follows:

$$E_t(x, y) = \sum_{(i,j) \in R_{S_0}} G_t^2(i, j) \quad (7)$$

where $E_t(x, y)$ denotes the energy accumulation value of the pixel grayscale in the region S_0 at (x, y) . The sum operation helps in the rapid enhancement of targets.

Finally, the local feature contrast value of the coordinates (x, y) in the image is provided in the following formula:

$$G_c(x, y) = E_t(x, y) / G_s(x, y) \quad (8)$$

$$G_L(x, y) = G_c(x, y) \times G_t(x, y) \quad (9)$$

where $G_c(x, y)$ represents the contrast factor and $G_L(x, y)$ denotes the values of local feature contrast at the coordinates (x, y) .

When obtaining the local feature contrast, the adaptive threshold segmentation will be conducted on the local feature contrast image to segment the target. The adaptive threshold T_1 is denoted as follows:

$$T_1 = m_L + k_1 \times std_L \quad (10)$$

where m_L and std_L represent the average value and standard deviation of local features' contrast $G_L(x, y)$. The range of the parameter k_1 confirms that a range of 30 to 40 is efficacious. In Section 5.1, the selection of the parameter k_1 will be discussed in depth.

Then, the binary image of the LFC result is segmented by the threshold T_1 . The formula is represented as follows:

$$b_1(i, j) = \begin{cases} 1, & G_L(i, j) > T_1 \\ 0, & G_L(i, j) \leq T_1 \end{cases} \quad (11)$$

where $b_1(i, j)$ denotes the grayscale at the coordinates (i, j) in the segmented image and $G_L(i, j)$ is the local feature contrast value at the coordinates (i, j) in the image.

In the ideal optical system, only one pixel of the detector is occupied by the point target. However, the targets will diffuse into several pixels due to circular aperture diffraction in practical situations. The precise pixel coordinates of the target are confirmed by the center location of the gray pixel. Since the target is susceptible to the effects of ambient background

noise, the precision of target positioning might be impacted by the classic centroid method. The centroid coordinates of targets are calculated using the distance-weighted centroid method that enhances the conventional centroid with distance-weighting. Based on the conventional centroid approach, the distance-weighted centroid method adds the grayscale and distance influence factor as the weight to lessen the impact of target edge noise on target location extraction. The steps of this method are described below.

Firstly, the maximum grayscale pixel coordinate data in the target region are provided by the target detection stage result. The target region S_t is separated by extending m pixels outward from the center of the maximum pixel gray value coordinates. The size of the target area is $n = 2 \times m + 1$. Moreover, the distance $D(i, j)$ between the maximum grayscale pixel and each pixel in the target area is defined by the following formula:

$$D(i, j) = \sqrt{(i - x)^2 + (j - y)^2} \quad (i, j \in S_t) \quad (12)$$

where (i, j) are the pixel coordinate data in the target region and (x, y) represent the maximum pixel grayscale value coordinates.

The formula of the distance weight $D'(i, j)$ is defined as:

$$D'(i, j) = 1/D(i, j) \quad (13)$$

where $D'(i, j)$ is the distance weight at the coordinates (i, j) in the target area. The distance weight at the maximum grayscale pixel is a ($3 \leq a \leq 5$).

The following formula computes the distance-weighted centroid coordinates of the target:

$$X = \frac{\sum_{(i,j) \in R_{S_t}} G(i, j) D'(i, j) i}{\sum_{(i,j) \in R_{S_t}} G(i, j) D'(i, j)} \quad (14)$$

$$Y = \frac{\sum_{(i,j) \in R_{S_t}} G(i, j) D'(i, j) j}{\sum_{(i,j) \in R_{S_t}} G(i, j) D'(i, j)} \quad (15)$$

where the $G(i, j)$ is the grayscale value at the coordinates (i, j) in the target area and (X, Y) is the target centroid coordinates calculated through the distance-weighted centroid method.

3.1.2. Target Tracking Algorithm

The target coordinate sequence of each frame can be obtained after the target detection for the optical image sequence. We establish tracking trajectories for each potential target during the tracking stage, estimate the candidate target motion state, and predict the target position using the Kalman filter to track space targets steadily. Moreover, the Hungarian matching algorithm is adopted to correlate the tracking trajectories with the target sequence to update the target coordinate positions. The specific operation flow is shown in Figure 3.

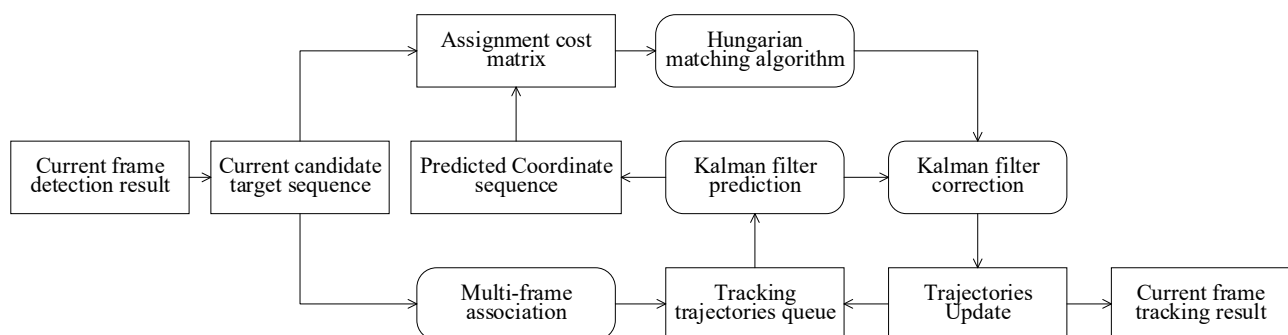


Figure 3. Procedure of the tracking stage.

Affected by the gravity of the Earth, both the space targets and the observation satellite platforms will run in a specific orbit. Thus, we can use a linear uniform model

to simulate the motion of the space target relative to the platform, which is unrelated to other targets and camera motion. It can be assumed that the running path of the space target in the continuous space image sequence is connected and that the target detection results from earlier frames can be used to estimate the target motion model and forecast the target position.

Before the target motion state prediction and tracking trajectory association, the multi-frame association operation as shown in Figure 3 will confirm the current candidate target queue, which reduces the calculation amount of the subsequent tracking process while suppressing noise points. The target that satisfies the trajectory creation condition initializes the corresponding tracking trajectory after this operation. The specific processing steps are as follows.

First, based on the detection result of the first frame image, a suspicious target queue TS_s will be created for each target in the current candidate target queue TS_c .

Then, the Euclidean distance between each target in the suspicious target queue TS_s and the current candidate's target queue TS_c will be calculated after the detection of the subsequent image frames. The specific formula is as follows:

$$TD(n, m) = \sqrt{(i_n - x_m)^2 + (j_n - y_m)^2}, n = 1 \dots N, m = 1 \dots M. \quad (16)$$

where $TD(n, m)$ is the Euclidean distance between the target $TS_s(n)$ in the suspicious target queue TS_s and the target $TS_c(m)$ in the current candidate target queue TS_c , (i_n, j_n) is the coordinate data of the target $TS_s(n)$, (x_m, y_m) is the coordinate data of the target $TS_c(m)$, and N and M are the numbers of targets in the queue TS_s and TS_c , respectively.

For a target in the suspicious target queue TS_s , if there is a target within the predetermined distance threshold range in the candidate target queue TS_c , the number of target occurrences is determined to increase. Then, the target coordinate data and the number of target occurrences in the suspicious target queue TS_s will be updated. If the current candidate target queue does not contain a target that fulfills the criteria, the number of target disappearances will be updated.

After the multi-frame suspicious target queue update, there will be a target in the suspicious target queue TS_s that appears more than the set threshold. It can be assumed that this is a real target rather than an independent noise point. For the real target, we adopt the Kalman filter to estimate the motion state and predict the coordinate position to achieve stable tracking of the target. The Kalman filter [25] is an optimal estimation algorithm for system state that uses the linear system state equation and system input and output observation data. It has been widely applied in the fields of orbit calculation [44], target tracking, and navigation [45], such as calculations of spacecraft orbit, tracking of maneuvering targets, and positioning of GPS. The specific calculation steps of the Kalman filter are as follows.

The tracking trajectory of this target will be initialized for subsequent target tracking and trajectory update. The invalid targets that disappear more than the set threshold in the suspicious queue will be cleared. When a trajectory is created in the target trajectory queue, the motion state estimation and target prediction of the target will be performed in subsequent frames as shown in Figure 3. The state of the target is defined according to the following model:

$$x_k = [u, v, p, q]^T \quad (17)$$

where u and v represent the horizontal and vertical coordinates of the target centroid and p and q represent the velocity component of the coordinate. The Kalman filter algorithm predicts the target position in subsequent frames according to the target state and updates the target state according to the measured value associated with the current target queue using the Hungarian matching algorithm. If a target has no correlation matching, its state is simply predicted using the linear velocity model without any correction.

One of the most well-known Bayesian filter theories is the Kalman filter, a linear optimal status estimate technique [46]. The estimation process of the Kalman filter consists

of the previous prediction step and the current measurement step. It includes two types of equations: status equation and observation equation. A dynamic model with the status and observation equations is given using a precise estimation that has been measured and altered. The status equation of the Kalman filter is represented as follows [47]:

$$x_k = Ax_{k-1} + Bu_k + w_k \quad (18)$$

where A is the status transition matrix, B is the control-input matrix, x_k is the status vector, u_k is the system control matrix, and w_k is the system noise vector.

The Kalman filter observation equation is defined as follows:

$$z_k = Hx_k + v_k \quad (19)$$

where H is the observation matrix, z_k is the observation vector, and v_k is the observation noise vector. w_k and v_k are assumed to be zero-mean Gaussian white noise with covariance Q and R , respectively, denoted as:

$$w \sim N(0, Q) \quad (20)$$

$$v \sim N(0, R) \quad (21)$$

When a discrete control process system satisfies the above conditions, the Kalman filter algorithm can be used to predict the system state.

The calculation procedure of the algorithm is as follows.

Firstly, the prediction equation is used to predict the next state of the system. The prediction equation is defined as follows [47]:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \quad (22)$$

where $\hat{x}_{k-1}$ represents the posterior status estimation combined with the measurements at the moment of $k-1$ and $\hat{x}_k^-$ denotes the prior status estimation derived from the status transition equation at the moment of k .

Then, the error covariance is calculated using the update equation. The update equation is as follows:

$$P_k^- = AP_{k-1}A^T + Q \quad (23)$$

where P_k^- is the prior estimation deviation covariance of the status $\hat{x}_k^-$, P_{k-1} is the posterior estimation deviation covariance of the status $\hat{x}_{k-1}$, and Q is the deviation covariance of the system noise vector.

The trajectory correlation operation follows the aforementioned prediction stage. As shown in Figure 3, the Hungarian matching algorithm is used to correlate the predicted coordinate sequence and the current candidate target sequence. The Hungarian matching algorithm [48] was proposed by two Hungarian mathematicians and is mainly used to solve some problems related to bipartite graph matching, such as data association [49], UAV task assignment [50], and multi-target tracking [51]. The core of the algorithm is to use the augmented path to find the maximum matching algorithm of the bipartite graph. The predicted coordinate sequence of the tracking trajectory and the current candidate target sequence form a bipartite graph that can be easily represented by a distance matrix.

Specifically, the Euclidean distance between the target prediction coordinates of the tracking trajectory sequence and the target coordinates in the current candidate target

sequence is calculated and integrated into a distance matrix ED , as shown in Figure 4a. The formula is as follows:

$$ED = \begin{pmatrix} Ed_{1,1} & Ed_{1,2} & \cdots & Ed_{1,n} \\ Ed_{2,1} & Ed_{2,2} & \cdots & Ed_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ Ed_{m,1} & Ed_{m,2} & \cdots & Ed_{m,n} \end{pmatrix} \quad (24)$$

$$Ed_{m,n} = \sqrt{(i_n - x_m)^2 + (j_n - y_m)^2}, \quad n = 1 \dots P, \quad m = 1 \dots C. \quad (25)$$

where $Ed_{m,n}$ is the Euclidean distance between the target $PC(n)$ in the predicted target coordinate sequence PC and the target $CT(m)$ in the current candidate target sequence CT , (i_n, j_n) is the coordinate data of the target $PC(n)$, (x_m, y_m) is the coordinate data of the target $CT(m)$, and P and C are the numbers of targets in the sequence PC and CT , respectively.

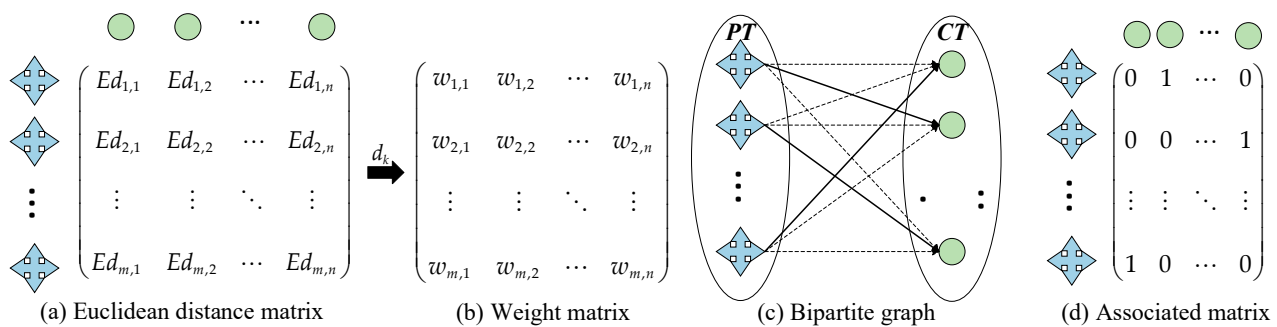


Figure 4. Diagram of the Hungarian matching algorithm.

Then, as shown in Figure 4b, the distance matrix will be transformed into a registration weight matrix WM according to the following formula:

$$WM = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,1} & w_{m,2} & \cdots & w_{m,n} \end{pmatrix}, \quad w_{m,n} = \begin{cases} dk - Ed_{n,m}, & Ed_{n,m} \leq dk \\ 0, & Ed_{n,m} > dk \end{cases} \quad (26)$$

where dk is the distance threshold parameter of the weight matrix. When the distance between the coordinates is close, we hope that the corresponding correlation weight is large, so the weight matrix should be inversely proportional to the distance matrix. At the same time, we introduce the parameter dk to limit the correlation of the target coordinates far away. If the distance between the targets is greater than dk , it is considered that the possibility of a large difference between the two coordinates is low, and the corresponding weight is set to zero directly to simplify the correlation calculation. After this operation, the weight of the coordinates with smaller distances becomes larger, and the possibility of registration association is also enhanced.

Through iterative optimization, the Hungarian matching algorithm generates a maximum weight distribution matrix, as shown in Figure 4c,d, which represents the correspondence between the target in the tracking trajectory and the latest subsequent target sequence. Each tracking trajectory is assigned to a current candidate target so that the posterior state can be calculated and updated according to the associated measurement. The Euclidean distance between the target predicted coordinate of the tracking trajectory sequence and the target coordinate in the current candidate target sequence is calculated and sorted into the correlation cost matrix. The assignment problem between the tracking trajectory and the current candidate target sequence is solved optimally using the Hungarian algorithm, which can provide the best matching for the two sequences.

For the tracking trajectory with correlation detection, the correction stage combines its predicted state with the measured value to obtain the best estimation x_k . The formula is represented as follows:

$$x_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \quad (27)$$

where x_k is the optimal estimation at the moment of k and K_k is the Kalman gain matrix. The formula of K_k is denoted as follows:

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \quad (28)$$

where R is the deviation covariance of the observation noise vector.

The posterior estimation deviation covariance of the state x_k is calculated using the following formula to keep the Kalman filter running until the processing system is finished.

$$P_k = (I - K_k H) P_k^- \quad (29)$$

where P_k is the filter deviation matrix and I is the unit matrix.

The target parameters in the target tracking trajectory queue are updated after the tracking association operation in each frame, including the coordinate data of the target and the number of occurrences. Similar to the suspicious target queue, when the number of target disappearances in the tracking trajectory exceeds the predetermined threshold, the target is removed from the tracking trajectory. This procedure stops the unrestricted expansion of the tracker population and positioning inaccuracies brought on by excessively extended forecast durations without detector correction.

3.2. Stellar Target Suppression Algorithm

As the target detection and tracking stage run alternately, the number of target historical coordinates in the tracking trajectory queue increases cumulatively. Both the space target and the stellar target are included in the trajectory queue. This section proposes a method for classifying the stellar targets and space targets using real-time satellite attitude data and the historical coordinate data of track trajectories. We postulate that due to the remote distance between the sidereal target and the Earth, the positions of the stars relative to the Earth remain unchanged for a short time. In contrast, the coordinate of the space targets relative to the Earth will change during this time because the moving space target has a certain velocity and is closer to the Earth. Therefore, we exploit the different motion states of stellar and space targets relative to the Earth to suppress the stellar targets. The specific methods are as follows.

For the candidate trajectory formed at the time t , this stage performs the subsequent operations on the latest target coordinates of each trajectory. The latest target point coordinates of a trajectory need to be transferred to the camera coordinate using the camera's intrinsic matrix. Since the plane image can only provide two-dimensional coordinate data, it is difficult to acquire the distance data of the target. During the process of coordinate transformation, we assume the Z-axis data of the target point to be 1.

$$\begin{bmatrix} x_c(t) \\ y_c(t) \\ z_c(t) \end{bmatrix} = R_{Int}^{-1} \begin{bmatrix} x(t) \\ y(t) \\ 1 \end{bmatrix}, R_{Int} = \begin{bmatrix} f/dx & 0 & x_0 \\ 0 & f/dy & y_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (30)$$

where R_{Int} is the intrinsic matrix of the camera, R_{Int}^{-1} is the inverse of the intrinsic matrix, $[x_c(t), y_c(t), z_c(t)]$ is the camera coordinate of the target, $[x(t), y(t)]$ is the pixel coordinate of the target at the time n , (dx, dy) is the focus of the camera on the X and Y axis, and (x_0, y_0) is the center pixel coordinate value of the camera.

The installation matrix calculates the target coordinate relative to the platform. The formula is as follows:

$$\begin{bmatrix} x_s(t) \\ y_s(t) \\ z_s(t) \\ 1 \end{bmatrix} = R_{Ins} \begin{bmatrix} x_c(t) \\ y_c(t) \\ z_c(t) \\ 1 \end{bmatrix}, R_{Ins} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x & o_x & a_x & t_x \\ n_y & o_y & a_y & t_y \\ n_z & o_z & a_z & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (31)$$

where R_{Ins} is the camera's installation matrix and $[x_s(t), y_s(t), z_s(t)]$ denotes the target coordinates relative to the platform.

The target coordinates relative to the Earth are calculated using the satellite attitude matrix at the time n . The formula is as follows:

$$\begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} = R_{Rot}(t) \begin{bmatrix} x_s(t) \\ y_s(t) \\ z_s(t) \end{bmatrix} \quad (32)$$

where $[x_e(t), y_e(t), z_e(t)]$ represents the target coordinates relative to the Earth t and $R_{Rot}(t)$ represents the satellite platform attitude matrix at the time n . This formula converts the target coordinate relative to the camera to the platform coordinate system using the camera external parameter matrix and the fourth-dimensional coordinate data are added to facilitate the calculation. The formula for the attitude matrix R_{Rot} is as follows [52]:

$$R_{Rot} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \quad (33)$$

where (q_0, q_1, q_2, q_3) denotes the attitude quaternion matrix of the satellite.

The target coordinates relative to the Earth can be obtained by transforming the target coordinates. The target coordinate data relative to the platform at the time $t + 1$ is predicted by the satellite attitude matrix at the time $t + 1$. The formula is as follows:

$$\begin{bmatrix} x_s(t+1) \\ y_s(t+1) \\ z_s(t+1) \end{bmatrix} = R_{Rot}^{-1}(t+1) \begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} \quad (34)$$

where $[x_s(t+1), y_s(t+1), z_s(t+1)]$ denotes the target coordinates relative to the platform at the time $t + 1$, $[x_e, y_e, z_e]$ represents the coordinates of the target relative to the Earth, and $R_{Rot}^{-1}(t+1)$ represents the satellite attitude inverse matrix at the time $t + 1$.

The camera coordinates are predicted by the coordinates of the target relative to the platform and the installation matrix. The formula is as follows:

$$\begin{bmatrix} x_c(t+1) \\ y_c(t+1) \\ z_c(t+1) \end{bmatrix} = R_{Ins}^{-1} \begin{bmatrix} x_s(t+1) \\ y_s(t+1) \\ z_s(t+1) \end{bmatrix} \quad (35)$$

where $[x_c(t+1), y_c(t+1), z_c(t+1)]$ represents the target camera coordinate data at the time $t + 1$ and R_{Ins}^{-1} denotes the inverse of the installation matrix. The target coordinates $[x(t+1), y(t+1)]$ at the time $t + 1$ are predicted by the target camera coordinates at the time $t + 1$ and the camera internal reference matrix. The formula is as follows:

$$\begin{bmatrix} x(t+1) \\ y(t+1) \\ 1 \end{bmatrix} = R_{Int} \begin{bmatrix} x_c(t+1) \\ y_c(t+1) \\ z_c(t+1) \end{bmatrix} \quad (36)$$

where $[x(t+1), y(t+1)]$ denotes the target pixel coordinates at the time $t + 1$.

After a sequence of coordinate transformations, we can obtain the predicted target coordinates of the tracked trajectory at the time $t + 1$. The actual target coordinate data in the track trajectories at the time $t + 1$ are provided after the tracking association operation in the above section. The difference between the target predicted and practical coordinates can be used as a criterion to determine whether the target is a real space target or not. When the difference exceeds a set threshold, the target is judged to be a real space target; otherwise, it is a stellar target. The threshold to confirm the target type is determined based on prior experience, which is selected as 1 in this paper.

3.3. Target Angle Calculation

The stellar target suppression module described in the above section classifies the space targets and stellar targets in the tracking trajectory queue. The calculation method of space target angle information will be introduced in this section. The position of the target with respect to the optical axis of the camera determines the azimuth and pitch angle of the target, and the precise calculation procedures are as follows.

Firstly, the intrinsic matrix of the camera is used to convert the target coordinate data from the image pixel coordinate system to the image physical coordinate system, as defined in the following formula.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = R_{Int} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1/dx & 0 & u_0 \\ 0 & 1/dy & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (37)$$

where (u, v) denotes the coordinate value of the target in the image pixel coordinate system, (x, y) indicates the coordinate value of the target in the image physical coordinate system, and R_{Int} is the intrinsic matrix of the camera.

Due to the lens distortion of the optical system, it is necessary to correct the target coordinate to ensure the accuracy of the target angle calculation. The radial and tangential distortion [53] are the major factors affecting the imaging quality of the wide field view optical system. The radial distortion can be fitted by quadratic and higher-order polynomial functions linked to the separation between target point coordinates and the image center pixel coordinates, as shown in the following formula [54]:

$$\begin{cases} x_d = x(1 + k_1r^2 + k_2r^4 + k_3r^6) \\ y_d = y(1 + k_1r^2 + k_2r^4 + k_3r^6) \end{cases}, r^2 = x^2 + y^2 \quad (38)$$

where (x_d, y_d) indicates the point coordinates after the radial distortion, (x, y) represents the coordinate value of the target in the image physical coordinate system, r^2 is equivalent to the distance between the coordinate point and the image center, and (k_1, k_2, k_3) denotes the parameters of the radial distortion model.

The tangential distortion is similar to radial distortion, which can be fitted using two other parameters, as shown in the following formula.

$$\begin{cases} x_d = x + 2p_1xy + p_2(r^2 + 2x^2) \\ y_d = y + p_1(r^2 + 2y^2) + 2p_2xy \end{cases} \quad (39)$$

where (x_d, y_d) is the point coordinates after the tangential distortion, (x, y) represents the coordinate value of the target in the image's physical coordinate system, r^2 is equivalent to the distance between the coordinate point and the image center, and (p_1, p_2) denotes the parameters of the tangential distortion model.

The complete distortion model of the optical system can be determined by combining the above two types of distortion models, as shown in the following formula [54].

$$\begin{cases} x_d = x(1 + k_1r^2 + k_2r^4 + k_3r^6) + 2p_1xy + p_2(r^2 + 2x^2) \\ y_d = y(1 + k_1r^2 + k_2r^4 + k_3r^6) + p_1(r^2 + 2y^2) + 2p_2xy \end{cases} \quad (40)$$

In this paper, we use the fitting approach to correct the distortion of the coordinate data of a single target to reduce the calculation amount of the distortion correction model. The distortion correction model is shown as follows:

$$\begin{cases} x_r = x_d(1 + k'_1 r_d^2 + k'_2 r_d^4 + k'_3 r_d^6) + 2p'_1 x_d y_d + p'_2 (r_d^2 + 2x_d^2) \\ y_r = y_d(1 + k'_1 r_d^2 + k'_2 r_d^4 + k'_3 r_d^6) + p'_1 (r_d^2 + 2x_d^2) + 2p'_2 x_d y_d \end{cases}, r_d^2 = x_d^2 + y_d^2 \quad (41)$$

where (x_d, y_d) represents the target coordinates of the image's physical coordinate system after distortion correction, (x_r, y_r) denotes the target coordinates of the image's physical coordinate system after distortion correction, and $(k'_1, k'_2, k'_3, p'_1, p'_2)$ indicates the parameters of the inverse distortion model. The parameters are fitted using the measured and actual angle data of the sampling target points.

Finally, the azimuth angle θ and pitch angle φ of the target can be calculated using the following formula.

$$\begin{cases} \theta = \arctan(x_r) \\ \varphi = \arctan(y_r) \end{cases} \quad (42)$$

The calibration experiments of the camera internal and distortion parameters are conducted before the DSP implementation. We calculate the relevant parameters on the PC platform, such as the intrinsic camera matrix and distortion correction parameters. Then, the relevant calculated parameters are solidified within the DSP program to achieve a fast target angle calculation task. The relevant calculated parameters can also be changed by sending instructions from the integrated control system. The calculation of target characteristics data, particularly azimuth and pitch information, can provide comprehensive space target position and grayscale characteristics for space-based surveillance and assist in generating the decision information for spacecraft obstacle avoidance.

4. Hardware Implementation

This paper presents a hardware implementation of the MJDTM model based on an embedded image processing system composed of FPGA and DSP. The FPGA processor, which is suited for parallel computing and has a low computational complexity, has major advantages in terms of large-scale image data processing. This model is suitable for implementing the image filtering algorithm to accomplish rapid target detection. The DSP chip with high-precision digital signal processing capability can complete the algorithm with high computational resource consumption during the tracking stage such as the Kalman filter. To meet the real-time processing requirement of the optical image sequences, we assign processing tasks to the designed multi-core heterogeneous system according to the resource requirements of each processing step. In this section, the constituent modules of the algorithm implementation will be explained in detail.

4.1. Overall Hardware Design

Figure 5 depicts the hardware architecture of the designed space target detection architecture. The onboard space target surveillance system comprises the image acquisition system, the image processing system, the integrated control system, and the external storage. The space-based optical image acquisition system consists of two CMOS sensors with a 1024×1024 pixel resolution and a grayscale value of 12 bits that output image data in the format of LVDS (low-voltage differential signaling) data. The image acquisition system captures the space optical image at a rate of five frames per second. The proposed image processing platform consists of a DSP processor for the target tracking association algorithm and an FPGA chip for image acquisition and target detection. When the image data are received, the processing system will cache the image data and perform the operations of the space target detection and tracking. We will obtain the space target optical image and detection results with the FPGA and DSP processing the image data. A Xilinx Kintex-7 XC7K325TFFG900 FPGA device with 326,080 logic cells, 16,020 KB Block RAM, and 840 DSP slices is used in the image processing system. The TMS320C6678 DSP processor

with eight C66x cores from TI, whose main frequency can reach 1GHz, has been adopted. Additionally, each C66x Core-Pac contains a 512 KB secondary memory (L2), a 32 KB primary program memory (L1P), and a 32 KB data memory (L1D) and can access 4 MB multicore shared memory (MSM). The external storage module contains SDRAM and flash, which are used to cache the image and software program data. The integrated control system sends operation commands to change the working mode of the system and receives the feedback data of the running state through the RS422 interface. It also receives image data and target detection results through the LVDS interface.

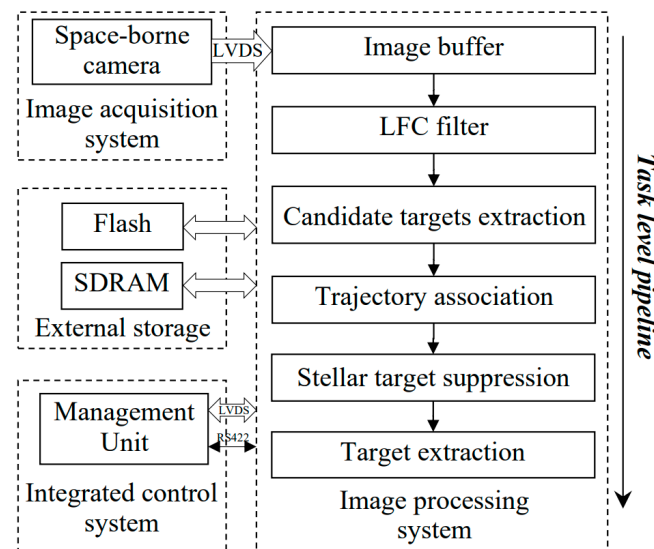


Figure 5. Overview of system conceptual schematic hardware structure.

The overall block structure of the algorithm implementation architecture is shown in Figure 6. The hardware implementation of the detection and tracking algorithm is separated into two parts. The detecting procedure, which requires pixel-by-pixel filtering of the image, is carried out on the FPGA with quick parallel processing capacity. The tracking procedure that executes predictive correlation on the detection target is carried out in the DSP with high-precision digital computing performance. The implementation architecture of the image processing platform comprises the functional modules and processing units. The input to the implementation is the LVDS digital image signals and the RS422 command signals, and the output is the detected target results. In a nutshell, the proposed architecture receives the digital image data from the LVDS interface and executes target detection and tracking operations.

The FPGA implementation comprises the data receive and analysis unit, the command analysis unit, the data cache and output buffer, the system configuration manager, the SRIO communicator, the image cache and slice extract unit, and the target detection module. The data receive and analysis unit first receives the digital image data, converts the serial LVDS data into parallel data, and reads the data packet headers to parse the data according to the communication protocol. Then, the data cache and output buffer store the image data in the external memory. The image cache and slice extract unit stores a whole frame of image data and sends it to the target detection module. The target detection module uses the local feature contrast filter to detect the space and stellar target. The command analysis unit receives various commands such as image processing parameters and configuration management commands from the integrated management unit via the RS422 interface. This unit is also responsible for forwarding instruction data to the DSP processor. Finally, the SRIO communicator packages and send the detected target coordinate and slice data and the auxiliary data to the DSP via the SRIO interface.

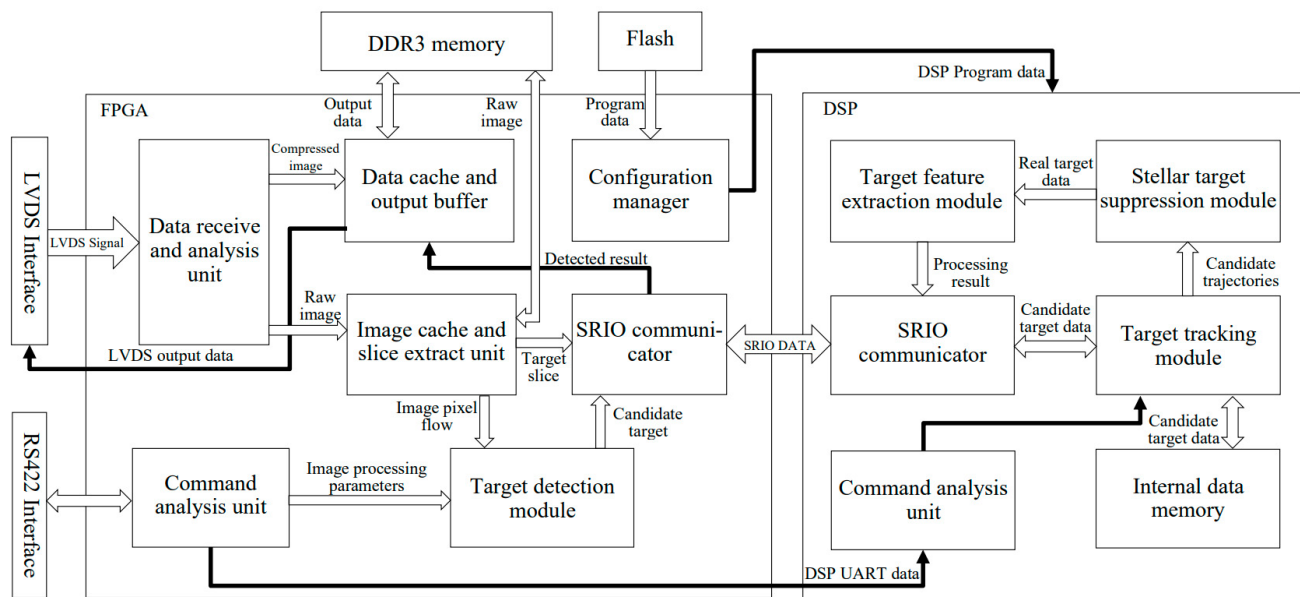


Figure 6. Architecture schematic of system hardware.

The DSP implementation consists of the target tracking module, the SRIO communicator, the stellar target suppression module, the target feature extraction module, and the command analysis unit. The SRIO communicator receives the detected target data and sends them to the target tracking module. The target tracking module adopts the Kalman filter and Hungarian matching algorithm to predict the target state and associated trajectories. The target coordinate data are stored in the internal data memory and read by the target tracking module when updating and associating the target trajectories. The stellar target suppression module uses the real-time satellite attitude data in the auxiliary data package to classify the sidereal and space targets. The feature information of the identified space target are calculated by the target feature extraction module, including the local SNR, the type of targets, and the azimuth and pitch angle. Then, the processing results are packaged and sent to the FPGA chip by the SRIO communicator. The data cache and output buffer in FPGA consolidate the detected results and send them to the integrated control system via the LVDS interface.

4.2. FPGA Implementation

The essential task of the target detection module is to run the LFC algorithm for the optical image sequences. After receiving the image from the space-based image acquisition system through the LVDS interface, FPGA executes the target detection module. The process of the target detection module is illustrated in Figure 7. This module contains five processing stages, including image down-sample, LFC filter, threshold segmentation, connected domain notation, and target centroid extraction. FPGA loads configuration data from Flash by self-starting. After sufficient testing and debugging, software configuration data with default parameters are burned into Flash. During the operation of the system, we can send relevant instructions via UART (universal asynchronous receiver/transmitter) to modify the algorithm parameters.

The hardware architecture of the LFC filter stage is shown in Figure 8. This module will perform the filtering operation in parallel for each pixel during the LFC filter stage. In detail, we split the $n \times m$ filter window from the image pixel stream centered on the point (x, y) . The filter window is scanned across the image in Figure 8a from top to bottom and left to right. To quickly calculate the grayscale average value and standard deviation of the filtering region, the filtering window is divided into nine image blocks.

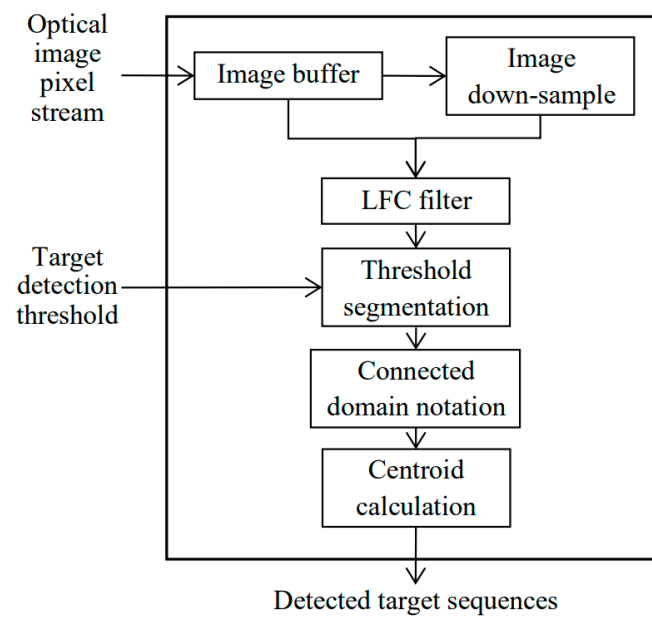


Figure 7. Block diagram of the target detection module.

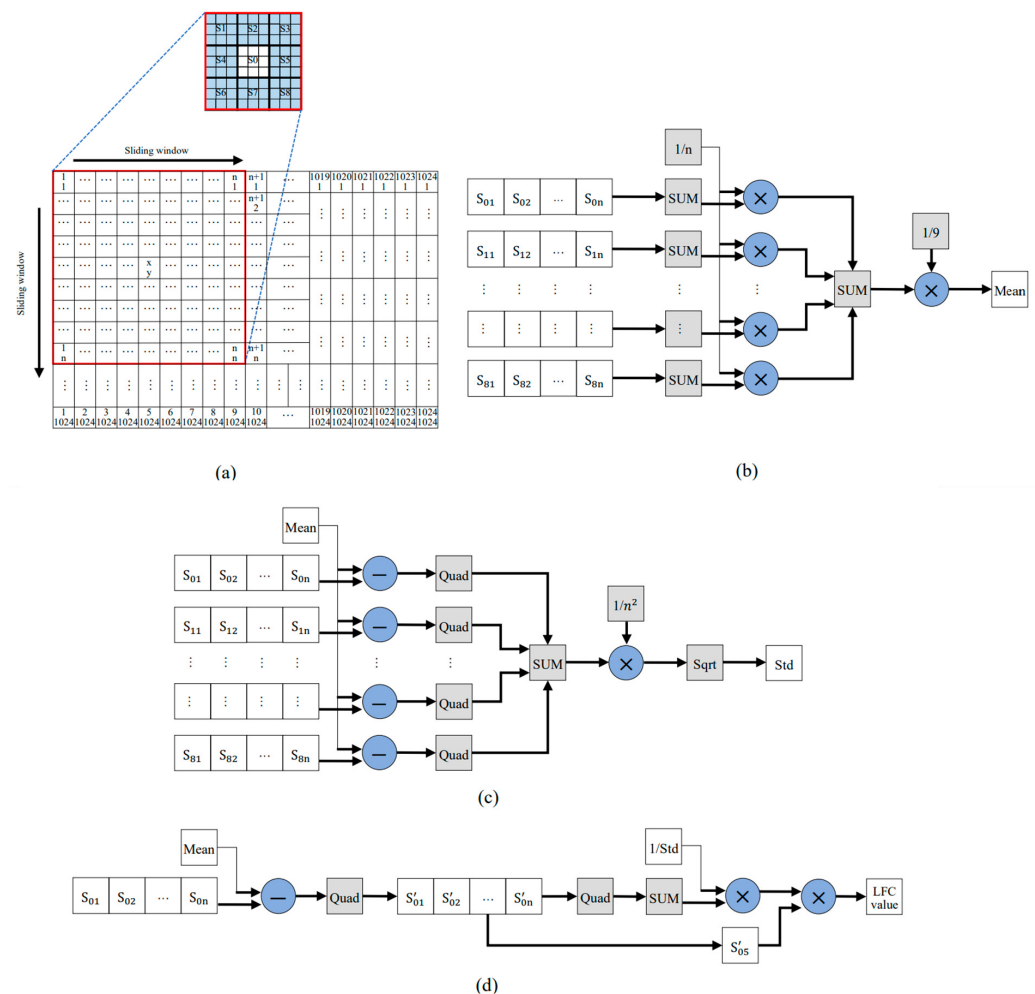


Figure 8. The hardware structure of the LFC filtering: (a) the filter window sliding operation; (b) the hardware structure of the image block gray mean calculation; (c) the hardware structure of the image block gray standard deviation calculation; (d) the hardware structure of the LFC value.

This module runs related calculations for the divided image blocks in parallel, as depicted in Figure 8b,c. Then, a series of calculating operations mentioned in Section 3 are also performed for the filtering window. Finally, the LFC value of the center point (x, y) is calculated, as shown in Figure 8d.

In the hardware implementation, the sliding window size of the LFC filter is set to 9×9 . In practical scenarios, the projection area of the target on the detector may expand, given the target movement and the change of detector lens parameters. As shown in Figure 9, the uneven grayscale distribution and threshold segmentation processing of large targets may lead to the segmentation of one target into several targets, which increases the difficulty of subsequent target tracking. However, by altering the size of the image filter window, the proposed algorithm might be able to adapt to the target size variation.

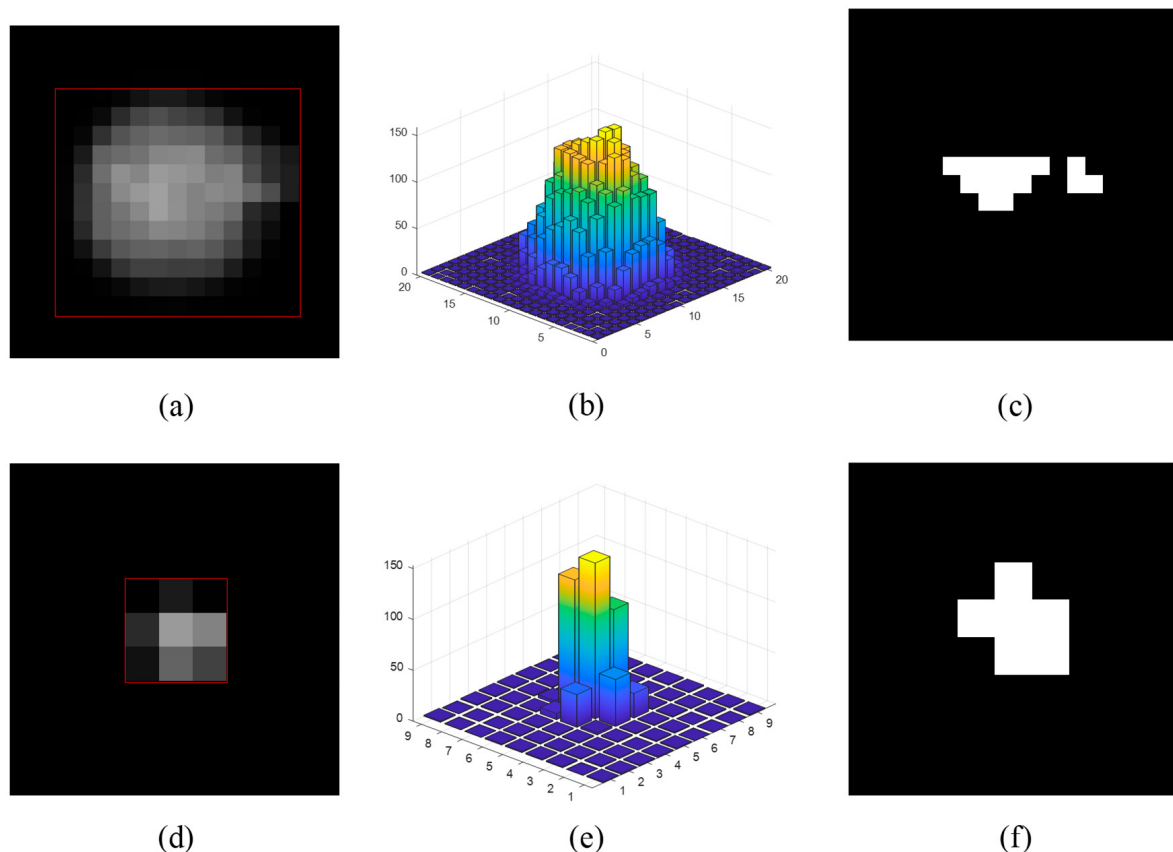


Figure 9. Region of the large target and adjacent neighbor before and after the down-sample operation: (a) the region of the large target and adjacent neighbor; (b) the 3D plot of the large target; (c) the saliency map of the large target; (d) the region of the large target and adjacent neighbor after the down-sample operation; (e) 3D plot of the large target after the down-sample operation; (f) the saliency map of the large target after the down-sample operation.

This will increase the difficulty and cost of the hardware implementation for the detection algorithm. To guarantee the precision and processing speed of target detection, the image downscaling stage executes the down-sample operation on the optical image sequence. We also apply the same scale LFC filter operation to the down-sampled image to extract large-size targets. As shown in Figure 9, the large targets can be accurately detected after the down-sample operation. Since the processing of original and down-sample data is independent, the two LFC filtering procedures for these images are executed concurrently in the FPGA implementation.

After the image LFC filter, the adaptive threshold segmentation is conducted on the local feature contrast image to extract the potential target. The selection of the adaptive segmentation threshold is discussed in Section 5.3. Then, this module applied the scanning

line technology to label the connected domain of the target. The distance-weighted centroid method is also adopted to calculate the target centroid coordinate data. Finally, we can obtain the detection target coordinate data based on the original and down-sampled images. The target centroid coordinates and the slice data segmented according to the coordinate data are packaged and sent to the DSP processor.

4.3. DSP Implementation

The flowchart of the software in the DSP system is shown in Figure 10. The DSP system mainly performs image processing tasks such as the multi-frame association of detected targets, matching update of candidate trajectories, stellar target suppression, and target feature calculation.

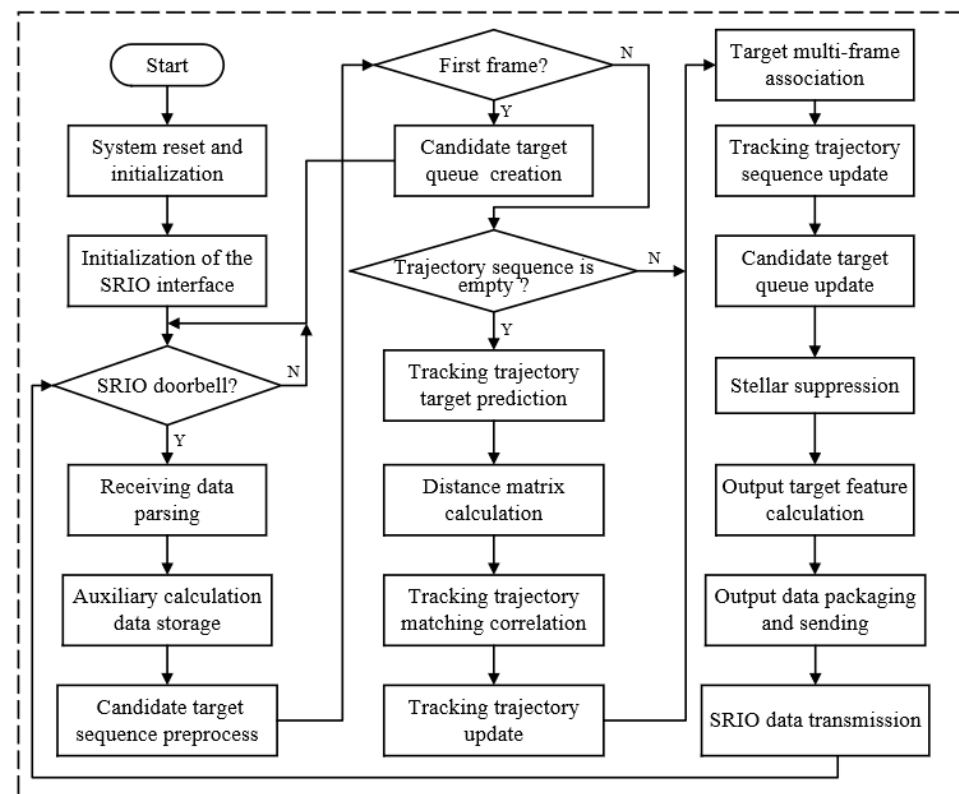


Figure 10. DSP software flow block diagram.

We employ the main Core 0 for software program development and use the kernel's local L2 SDRAM and MSM SDRAM as data storage because the DSP software processing tasks in this system are relatively simple and the amount of data processed is small. The peripheral interfaces of other cores are turned off to reduce system power consumption. The DSP is started by SPI boot mode, and the configuration interface is connected to the FPGA. The FPGA realizes the option of the SPI configuration mode by controlling the high and low levels of the configuration interface. After the DSP is powered up in the prescribed order and the configuration mode is set up, it reads the program data through the SPI interface and loads it into the program storage space in the core and starts to run the program. After sufficient testing and debugging, the DSP software program is written into FLASH, and it comes with a set of default algorithm parameters. During the process of software operation, we can send the relevant execution through the UART interface to change and optimize the algorithm parameters to achieve the optimal processing effect.

When the DSP system completes the program loading and system initialization, it enters the idle state and waits for the SRIO doorbell interrupt. After the FPGA completes the candidate target extraction of a frame image, the detection result data package is written to the memory of the DSP through the SRIO interface, and the doorbell signal is

delivered to the DSP after the data transmission is completed by the FPGA. The DSP starts the processing of the candidate target data for the current frame under the trigger of the doorbell interrupts. Firstly, the software parses the candidate target data and auxiliary calculation data in the data packet according to the relevant protocol. The software updates the candidate target data and auxiliary calculation data in the packet to the corresponding storage array. Then, the software calls different functions according to the processing stage to associate the target. In the initial tracking stage, the software performs multi-frame correlation to confirm the real target for multiple consecutive frames of candidate targets. In the middle tracking stage, the software creates the tracking trajectory for the real targets and predicts their motion state in subsequent frames. In the subsequent stable tracking stage, multi-frame association and trajectory association operations are alternated between the candidate target queue and the trajectory sequence, and for the trajectory sequence, the software discriminates the type of the target and suppresses the stellar target to confirm the real space target. Finally, after continuous multi-frame stable tracking correlation and recognition, the feature data of the real target are calculated by the software, packaged, and sent to FPGA. If there is no target in the current frame that is judged to be the true target, the relevant data are not sent. The software enters the idle state after completing the processing of the current frame.

4.3.1. Target Tracking Module

In the DSP implementation, the target tracking module predicts the candidate target state prediction and associates the tracking trajectories with the detected target sequence. Figure 11 illustrates the block diagram of this module. Firstly, this module executes the multi-frame association before the tracking trajectory update to suppress noise points. It calculates the distance between targets in the current frame detected target sequence and the former target sequence in the candidate target queue. When the separation between the targets is below a predetermined threshold, the target is considered a potential target, and the coordinates of the latest frame are updated to the candidate target queue. The target data in the detected target sequence are directly updated to the candidate target queue when the system first receives the target data. This module generates the associated tracking trajectory for subsequent target state prediction and trajectory association when the target in the candidate target queue has more occurrences than a set threshold. In detail, we use the Vision Library, a collection of optimized computer vision algorithm libraries developed by Texas Instruments for digital media processors. It contains the Application Programming Interface for Kalman filter algorithms, which can quickly perform complex function operations in hundreds of machine cycles. The `VLIB_kalmanFilter_2 × 4` is the structural variable type used for the Kalman filter calculation with two-dimensional observation and four-dimensional state vectors. This module creates the corresponding `VLIB_kalmanFilter_2 × 4` structural variable for each tracking trajectory, which is convenient for calling the predict and correct API function to achieve the prediction and state update of the tracking trajectory.

Moreover, this module employs the Hungarian matching algorithm to associate the tracking trajectory queue and the detected target sequence. The distance between the tracking trajectory predicted sequence and the detected target sequence is calculated and integrated into a distance matrix used as the weight of trajectory matching. The Hungarian matching algorithm uses a recursive method to find the path with the maximum expectation value to match the trajectories with the target sequence. The detected target sequence of each frame image is preferentially associated with the tracking trajectories in this module. Moreover, in the multi-frame association stage, the targets in the detected target sequence that successfully matched the tracking trajectory are not associated with the candidate target queue.

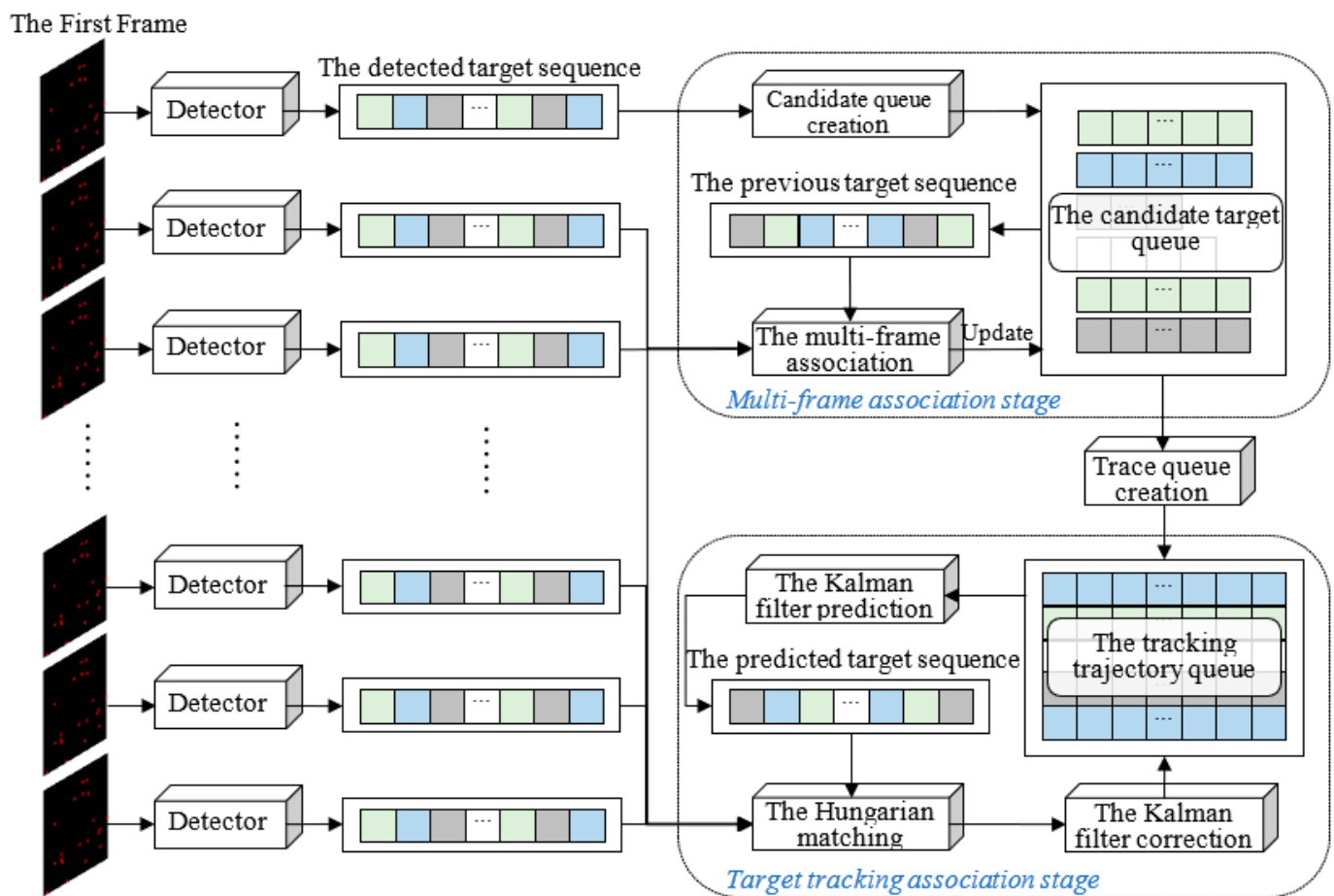


Figure 11. Block diagram of the target tracking module.

4.3.2. Stellar Target Suppression Module

After the latest tracking trajectories of the candidate targets are associated and updated, the potential target type needs to be confirmed, as mentioned in Section 2. The stellar target suppression module, which predicts the target position using the real-time satellite attitude data to classify the stellar targets and space targets, is shown in Figure 12. In the implementation, in order to eliminate the influence of platform jitter on the detection results, the historical coordinate data in the candidate tracking trajectory are utilized to identify the target type. As shown in the diagram, this module computes the target coordinate data relative to the Earth by using the image plane coordinate data and the satellite platform attitude data. Each frame's satellite platform attitude data are included in the auxiliary package transmitted together with the detected target package sent by FPGA. Moreover, the predicted target coordinate data at time n in the image coordinate system could be calculated with the attitude data of the satellite platform at time n . The mean value of the difference between the target actual coordinate and the predicted coordinate calculated is used to judge whether the target is a real space target or not. In detail, this module begins to classify the type of candidate target for each tracking trajectory when its length surpasses a specific value. The threshold value that determines the target type also can be changed by sending instructions from the integrated management system.

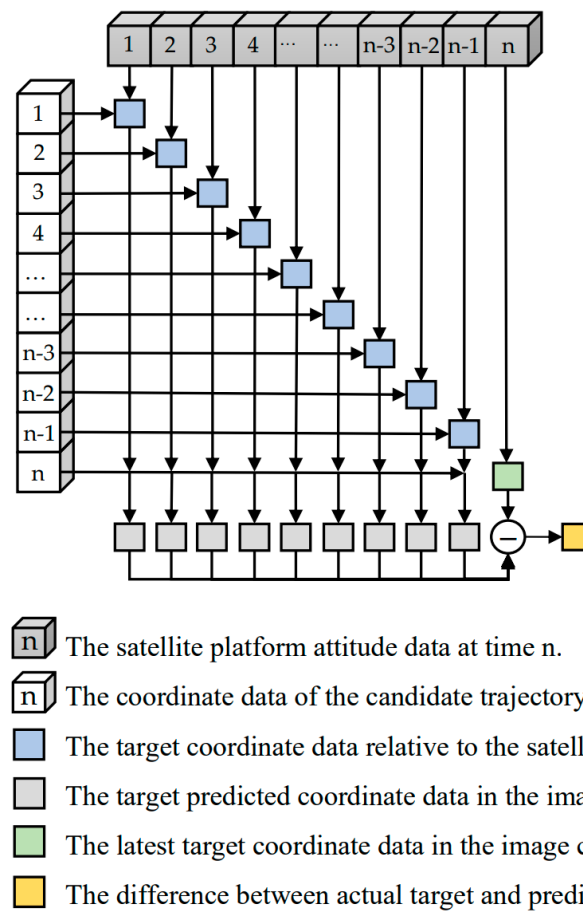


Figure 12. Stellar target suppression module.

5. Experiment

The hardware architecture described in Section 4 was implemented on a dedicated embedded image processing platform. The pictorial diagram of the platform is shown in Figure 13. The architecture was implemented using Verilog and C. To better evaluate the performance of the target detection algorithm, the simulations were conducted using Matlab2018b in the Win10 system, using an i7-10750H CPU with 2.6 GHz and 16 GB of main memory. The remainder of the section is organized as follows. First, the optical image dataset used in the experiment is described. Then, we assess the proposed architecture regarding the target detection rate, the efficiency of the stellar target suppression algorithm, and the target angle calculation accuracy. Finally, we present an evaluation of processing efficiency for the hardware implementation.

5.1. Experimental Dataset

To validate the detection performance of the proposed algorithm, we use the simulated and real image sequences. The simulated image dataset comprises image sequences of the wide and narrow field of view, of which the image size is 1024×1024 pixels and the grayscale value is 12 bits. The simulated images whose background is the deep space background contain stellar targets and moving space targets that simulate the space optical image in space-based scenarios. The motion attitude data of the satellite platform are synchronously generated with the image sequences.



Figure 13. Picture of the embedded image processing platform.

Furthermore, the real image data are captured by two cameras of the space-based optical image acquisition system introduced in Section 4 in the ground simulation scenario. The wide and narrow field-of-view camera covers the range of $90^\circ \times 90^\circ$ and $8^\circ \times 8^\circ$ field of view, respectively. The image size and pixel gray level are the same as the simulated image. The real image data are taken on a clear cloudless night, and the camera is placed vertically on the ground, capturing the sidereal points and the unmanned aerial vehicle and civil aviation aircraft targets that simulate the moving space target under the night sky background. The image dataset introduced in detail in Table 1 contains two groups of simulated image sequences and two groups of real image sequences.

Table 1. Details of the space target image sequences.

Sequence	Frame	Field View	Background Details	Target Details
Seq.1	300	Wide field	Simulated deep space background; random noise	Simulated target; 3×3
Seq.2	300	Narrow field	Simulated deep space background; random noise	Simulated target; 3×3
Seq.3	300	Wide field	Real background; sky	Civil aviation aircraft; 3×3
Seq.4	300	Narrow field	Real background; cloud and sky	Unmanned aerial vehicle; 12×12

5.2. Target Detection and Tracking Experiment

In this section, we evaluate the performance of the proposed space target detection and tracking algorithm, including the accuracy of target detection, the efficiency of target tracking, and the accuracy of the stellar target suppression algorithm, by using the simulated and real image sequence.

First, we evaluate the proposed target detection algorithm using the four groups of image datasets mentioned in the previous section. To quantitatively assess the performance of the detection algorithm, we use two evaluation criteria: the detection probability and the false alarm rate. The detection probability P_d and false alarm rate P_f are defined as

$$P_d = \frac{N_d}{N_t} \quad (43)$$

$$P_f = \frac{N_f}{N_p} \quad (44)$$

where N_d represents the number of the true detected targets, N_t denotes the number of real targets, N_f indicates the number of false targets, and N_p denotes the total number of pixels in the processed images. Meanwhile, we present the receiver operating characteristic (ROC) curves and calculate the area under the curve (AUC) to intuitively appraise the algorithm's performance. The ROC curve could illustrate the corresponding relationship between the detection probability and the false alarm rate, which is one of the quantitative methods to evaluate detection efficiency. The closer the curve is to the upper left corner, the better the algorithm performs.

Simultaneously, the proposed algorithm is compared with state-of-the-art small target detection algorithms, including the multiscale tri-layer local contrast measure (TLLCM) and the weighted strengthened local contrast measure (WSLCM) [55]. Figure 14 shows the ROC curves of these three algorithms for four groups of space optical image sequences. As shown in Figure 14a, the proposed method obtained a P_d exceeding 95% when P_f reached 10^{-4} . As shown in Figure 14c,d, the proposed and comparative algorithm could reach 95% P_d when $P_f < 10^{-6}$. As shown in Figure 14b, both the proposed algorithm and the comparison algorithm reach 95% P_d when the P_f does not exceed 10^{-4} . Since the target is weak, the performance of the proposed algorithm is slightly inferior to that of WSLCM. The area under the curve (AUC) can further be used to evaluate the performance of the target detection method and provide a more comprehensive comparison. The results of AUC are shown in the figure. The AUC values of the proposed method were 0.9759, 0.9819, 0.9843, and 0.9919. In contrast to Seq.2, the proposed algorithm obtains the maximum AUC value on the other three datasets. The experimental results show that compared with other algorithms, the energy accumulation step in the proposed algorithm enhances the target, achieving a better target detection performance and ensuring the algorithm implementation detection efficiency.

The adaptive segmentation threshold is a significant factor that decides the accuracy of target detection. Figure 15 demonstrates the detection rate and false alarm rate performance of the algorithm on four sequences under different values of the parameter k_1 . Figure 15a indicates that when the parameter k_1 is between 30 and 50, the target detection rate exceeds 95%, which is sufficient for practical applications. When the parameter k_1 is less than 35, the P_f is less than 10^{-4} , which satisfies the majority of application requirements, as shown in Figure 15b. In conclusion, we recommend that the parameter k_1 should have a range of values between 35 and 50, so that the detection performance can attain over 95% P_d and $P_f < 10^{-4}$.

After this, we performed a series of experiments using the simulated wide-field and narrow-field image sequences. There are twenty to thirty sidereal points and one space target setting in the two simulated groups of image sequences. The wide- and narrow-field image detection results are shown in Figure 16a,b. It can be seen from the figure that the detection algorithm can accurately detect space targets and stellar points. For the target sequence of each image frame, we created the corresponding tracking trajectory to track the target. When the number of consecutive occurrences of the target is greater than seven, the satellite attitude data generated via simulation are used to classify the space target and the sidereal points of the track trajectories. The results of trajectory tracking and stellar target suppression can be seen in Figure 16b,c, and the efficiency of target tracking after the stellar target suppression is shown in Figure 16e,f. The results show that the algorithm can successfully identify space targets.

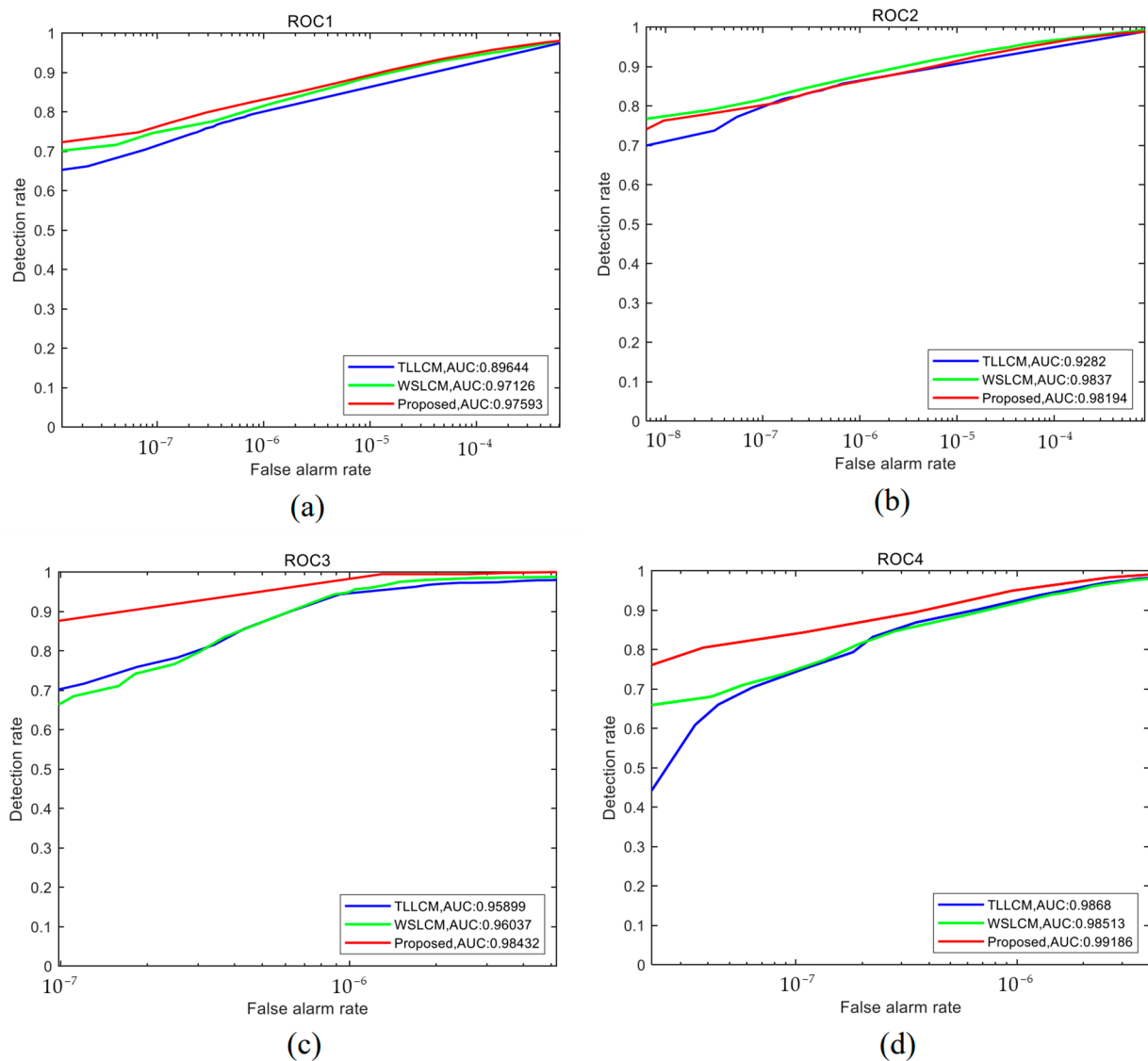


Figure 14. ROC curves of four groups of the image sequences. (a) ROC curve of Seq.1; (b) ROC curve of Seq.2; (c) ROC curve of Seq.3; (d) ROC curve of Seq.4.

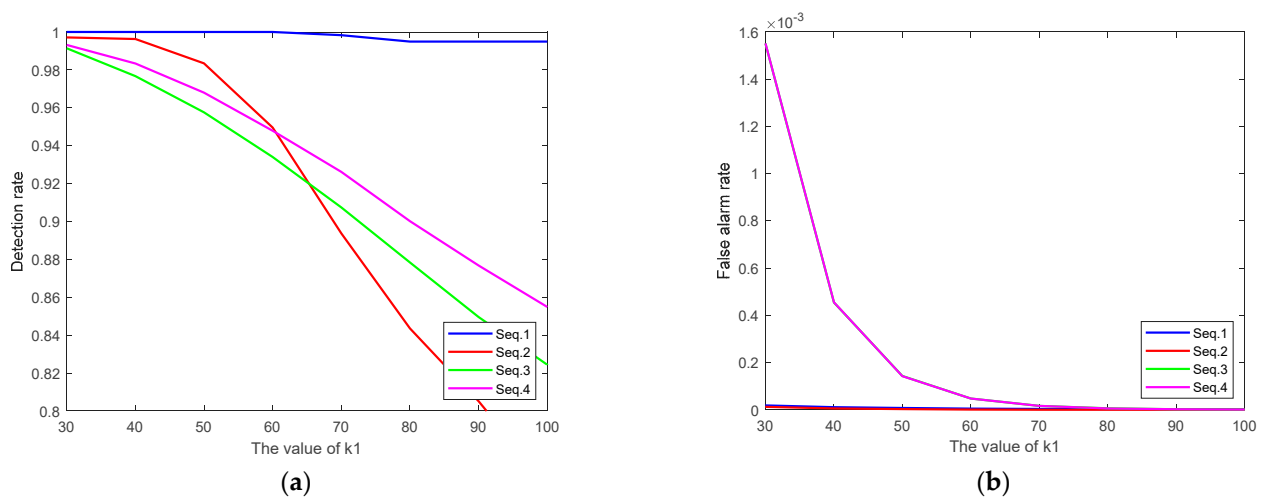


Figure 15. Influence of k_1 . (a) Relationship between k_1 and P_d ; (b) relationship between k_1 and P_f .

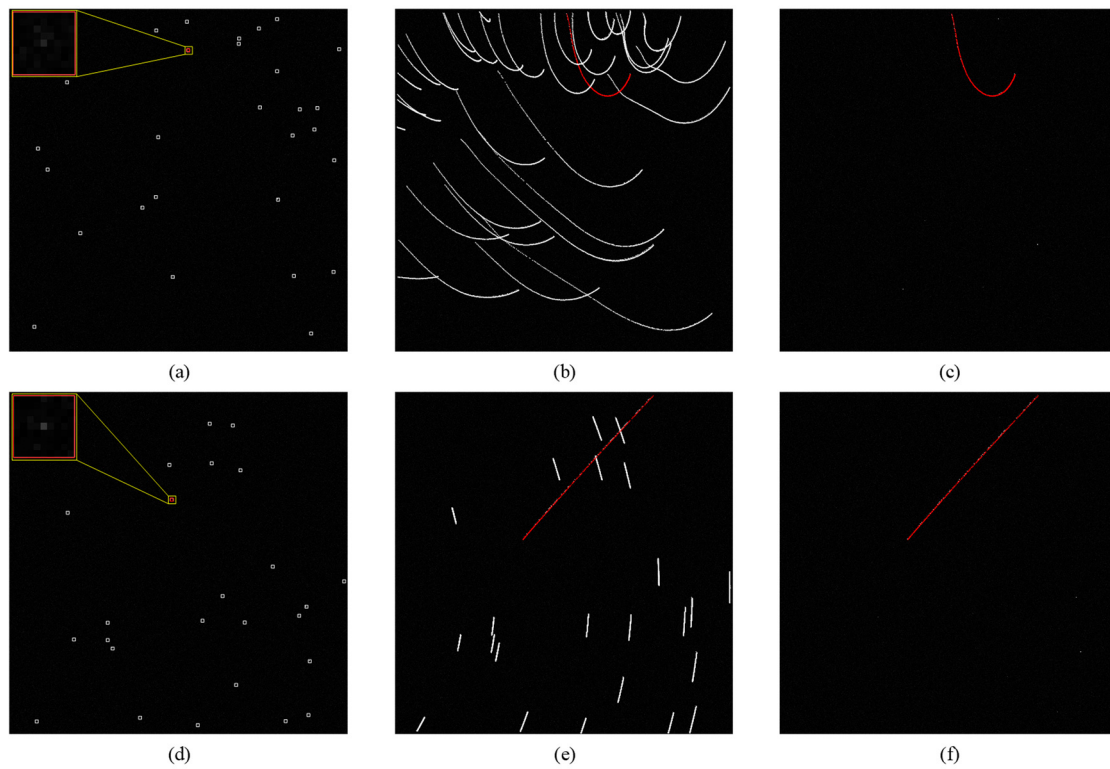


Figure 16. Simulation image detection and tracking results. (a) Target detection results of Seq.1 (left upper corner is space target area slice); (b) trajectory tracking results of Seq.1 (red is space target trajectory, white is star trajectory); (c) tracking trajectory result of Seq.1 after the stellar target suppression; (d) target detection results of Seq.2 (upper left corner is target area slice); (e) trajectory tracking results of Seq.2; (f) tracking trajectory result of Seq.2 after the stellar target suppression.

In addition, to illustrate the effect of the stellar target suppression algorithm, we define the detection probability and false alarm rate based on the whole image sequence to quantitatively evaluate the stellar target suppression algorithm. The detection probability P_t and false alarm rate F_a are defined as

$$P_t = \frac{N_{rd}}{N_{rt}} \quad (45)$$

$$F_a = \frac{N_{rf}}{N_{pf}} \quad (46)$$

where N_{rd} represents the number of detected frames of real space targets, N_{rt} denotes the frame number of real space targets, N_{rf} represents the number of detected frames of false space targets, and N_{pf} denotes the total number of frames in the sequence of images. Table 2 shows the calculation results of this algorithm's detection rate and false alarm rate on two sets of simulated image sequences. The target tracking stage monitors potential targets in multi-frame optical image sequences and suppresses the stellar targets using platform attitude data and historical coordinate data that can reflect target motion differences. The experimental results show that this stage achieves high-precision detection of real space targets.

Table 2. Detection probability and false alarm rate of the stellar target suppression algorithm on different image sequences.

Sequence	P_t	F_a
Seq.1	91.72%	1.33%
Seq.2	97.25%	0%
Seq.3	80.9%	0%
Seq.4	95.67%	0%

We also conducted relative experiments for the real image set, including space target detection, tracking, and stellar target suppression. First, we perform the target detection experiment on the image sequence. Besides the target detection of the original image, we also detect the target in the down-sample real image sequence, which is used to realize the detection of large-size targets. As shown in Figure 17e, when the target is close to the detector, the number of pixels occupied by the target on the detector plane will increase, leading to the algorithm marking one target as two targets. The down-sample processing step of the image reduces the area of the big target to ensure the accuracy of the target detection. Specifically, this step reduces the image size from 1024×1024 pixels to 256×256 pixels by sampling the original image every four pixels. The target can be down-sampled from 12×12 to 3×3 by quadrupling this down-sample rate. The hardware implementation of the detection algorithm's filter window can accurately detect targets with a diameter of less than 3, and after down-sampling, the diameter of targets with a diameter of 4 to 12 is reduced to between 3 and 1, allowing our hardware implementation to also accurately detect the target. A target with an area greater than 12×12 is not considered in this paper. The detection results of the original image and down-sampled image are shown in Figure 17. We obtain two sets of target sequences after the space detection of the original image and the down-sampled image. A fusion operation is executed to merge the two target sequences.

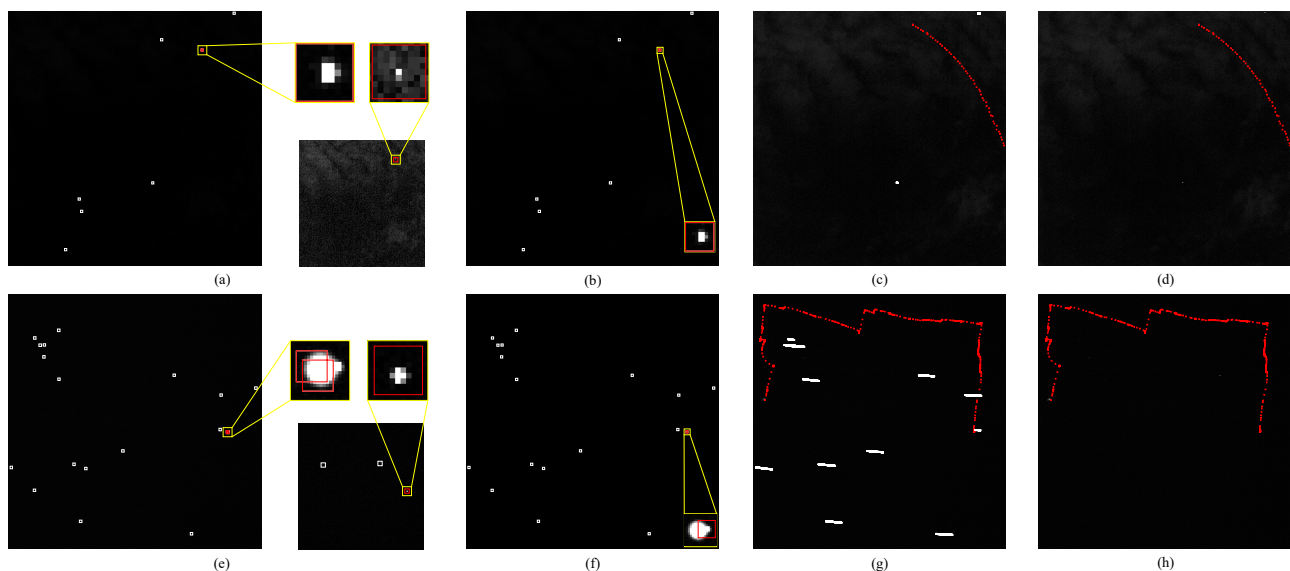


Figure 17. Real image detection and tracking results: (a) the detection results of the original image and the down-sampled image of Seq.3; (b) the fusion results of the dual-size target detection results of Seq.3; (c) trajectory tracking results of Seq.3; (d) tracking trajectory results of Seq.3 after the stellar target suppression; (e) the detection results of the original image and the down-sampled image of Seq.4; (f) the fusion results of the dual-size target detection results of Seq.4; (g) trajectory tracking results of Seq.4; (h) tracking trajectory results of Seq.4 after the stellar target suppression.

The detection results are shown in Figure 17b,f, and the large target is marked as one target after fusion. We create the corresponding tracking trajectories for the target sequence

in the target tracking experiments. When the length of the tracking trajectory is greater than 7, we use the simulated satellite attitude data to classify the space targets and sidereal points in the trajectory sequence. The target tracking results are shown in Figure 17c,g, in which the white trajectories belong to sidereal points and the red one belongs to the simulated space target (UAV and aircraft) trajectory. The results of target tracking after the stellar target suppression are shown in Figure 17d,h. It can be seen from the figure that the stellar points and space targets are precisely distinguished using the difference in motion between them. Table 2 also shows the stellar target suppression results of the proposed algorithm on the real image sequences.

Finally, the target angle information of the space target is calculated. The accuracy of the target angle measurement method will also be calculated in the following experiment. For the wide-field camera with a $90^\circ \times 90^\circ$ field of view, camera distortion correction is required to ensure the accuracy of the angle calculation before performing the target angle calculation. The specific correction scheme is as follows. The camera is mounted on a two-dimensional rotating platform, and a point target is set in front of the rotating platform to simulate a space target. The correction method is shown in Figure 18a. First, we calculate the mounting matrix of the camera, in which the rotating platform rotates to the corresponding angle according to the set angle sequence. The camera acquires 25 target images of the angle near the center point to generate a set of image sequences. We extract the coordinates of 25 target points and fit the camera mounting matrix using these points.

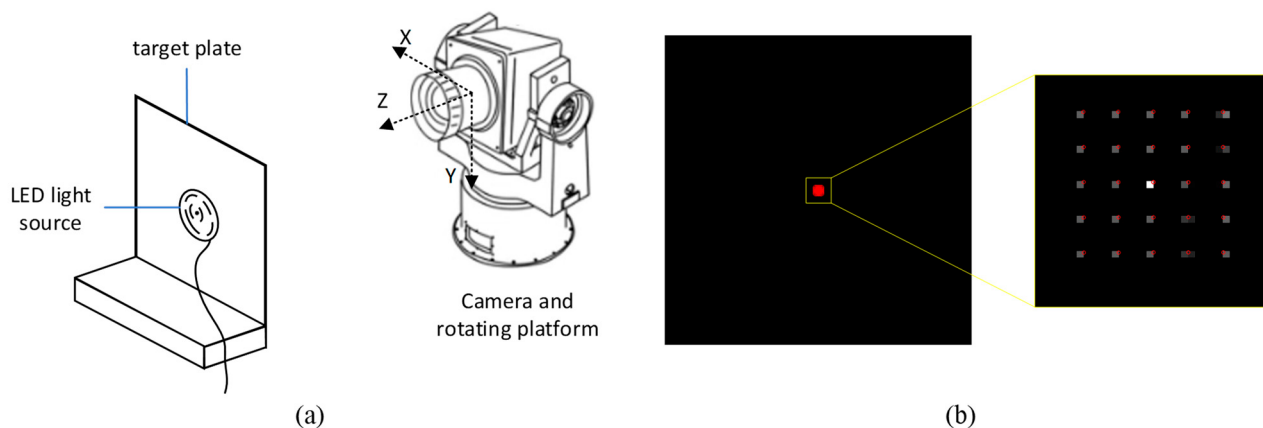


Figure 18. Wide-field camera distortion correction method. (a) Distortion correction method; (b) collated sampled point grid image.

Then, the rotating platform also rotates to the corresponding angle according to the set angle sequence, and the camera acquires 225 setting target points images to generate a set of image sequences. We also detect and extract the coordinates of target points for the acquired image sequence, and the collated sampled point grid image is shown in Figure 19a. Due to the barrel distortion of the wide-field camera, the target's actual imaging position is often not in the ideal projection model coordinates. Consequently, we use the target ideal coordinate sequence and the actual imaging coordinate sequence to generate the inverse distortion model. The grid of sampling points corrected by the inverse distortion model is shown in Figure 19b. To verify the accuracy of this inverse distortion model, we take images of 25 test target points. The target points are detected using the proposed algorithm, and the azimuth and pitch angle are also calculated using the inverse distortion model. The measured angle results of the test target point before and after the distortion correction are shown in Figure 19c,d. The red star mark in the figure is the angle value calculated using the imaging position of the target image plane, and the blue is the actual angle value of the target. As shown in the figure, the azimuth and pitch angle of the target are accurately calculated after the aberration correction. The average angle measurement error is 0.1334, and the maximum value of the side angle error is 0.2419. The angle measurement accuracy

can reach 99.73%. The formula of the side angle error ϕ and the angle measurement accuracy ε are as follows.

$$\phi = \sqrt{(\theta_i - \theta_r)^2 + (\varphi_i - \varphi_r)^2} \quad (47)$$

$$\varepsilon = 1 - \phi/FOV \quad (48)$$

where (θ_i, φ_i) denotes the actual value of the azimuth and pitch angle of the target point, (θ_r, φ_r) represents the corrected value of the calculated azimuth and pitch angle of the target point after the distortion correction operation, FOV denotes the camera field of view, and ε is the angle measurement error.

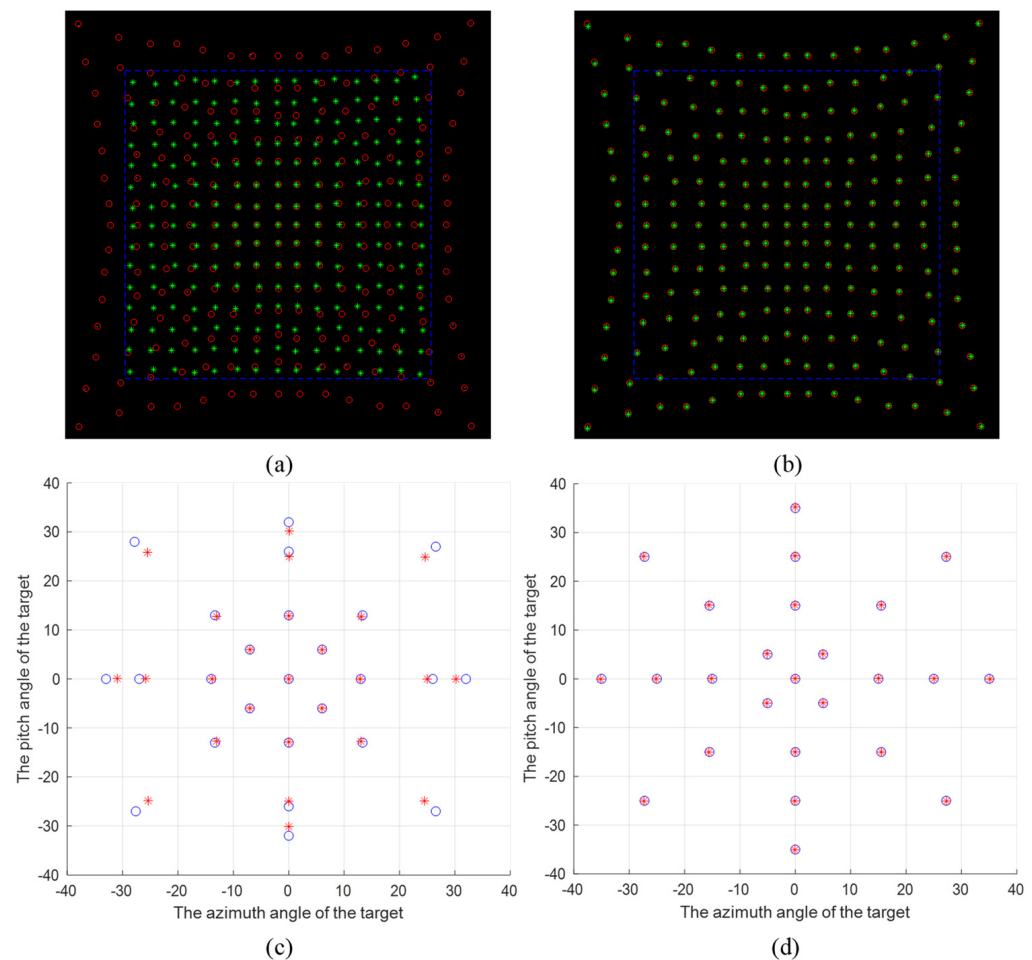


Figure 19. Wide-field camera distortion correction results. (a) Distribution of scanned target imaging position and theoretical position before aberration correction; (b) distribution of scanned target correction position and theoretical position after aberration correction; (c) calculated and actual angles of test target points before distortion correction; (d) calculated and actual angles of test target points after distortion correction.

5.3. Hardware System Computational Performance Analysis

In this section, we conduct several experiments to evaluate the proposed implementation's computational performance and operational efficiency. As described in Section 4, the proposed target detection algorithm is implemented on the Xilinx Kintex-7 FPGA with the specific hardware resource consumption rates shown in Table 3. We utilized these resources in the FPGA implementation to optimize the design. Table 4 reports the processing time and power consumption obtained for the hardware implementation of the proposed algorithm on the considered FPGA and DSP architecture. The single-frame image processing time

measured in the FPGA is only 22.064 ms. Therefore, the designed space target detection architecture can realize a processing speed of 45 frames per second. In experimental tests, the power consumption of DSP and FPGA is 7.02 W and 6.168 W, respectively, and the overall power consumption can be controlled within 15 W, which satisfies the space-based platform application requirements.

Table 3. Summary of resource utilization for the FPGA implementation of the proposed target detection algorithm.

Component	Number of LUTs	Number of FFs	Number of BRAMs	Number of DSPs	Number of BUFs
Units	4.9999	6.1274	233	53	22
Percentage	24.53%	15.03%	52.36%	6.31%	68.75%

Table 4. Processing time measured for space detection and tracking method in the hardware system.

Hardware Platform	Processing Time	Clock Period	Hardware Operation Frequency	Hardware Power Consumption
FPGA	22.046 ms	1,102,300	50 MHz	7.02 W
DSP	0.5946 ms	595,460	1000 MHz	6.168 W

Finally, we use the simulated image sequences Seq.1 and Seq.2 to evaluate the performance of the hardware implementation for target detection and tracking. The experiment results of the target detection and stellar target suppression are shown in Table 5. Due to the complexity of implementation, we have not drawn the ROC curve on the FPGA implementation program. The threshold of the adaptive segmentation algorithm is set between 10 and 20 for testing. Finally, a detection rate of 96.36% can be achieved when the false alarm rate is less than 0.4% during the detection stage of the FPGA implementation. The stellar suppression algorithm of the target tracking algorithm module implemented by DSP can accomplish an average detection rate of 87.8% for actual space targets in sequence images. The DSP implementation is inferior to that of the PC platform. The primary reason is that the multi-frame correlation module will confirm the target in the first few frames of the target. The target tracking stage of the DSP implementation will execute the stellar suppression algorithm when the tracking trajectory length exceeds a certain threshold. Consequently, the detection rate will be low in the early stages of tracking and will increase as the trajectory length increases.

Table 5. The result of the hardware implementation for the proposed algorithm.

Hardware Platform	Sequence	Seq.1	Seq.2
FPGA implementation	P_d	97.37%	96.36%
	P_f	0.0332%	0.0335%
DSP implementation	P_t	87.27%	88.33%
	F_a	0%	0%

6. Discussion

In this paper, a space target detection and tracking model is presented with its hardware implementation scheme. A dim small space target detection approach is proposed in the target detection stage, which improves the local contrast method. According to the experimental results on the real and simulated image datasets, as illustrated in Figure 14, its detection performance is stronger than that of TLLCM and WSLCM. In detail, our algorithm can obtain 95% P_d when $P_f < 10^{-4}$ on all sequences. The target detection method is implemented on an FPGA, and Tables 3–5 reveal the resource and time consumption and the experimental results. The target detection software is self-started by the FPGA, and the segmentation parameter will remain constant for a while. The detection performance is significantly impacted by the selection of the adaptive segmentation threshold. On the

one hand, we define the default threshold using the results of the PC platform. On the other hand, the detection rate and false alarm rate information are calculated with the output of the detection results from the FPGA, and the best detection effect can be obtained by appropriately modifying the segmentation threshold parameters in accordance with the detection effect. Furthermore, the hardware development could complete the target detection of a single frame image in 22 ms thanks to the parallel processing capabilities of FPGA, which guarantees real-time performance of image processing.

The Kalman filter algorithm and the Hungarian matching algorithm collaborate at the target tracking stage to stabilize the tracking target. The experimental results of the model and the detection effect are displayed in Figure 17 and Table 2. The satellite's attitude data are easily obtained on the space-based platform. Given that attitude data undoubtedly contain errors, we begin the stellar suppression algorithm once the tracking trajectory reaches a particular threshold to avoid the effects of incorrect attitude data on the star suppression effect. We utilize simulated attitude data for the experiment on the PC platform, and the threshold is set at 8 since the simulated data error is minor. On the one hand, calculation errors due to platform differences may have an impact on the detection effect. On the other hand, employing more frames of historical target coordinate data for statistics can eliminate the calculation error caused by attitude data error and ensure the target's detection rate in the actual space-based scene. In order to provide accurate target angle information, we also present a distortion correction scheme for the large field-of-view-optical lens. As illustrated in Figure 19, the distortion correction scheme could reduce the angle measurement error to less than 0.3%.

In conclusion, the experimental results validate the efficacy and viability of the model and hardware architecture and confirm that the processing system is capable of real-time space target detection and tracking, thereby meeting the requirements of the space-based platform application.

7. Conclusions

In this paper, a multi-stage joint detection and tracking model is developed to solve the problem of space target detection and tracking in the deep space background and a hardware implementation of this model for space-based surveillance applications is provided. The experiments conducted with the simulated and real image sequences demonstrate that the proposed implementation can lead to improvements in detection accuracy while maintaining real-time processing speed. However, the proposed model may not have a good detection performance for low SNR targets and depends on real-time satellite attitude data. In future work, we will improve the method to address these shortcomings and apply it in other complex scenarios.

Author Contributions: Conceptualization, P.R.; methodology, Y.S. and X.C.; software, Y.S. and G.L.; validation, Y.S., X.C. and C.C.; formal analysis, Y.S.; investigation, Y.S. and G.L.; resources, Y.S. and C.C.; data curation, Y.S. and X.C.; writing—original draft preparation, Y.S.; writing—review and editing, X.C.; visualization, G.L.; supervision, P.R.; project administration, X.C.; funding acquisition, P.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, M.; Yan, C.; Hu, C.; Liu, C.; Xu, L. Space Target Detection in Complicated Situations for Wide-Field Surveillance. *IEEE Access* **2019**, *7*, 123658–123670. [[CrossRef](#)]
2. Wang, X.; Chen, Y. Application and Development of Multi-source Information Fusion in Space Situational Awareness. *Spacecr. Recovery Remote Sens.* **2021**, *42*, 11–20. [[CrossRef](#)]
3. Chen, L.P.; Zhou, F.Q.; Ye, T. Design and Implementation of Space Target Detection Algorithm. *Appl. Mech. Mater.* **2015**, *738–739*, 319–322. [[CrossRef](#)]

4. Barniv, Y. Dynamic programming solution for detecting dim moving targets. *IEEE Trans. Aerosp. Electron. Syst.* **1985**, *AES-21*, 144–156. [[CrossRef](#)]
5. Barniv, Y.; Kella, O. Dynamic programming solution for detecting dim moving targets part II: Analysis. *IEEE Trans. Aerosp. Electron. Syst.* **1987**, *AES-23*, 776–788. [[CrossRef](#)]
6. Doucet, A.; Gordon, N.J.; Krishnamurthy, V. Particle filters for state estimation of jump Markov linear systems. *IEEE Trans. Signal Process.* **2001**, *49*, 613–624. [[CrossRef](#)]
7. Salmond, D.; Birch, H. A particle filter for track-before-detect. In Proceedings of the 2001 American Control Conference (Cat. No. 01CH37148), Arlington, VA, USA, 25–27 June 2001; pp. 3755–3760.
8. Reed, I.S.; Gagliardi, R.M.; Shao, H. Application of three-dimensional filtering to moving target detection. *IEEE Trans. Aerosp. Electron. Syst.* **1983**, *AES-19*, 898–905. [[CrossRef](#)]
9. Zhang, C.; Chen, B.; Zhou, X. Small target trace acquisition algorithm for sequence star images with moving background. *Opt. Precision Eng.* **2008**, *16*, 524–530.
10. Cheng, J.; Zhang, W.; Cong, M.; Pan, H. Research of detecting algorithm for space object based on star map recognition. *Opt. Tech.* **2010**, *36*, 439–444.
11. Zhang, J.; Ren, J.-C.; Cheng, S.-C. Space target detection in star image based on motion information. In *International Symposium on Photoelectronic Detection and Imaging 2013: Optical Storage and Display Technology*; SPIE: Bellingham, WA, USA, 2013; pp. 35–44.
12. Xi, X.-L.; Yu, Y.; Zhou, X.-D.; Zhang, J. Algorithm based on star map matching for star images registration. In *International Symposium on Photoelectronic Detection and Imaging 2011: Space Exploration Technologies and Applications*; SPIE: Bellingham, WA, USA, 2011; p. 81961N.
13. Boccignone, G.; Chianese, A.; Picariello, A. Small target detection using wavelets. In Proceedings of the Fourteenth International Conference on Pattern Recognition (Cat. No. 98EX170), Brisbane, QLD, Australia, 20 August 1998; pp. 1776–1778.
14. Jiang, P.; Liu, C.; Yang, W.; Kang, Z.; Li, Z. Automatic Space Debris Extraction Channel Based on Large Field of view Photoelectric Detection System. *Publ. Astron. Soc. Pac.* **2022**, *134*, 024503. [[CrossRef](#)]
15. Chen, C.P.; Li, H.; Wei, Y.; Xia, T.; Tang, Y.Y. A local contrast method for small infrared target detection. *IEEE Trans. Geosci. Remote Sens.* **2013**, *52*, 574–581. [[CrossRef](#)]
16. Chen, L.; Rao, P.; Chen, X. Infrared dim target detection method based on local feature contrast and energy concentration degree. *Optik* **2021**, *248*, 167651. [[CrossRef](#)]
17. Sun, R.-Y.; Zhan, J.-W.; Zhao, C.-Y.; Zhang, X.-X. Algorithms and applications for detecting faint space debris in GEO. *Acta Astronaut.* **2015**, *110*, 9–17. [[CrossRef](#)]
18. Deshpande, S.D.; Er, M.H.; Venkateswarlu, R.; Chan, P. Max-mean and max-median filters for detection of small targets. In *Signal and Data Processing of Small Targets 1999*; SPIE: Bellingham, WA, USA, 1999; pp. 74–83.
19. Bai, X.; Zhou, F. Infrared small target enhancement and detection based on modified top-hat transformations. *Comput. Electr. Eng.* **2010**, *36*, 1193–1201. [[CrossRef](#)]
20. Lv, P.; Sun, S.; Lin, C.; Liu, G. A method for weak target detection based on human visual contrast mechanism. *IEEE Geosci. Remote Sens. Lett.* **2018**, *16*, 261–265. [[CrossRef](#)]
21. Han, J.; Moradi, S.; Faramarzi, I.; Liu, C.; Zhang, H.; Zhao, Q. A local contrast method for infrared small-target detection utilizing a tri-layer window. *IEEE Geosci. Remote Sens. Lett.* **2019**, *17*, 1822–1826. [[CrossRef](#)]
22. Lan, Y.; Peng, B.; Wu, X.; Teng, F. Infrared dim and small targets detection via self-attention mechanism and pipeline correlator. *Digit. Signal Process.* **2022**, *130*, 103733. [[CrossRef](#)]
23. Shi, F.; Qiu, F.; Li, X.; Tang, Y.; Zhong, R.; Yang, C. A method to detect and track moving airplanes from a satellite video. *Remote Sens.* **2020**, *12*, 2390. [[CrossRef](#)]
24. Fujita, K.; Hanada, T.; Kitazawa, Y.; Kawabe, A. A debris image tracking using optical flow algorithm. *Adv. Space Res.* **2012**, *49*, 1007–1018. [[CrossRef](#)]
25. Kalman, R.E. A New Approach to Linear Filtering and Prediction Problems. *J. Basic Eng.* **1960**, *82*, 35–45. [[CrossRef](#)]
26. La Scala, B.F.; Bitmead, R.R. Design of an extended Kalman filter frequency tracker. *IEEE Trans. Signal Process.* **1996**, *44*, 739–742. [[CrossRef](#)]
27. Huang, T.; Xiong, Y.; Li, Z.; Zhou, Y.; Li, Y. Space Target Tracking by Variance Detection. *J. Comput.* **2014**, *9*, 2107–2115. [[CrossRef](#)]
28. Hao, L.; Mao, Y.; Yu, Y.; Tang, Z. A method of GEO targets recognition in wide-field opto-electronic telescope observation. *Opto-Electron. Eng.* **2017**, *44*, 418–426.
29. Lin, J.; Ping, X.; Ma, D. Small target detection method in drift-scanning image based on DBT. *Infrared Laser Eng.* **2013**, *42*, 3440–3446.
30. Mehta, D.S.; Chen, S.; Low, K.-S. A rotation-invariant additive vector sequence based star pattern recognition. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *55*, 689–705. [[CrossRef](#)]
31. Yang, L.; Niu, Y.; Zhang, Y.; Lü, J.; Li, J.; Niu, H.; Liu, W.; Zhang, Y. Research on Detection and Recognition of Space Targets Based on Satellite Photoelectric Imaging System. *Laser Optoelectron. Prog.* **2014**, *51*, 121102. [[CrossRef](#)]
32. Bo, M. Research on Aerial Infrared Small Target Detection and Hardware Acceleration. Master's Thesis, Beijing University of Technology, Beijing, China, 2016.
33. Zhang, Q. Design and Implementation of Spaceborne Infrared Small Target Detection System Based on FPGA. Master's Thesis, Huazhong University of Science and Technology, Wuhan, China, 2019.

34. Liu, W. Object tracking under complicated background based on DSP+FPGA platform. *Chin. J. Liq. Cryst. Disp.* **2014**, *29*, 1151–1155.
35. Seznec, M.; Gac, N.; Orioux, F.; Naik, A.S. Real-time optical flow processing on embedded GPU: An hardware-aware algorithm to implementation strategy. *J. Real-Time Image Process.* **2022**, *19*, 317–329. [\[CrossRef\]](#)
36. Diprima, F.; Santoni, F.; Piergentili, F.; Fortunato, V.; Abbattista, C.; Amoroso, L. Efficient and automatic image reduction framework for space debris detection based on GPU technology. *Acta Astronaut.* **2018**, *145*, 332–341. [\[CrossRef\]](#)
37. Tian, H.; Guo, S.; Zhao, P.; Gong, M.; Shen, C. Design and Implementation of a Real-Time Multi-Beam Sonar System Based on FPGA and DSP. *Sensors* **2021**, *21*, 1425. [\[CrossRef\]](#)
38. Sun, Q.; Niu, Z.D.; Yao, C. Implementation of Real-time Detection Algorithm for Space Debris Based on Multi-core DSP. *J. Phys. Conf. Ser.* **2019**, *1335*, 012003. [\[CrossRef\]](#)
39. Gyaneshwar, D.; Nidamanuri, R.R. A real-time FPGA accelerated stream processing for hyperspectral image classification. *Geocarto Int.* **2022**, *37*, 52–69. [\[CrossRef\]](#)
40. Han, K.; Pei, H.; Huang, Z.; Huang, T.; Qin, S. Non-cooperative Space Target High-Speed Tracking Measuring Method Based on FPGA. In Proceedings of the 2022 7th International Conference on Image, Vision and Computing (ICIVC), Xi'an, China, 26–28 July 2022; pp. 222–231.
41. Yang, B.; Yang, M.; Plaza, A.; Gao, L.; Zhang, B. Dual-mode FPGA implementation of target and anomaly detection algorithms for real-time hyperspectral imaging. *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.* **2015**, *8*, 2950–2961. [\[CrossRef\]](#)
42. Xu, Y.; Zhang, J. Real-time detection algorithm for small space targets based on max-median filter. *J. Inf. Comput. Sci.* **2014**, *11*, 1047–1055. [\[CrossRef\]](#)
43. Han, L.; Tan, C.; Liu, Y.; Song, R. Research on the On-orbit Real-time Space Target Detection Algorithm. *Spacecr. Recovery Remote Sens.* **2021**, *42*, 122–131.
44. Choi, E.-J.; Yoon, J.-C.; Lee, B.-S.; Park, S.-Y.; Choi, K.-H. Onboard orbit determination using GPS observations based on the unscented Kalman filter. *Adv. Space Res.* **2010**, *46*, 1440–1450. [\[CrossRef\]](#)
45. Babu, P.; Parthasarathy, E. FPGA implementation of multi-dimensional Kalman filter for object tracking and motion detection. *Eng. Sci. Technol. Int. J.* **2022**, *33*, 101084. [\[CrossRef\]](#)
46. Zhang, X.; Xiang, J.; Zhang, Y. Space Object Detection in Video Satellite Images Using Motion Information. *Int. J. Aerosp. Eng.* **2017**, *2017*, 1024529. [\[CrossRef\]](#)
47. Li, Q.; Li, R.; Ji, K.; Dai, W. Kalman filter and its application. In Proceedings of the 2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS), Tianjin, China, 1–3 November 2015; pp. 74–77.
48. Kuhn, H.W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [\[CrossRef\]](#)
49. Zhu, H.; Zhou, M. Efficient role transfer based on Kuhn–Munkres algorithm. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2011**, *42*, 491–496. [\[CrossRef\]](#)
50. Mirzaeinia, A.; Hassanalian, M. Minimum-cost drone–nest matching through the kuhn–munkres algorithm in smart cities: Energy management and efficiency enhancement. *Aerospace* **2019**, *6*, 125. [\[CrossRef\]](#)
51. Luetkeke, F.; Zhang, X.; Franke, J. Implementation of the hungarian method for object tracking on a camera monitored transportation system. In Proceedings of the ROBOTIK 2012: 7th German Conference on Robotics, Munich Germany, 21–22 May 2012; pp. 1–6.
52. Kuipers, J.B. *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace, and Virtual Reality*; Princeton University Press: Princeton, NJ, USA, 1999.
53. Tang, Z.; Von Gioi, R.G.; Monasse, P.; Morel, J.-M. A precision analysis of camera distortion models. *IEEE Trans. Image Process.* **2017**, *26*, 2694–2704. [\[CrossRef\]](#)
54. Weng, J.; Cohen, P.; Herniou, M. Camera calibration with distortion models and accuracy evaluation. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 965–980. [\[CrossRef\]](#)
55. Han, J.; Moradi, S.; Faramarzi, I.; Zhang, H.; Zhao, Q.; Zhang, X.; Li, N. Infrared small target detection based on the weighted strengthened local contrast measure. *IEEE Geosci. Remote Sens. Lett.* **2020**, *18*, 1670–1674. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.