Graph-Based Horizon Line Detection for UAV Navigation

Yong Xu, Hongtao Yan, Yue Ma, and Pengyu Guo

***Abstract*—Perceiving the horizon line is a critical alternative for unmanned aerial vehicle (UAV) autonomous navigation, especially in the presence of noise-induced drift, unavailability of satellite navigation, and multipath errors. However, it's quite tough to detect the horizon line, due to the remotely sensed big data, the dynamic changes in flight, and the serious consequences of failure. To address these problems, we propose a graph-based horizon line detection technique that is composed of graph-based image segmentation, connected domain cascade filtering, horizon line extraction, and UAV attitude estimation. We improve the graph-based image segmentation algorithm so that the segmentation results are more conducive to horizon line detection. We then determine the sky-component by cascade filtering and extract the horizon line based on the boundaries of the sky-component. Furthermore, we directly compute the roll and pitch according to the extracted horizon line and eliminate the ambiguity of the angles. To validate our approach qualitatively and quantitatively, we designed a fixed-wing UAV system. We then validated our algorithm through extensive flights under various conditions and compared the estimated rolls and pitches to the IMU ones. Statistical results show that the proposed technique provides unbiased attitude angles with error variance of about 2^o, which verify the validity and robustness of our method. For engineering, our program runs at approximately 60 fps on the fly after optimizing.**

***Index Terms*—Connected domain cascade filtering, graph-based image segmentation, horizon line detection, unmanned aerial vehicle (UAV) navigation.**

I. Introduction

THE APPLICATIONS of unmanned aerial vehicle (UAV) cover all walks of life, including agricultural plant protection, power line inspection, forest fire monitoring, security surveillance, transportation, and military competition [1]. Thus, researches about UAVs have attracted extensive attention in recent years, especially the task of UAV navigation [2]. However, due to the big data captured by various sensors, the highly dynamic changes in flight, and the serious consequences of collision and failure, it's challenging to realize a fully autonomous UAV navigation system, particularly for vision-based UAV autonomous navigation [3].

For UAV autonomous navigation, it is essential to measure, stabilize, and maintain the attitude of the aircraft. Traditionally, the attitudes of UAVs are measured by the IMUs which can capture the angular rate and acceleration of the aircraft. The flight controller then exploits the sensed values to stabilize attitudes by correcting the unwanted rotations in roll, pitch, and yaw [4]. This is a standard stabilization scheme of many automatic crewless systems. However, it severely suffers from drift especially for long-term flights or in the case of big maneuver [5]. The primary reason for drifting is that the values sensed by gyro are contaminated by noise and it only captures the angular rate about IMU reference, not the orientation about absolute reference. To obtain the attitude angles, the sensed angular velocities must be integrated that may cause substantial noise-induced drift, particularly for the low-cost MEMS. The weaknesses mentioned above might be addressed by adopting different sensors which can directly capture the information about absolute orientation [5] or exploiting signals of the navigation satellites to correct the attitude of UAV such as GPS, GNSS, and BeiDou.

Compared to the traditional attitude stabilization methods, horizon-based UAV stabilization techniques can avoid integration in the time domain, and do not depend on the signals of the navigation satellites, which results in no drifts. In the flight of aircraft, the horizon is approximately a straight line in the captured images, which can directly provide the absolute roll and pitch orientations. By detecting the horizon line on the fly, it can assist the obstacle avoidance, attitude estimation, path planning, and autonomous navigation tasks of the human-crewed and crewless aircraft, which could improve the reliability of the aircrafts. Besides, the horizon line separates the sky component from the ground part, which allows the image algorithm to process the sky and ground components separately and helps to improve the performance of the algorithm. However, with the rapid development of aerospace science and technology, the resolution and acquisition frequency of remote sensing image systems have significantly improved, accelerating the entry of remote sensing into the era of big data. To effectively extract and exploit the horizon line in the remotely sensed image data, we develop a fast and robust graph-based horizon line extraction technique to promote the fixed-wing UAV navigation tasks.

Manuscript received June 26, 2021; revised August 9, 2021, October 14, 2021, and November 1, 2021; accepted November 5, 2021. Date of publication November 9, 2021; date of current version November 30, 2021. This work was supported in part by Science and Technology Innovation 2030 Major Project under Grant 2020AAA0104801 and in part by the Natural Science Foundation of China under Grant 61903364. (*Corresponding author: Pengyu Guo.*)

Yong Xu, Hongtao Yan, and Yue Ma are with the Aerospace Technology Institute of CARDC, Mianyang 621000, China (e-mail: xuyong22@cardc.cn; yanhongtao@cardc.cn; mayue@cardc.cn).

Pengyu Guo is with the National Innovation Institute of Defense Technology, Academy of Military, Beijing 100071, China (e-mail: pengyu.guo@nudt.edu.cn).

Digital Object Identifier 10.1109/JSTARS.2021.3126586

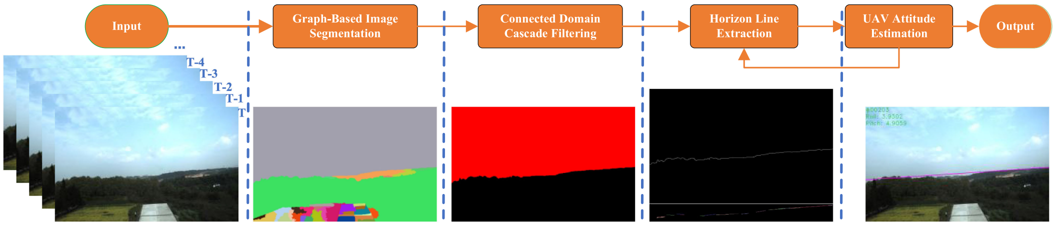


Fig. 1. Inputs, processing pipeline, and outputs of the GHLD method.

The main purpose of this article is to study the horizon line detection method in the course of UAV flight. Meanwhile, the roll and pitch angles of the UAV can be directly estimated based on the detected horizon line. We then apply these two angles to assist UAV autonomous navigation. However, due to the rapid changes in the UAV's position, attitude, terrain, and lighting conditions during flight, this task is extremely difficult, which requires the detection algorithm to adapt to this highly dynamic environment. Besides, considering the high cost of collision or failure, higher requirements are placed on the accuracy, real-time, and reliability of detection results.

Traditional horizon line detection usually uses edge detection or cluster segmentation, which do not adapt to dynamic flight, especially for fixed-wing UAV. While the deep neural network (DNN) based methods achieve good performance but significantly increase execution times. In view of the above problems, we propose a graph-based horizon line detection (GHLD) method and exploit the estimated roll and pitch for UAV navigation. Fig. 1 shows the pipeline of the algorithm. By analyzing the overall characteristics of the UAV aerial image, we divide the horizon detection task into four subtasks: graph-based Image segmentation; connected domain cascade filtering; horizon line extraction; and UAV attitude estimation. Furthermore, we design a UAV system for verification. Qualitative and quantitative results illustrate the effectiveness and robustness of the proposed algorithm. The main contributions of our approach are as follows.

1) We propose a robust and effective GHLD technique, which produces unbiased attitude angles with an error variance of about 2^{o} and can operate at about 60 fps with limited airborne resources. Results could be applied to assist the task of fixed-wing UAV navigation.
2) We improve the graph-based segmentation technology so that the segmentation results are more conducive to horizon line detection.
3) We exploit connected domain cascade filtering to accelerate the algorithm, meanwhile employ LSD [29], clustering, and length-weighted average technology to enhance the robustness of horizon line detection.

The article is organized as follows. Section II reviews the related work about horizon line detection. Section III describes the methodology of our approach. In Section IV, a fixed-wing UAV system is designed, and extensive tests are performed; meanwhile, several engineering tricks are discussed. Section V analyzes the limitations of the proposed GHLD and suggests some future works. In Section VI, brief concluding remarks are presented.

II. Related Work

Vision-based UAV navigation has attracted considerable attention in recent years. Compared with the traditional UAV autonomous navigation techniques, the role of vision-based UAV navigation is highlighted in the case that traditional schemes collapse due to the presence of noise-induced drift, unavailability of satellite navigation, multipath errors, or man-made electromagnetic interference [6]. In these situations, an autonomous and robust estimation of vision-based six-dimensional (6-D) positions and attitudes could be adopted as a considerable alternative. There are two topics related to vision-based navigation, one is simultaneous localization and mapping, and the another is horizon line-based navigation. In the last two decades, intense researches about SLAMs are conducted which produces different kinds of SLAM systems, including mono-camera SLAM, stereo-cameras SLAM, and visual-inertial SALM [7]. Meanwhile, the accuracy, robustness, and survivability of these systems have also been improved. Some remarkable works are presented in [7]–[10]. To the best of our knowledge, all the SLAM algorithms suffer from drift during long-duration flights, due to the integration of IMUs and the accumulation of position uncertainty by the structure from motion. However, the horizon line can directly provide the absolute roll and pitch orientations without any drifts which not only helps to mitigate the cumulative errors of SLAMs, but also promotes the stabilization of aircraft. Thus, lots of approaches have been proposed to tackle the problem of horizon line detection. According to the employed features and methodology, the ways to accomplish horizon line detection are divided into three categories: lower-level feature approaches; segmentation approaches; and deep learning approaches.

A. *Lower-Level Feature Approaches*

The first category of methods attempts to directly extract the horizon line basing on low-level features, such as edge, color, pixel intensity, corner, and gradients, which results in low computation, but less robustness. Making use of the image edges, Stavros *et al.* [11] detected the horizon line basing on color, texture, and SIFT features. They exploited a Canny edge detector and a Hough line detector to detect the line followed by a particle swarm optimization step. Ahmad *et al.* [12] proposed to label the pixels of the horizon line and train an SVM classifier, depending on the edges and local features. Then, they tried to discriminate the horizon line in an edge image. Besides, Ahmad *et al.* [13] detailly compared the existing edge-based and edge-less methods and proposed a fusion method by taking both the advantages of the two, which performed an edge detection on a query image and applied a boosting strategy if a pixel lies on the edge.

To solve the horizon line detection problem under maritime environment, Prasad *et al.* [14] applied a Hough transformation on the edge map and the intensity gradient to a detect the line. To improve the accuracy of horizon detection in marine images, they proposed a new image feature named multiscale cross-modal linear feature. In [15], Liang *et al.* proposed a technique to extract the horizon semantic boundaries under the maritime environment. They exploited the weighted textures to computed the probabilities of each region and determined the sea-sky region. Then, they extracted a number of candidate line segments by utilizing a canny edge detector and Hough line detector. Finally, they voted for each candidate and the one with the maximum score was adopted as the final result line. In [16], Zou *et al.* attempted to extract the horizon line via computing the gradients at each column of the image. They treated the pixel with max gradient value as potential points. To eliminate the effects of clouds and sea waves, they adopted a total variation algorithm to denoise the maritime images. For actual application, they exploited the resulting horizon line to calculate the roll of the tug.

B. Segmentation Approaches

The second category of methods focuses on segmenting the image into different components and then extract the boundary between the sky component and the ground part. Exploiting the machine learning methods, Fefilatyev *et al.* [17] classified the image into two components, sky and ground. They investigated the horizon line detection problem by applying the Naive Bayes classifier, support vector machine (SVM), and decision tree; train the corresponding classifiers, and employed them to discriminate each pixel. Timothy *et al.* [18] exploited the SVMs technique to classify the sky region and nonsky areas. They classified each pixel in the YCrCb color space after denoised by the Gaussian filter. Then, they extracted the horizon line basing on a Hough line detector nearby the border and detected the target in the sky by taking the detected line as a prior. For validation, they steered a small crewless drone to fly toward one of the targets. Boroujeni *et al.* [19] introduced a technique to detect the horizon line based on the presence of the unique light field. They segmented the image by K-means clustering based on pixel intensities. Sun and Fu [20] applied a coarse-fine-stitched strategy to detect the horizon line. At the coarse step, they detected a series of line segments by applying the gradient features, which formed the candidate pool. At the fine step, the candidate line segments were refined by combining the morphology features such as length and direction with color features to eliminate the effects of marine waves, illumination change, and occlusion. At the stitching step, they stitched the remained line segments by exploiting RANSAC to remove the false positive ones.

Besides, Zhan *et al.* [21] aimed to dynamically recognize the water regions in the image and detect the safe region with the help of a lidar sensor. First, they applied a modified graph-based segmentation technique to preprocess the input image. The segmentation results were used to train a CNN online, which was employed to classify the image. Then, a conditional random field was adopted to refine the results. Zhan *et al.* [22] proposed to classify the marine image by a VGG network followed by CRF. To further improve the performance of the proposed approach, they clustered image pixels into a number of superpixels and generated a superpixel-level label map which refined the segmentation results. Results show that the presented method adapted to new environment.

C. Deep Learning Approaches

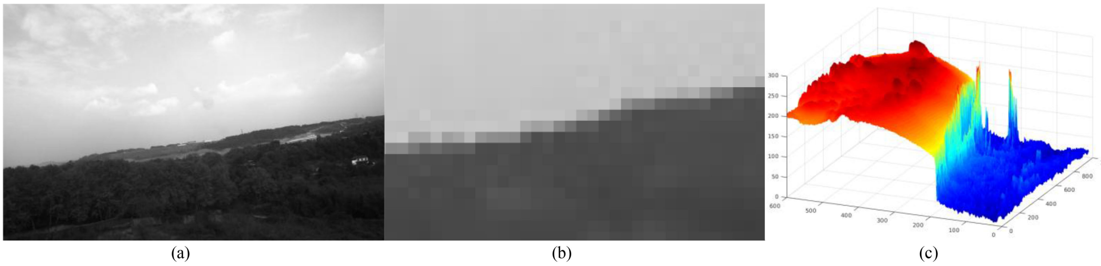
Deep learning approaches try to take the advantage of DNNs to discriminate the horizon line. Basing on the DNN technology, Workman *et al.* [23] investigated methods for directly detecting the horizon line using convolutional neural networks, including both classification and regression formulations, and achieved good performance, but time-consuming. Porzi *et al.* [24] proposed a deconvolutional architecture to detect the horizon line, which was composed of convolutional blocks, deconvolutional blocks, and predictors blocks. The approach was based on two-steps, computing the boundary by learning-based contour detection algorithm and finding the most likely horizon line via dynamic programming. Besides, Zhai *et al.* [25] innovatively utilized a deep CNN to extract the global image context; they then detected the vanishing points and the zenith vanishing point based on the extracted context information. The proposed method could detect the horizon line in challenging nonlinear scene, such as man-made environment. Jeong *et al.* [26] proposed a scene paring network based on deep CNN, which was capable of classifying the image into semantic categories, and then the horizon line could be iteratively retrieved by least-squares accompanying with a median filter.

Moreover, to eliminate the effects of atmospheric and illumination, Carrio *et al.* [27] attempted to detect the horizon line in thermal images and achieved favorable results. They proposed two methods for this task. For the first method, they estimated the potential points in subimages and then fitted the horizon line based on these points. Besides, they tried to estimate the attitude angles by solving a regression learning problem based on raw image pixels.

Our work falls into the second category and is closely related to the work in [21]. However, our method aims to detect the horizon line in the sky-ground scenes and executes fast enough to fulfill the dynamic requirements of the fixed-wing UAV navigation. In this article, we propose an effective GHLD method. We modify the graph-based image segmentation algorithm [28] so that the segmentation results are more conducive to horizon line detection. We then determine the sky-component by cascade filtering and extract the horizon line based on the boundary between the sky and ground image areas. The methodology is detailly presented in the succeeding section.

III. Methodology

In this section, we first analyze the overall properties of UAV aerial images and then discuss the four modules of the proposed method in detail.



(a) (b) (c)

Fig. 2. Analysis of UAV aerial image properties. (a) Gray image. (b) Enlarged view nearby the sky-ground boundary. (c) Intensities of the gray image.

A. Properties of UAV Aerial Images

As mentioned earlier, UAVs are employed in a diversity of applications, such as agricultural protection, traffic surveillance, power line inspection, forest fire monitoring, dangerous area observation, transportation, 3-D mapping, and military reconnaissance. For these applications, the altitude of the UAV ranges from several meters to hundreds of meters. As shown in Fig. 2, the horizon line in the aerial image is a critical feature, which is beneficial for vision-based autonomous navigation, especially for stabilizing the attitude of the UAV.

Fig. 2(a) shows the grayscale image captured on the fly, Fig. 2(b) illustrates the enlarged view nearby the sky-ground boundary in Fig 2(a), and Fig. 2(c) presents the intensity of each pixel in three dimensions. Through analysis, we obtain the following four properties of the aerial images.

1) The boundary between the sky component and the ground part in an aerial image appears as a straight line, that is the horizon line.
2) Normally, the mean intensity of the sky component is greater than that of the ground part.
3) The pixel intensities nearby the boundary between the sky and ground image areas change slowly, not stepwise.
4) The sky-component occupies a relatively large part of the entire image and is at least connected to one of the image boundaries.

Based on the above properties, we first adopt the graph-based segmentation approach to segment the image into multiple connected domains. Then, we calculate the characteristics of each connected domain and determine the sky-component through cascade filtering. Finally, we extract the horizon line basing on the binary image nearby the sky-ground boundary and estimate the UAV roll and pitch angles.

B. Graph-Based Image Segmentation

The segmentation algorithm in [28] is a graph-based greedy clustering algorithm, which is simple to implement and runs fast. The main purpose of [28] is to segment the similar areas into connected components. The segmentation results are neither too fine nor too rough, which are often used as a preprocessing technology for many image processing algorithms. In this article, we utilize this method for the horizon line detection task. Unlike [28], our goal is to distinguish the sky area from the ground one, without caring about the details in the image. To make the segmentation results more conducive to horizon line detection, we modify the algorithm as follows.

1) We remove the preprocessing part that smooths the image, to fully retain the gradient information around the edges.
2) We add the distance of elements that is used to calculate the edge-weight, to alleviate the effects of the slow changes of the image intensities nearby the sky-ground boundary.
3) We modify the segmentation criterion by utilizing the local dissimilarity and global dissimilarity between the minimum spanning trees (MSTs) of the graph.
4) To remove the details in the segmentation results, we design the size threshold of post-processing to merge the small component with the large connected one.

In the proposed GHLD approach, the undirected graph is composed of vertices, edges, and weights, where each vertex corresponds to an image pixel, each edge connects the pair of adjacent vertices. Non-negative weights are applied to measure the dissimilarities of connected vertices. The principal goal then is to segment the vertices into different components. However, the value of the corresponding weight is not computed by the connected elements, but ones with a distance of d_n, which is equivalent to [28] when d_n equals to 1. This may seem problematic, but it effectively alleviates the effects of the slow changes of image intensities around the sky-ground boundary. In similar areas, the dissimilarities of image intensities are small, so that the increase of d_n does not affect the value of weight. However, the image intensities gradually change nearby the boundary line, and the increase of d_n leads to the accumulation of the gradient along distance d_n, thereby increasing the dissimilarities. Through modifying weights, the GHLD approach will preferentially process the similar regions and then the areas around the edges.

As for the initial state of segmentation, each vertex corresponds to a component. To measure the local difference between two connected components, we calculate the internal dissimilarity of each component and local dissimilarity between the two connected components. For simplicity, $\mathrm{Int}(C_n)$ denotes the internal dissimilarity of component C_n, which equals to the largest weight in the MST of C_n; $\mathrm{Dif}(C_m, C_n)$ denotes the local dissimilarity between the two connected components C_m and C_n, which equals to the minimum weight edge connecting the two components. Besides, to measure the global saliency, we compute the mean weight and mean intensity of each component, denote as $\mathrm{MnW}(C_n)$ and $\mathrm{MnI}(C_n)$ respectively. We then define the merger criteria to evaluates whether to merge the

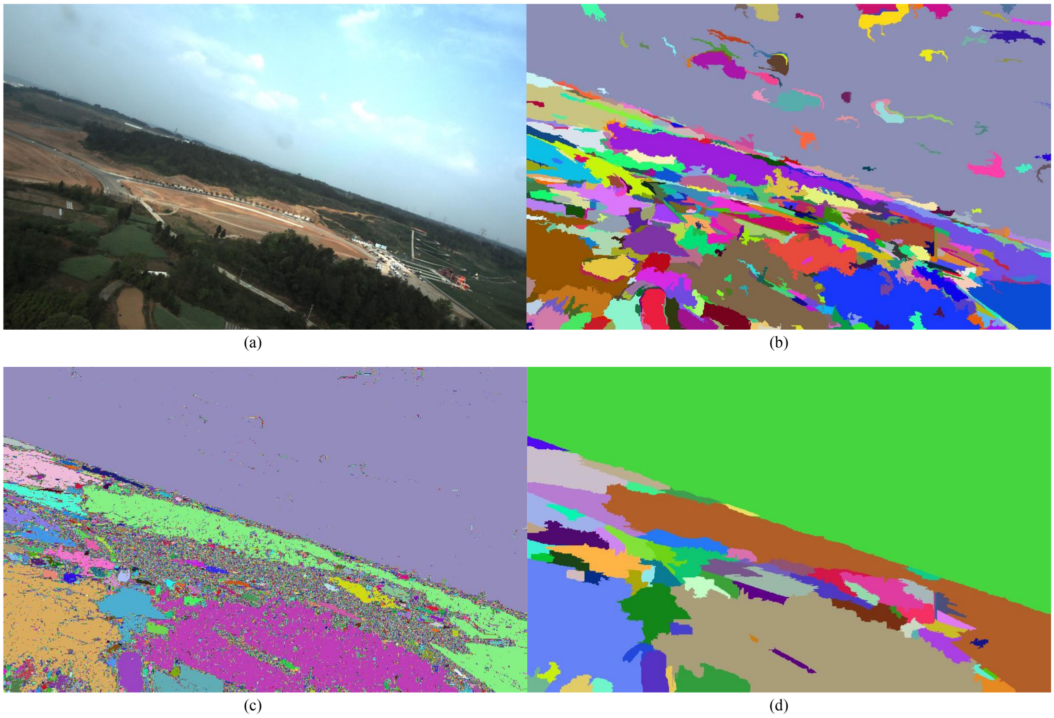

(a) (b) (c) (d)

Fig. 3. Segmentation results. (a) Original image. (b) Segmentation results by [28] with postprocessing, $\boldsymbol{k} = 50$, $\mathbf{ThrS} = 100$. (c) Segmentation results by our algorithm. (d) Postprocessing results of (c).

connected two components into one. That is

$$\begin{cases} Dif\left(C_m, C_n\right) < MInt\left(C_m, C_n\right) \\ \left|\mathrm{MnI}\left(C_n\right) - \mathrm{MnI}\left(C_m\right)\right| < ThrI \\ \left|\mathrm{MnW}\left(C_n\right) - \mathrm{MnW}\left(C_m\right)\right| < ThrW \end{cases} \tag{1}$$

where ThrW and ThrI denote the thresholds of edge weight difference and image intensity difference, respectively. $\mathrm{MInt}(C_m, C_n)$ is the threshold of local dissimilarity of the connected two components, defined as

$$\mathrm{MInt}\ \left(C_m, C_n\right) = \min\left(\mathrm{Int}\left(C_m\right) + \tau\left(C_m\right), \mathrm{Int}\left(C_n\right) + \tau\left(C_n\right)\right) \tag{2}$$

where $\tau\ (C) = \ k/|C| + b$, $|C|$ denotes the size of C, k, and b are positive constant parameters. Let ThrS be the threshold of component size. A small component whose size is less than the threshold should be merged to the connected large one by post-processing.

In this article, we set parameters $d\_n = 3$, $k = 5$, $b = 3.5$, $\mathrm{ThrW} = 6$, $\mathrm{ThrI} = 50$, $\mathrm{ThrS} = 100$. Fig. 3(d) shows the results produced by our improved segmentation algorithm, where details in the image are removed, semantic components such as the sky, forest, farmland, and the motor road are retained. Comparing to the results produced by [28], which are shown in Fig. 3(b), our results are more conducive to the subsequent horizon line detection task.

C. Connected Domain Cascade Filtering

Given the segmentation results, we have to determine whether the image contains the horizon line. To determine the horizon line, we first try to find the sky-component instead. As discussed in Section III-A, the sky-component has three important characteristics comparing to other connected domains, listed as following.

1) First, the sky-component occupies a relatively large part of the entire image.
2) Second, the mean intensity of the sky-component is greater than that of the ground part.
3) Third, the sky-component is at least connected to one of the image boundaries.

Based on the above characteristics, we propose a connected domain cascade filtering technique to evaluate whether or not there is a component corresponding to the sky area, meanwhile accelerating the filtering process. To determine the sky-component from the segmentation results, we first calculate the attributes of each connected domain, including size, centroid, average intensity, circumscribed rectangle, and the number of points that coincide with the image boundaries. Then, we calculate the score of each connected domain, which measures the likelihood of the corresponding connected domain to be the sky-component. That is

$$\mathrm{Scor\ e}_n = \left(\mathrm{Size}_n/\max\_S + \mathrm{Intensity}_n/\max\_I + B\_\mathrm{num}_\mathrm{n}/\max\_B\right)\ /3 \tag{3}$$

where Size indicates the size of the connected domain, $\mathrm{Intensity}$ indicates the average intensity, and B_num indicates the number of points at the image boundaries. $\max\_S$, $\max\_I$, and $\max\_B$ denote the maximum of the above three parameters respectively.

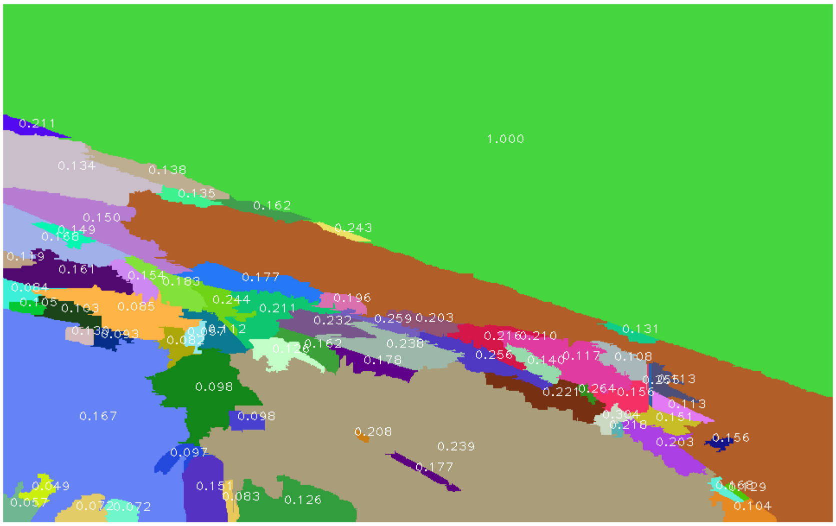
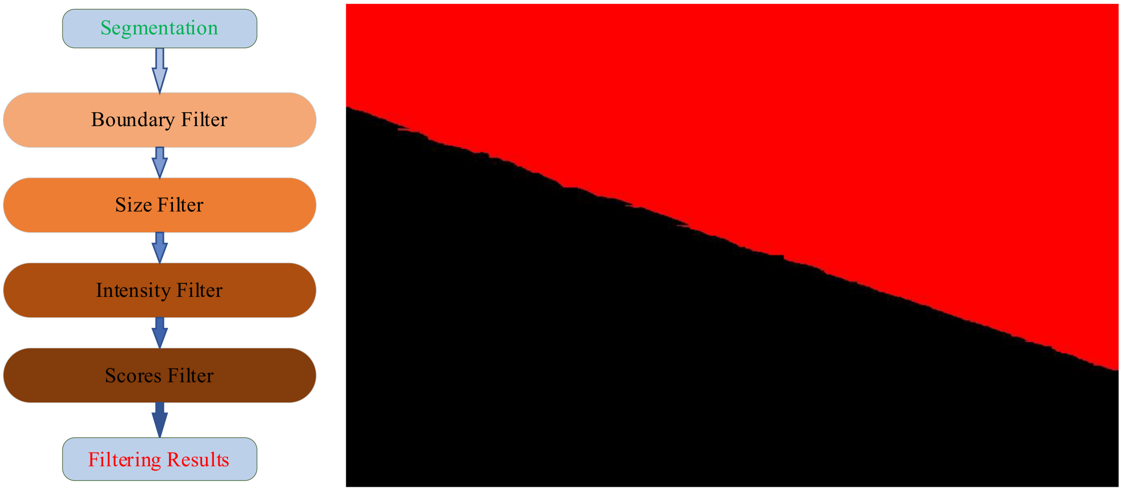


Fig. 4. Scores of the connected domains.



(a) (b)

Fig. 5. Processing pipeline and results of connected domain cascade filtering. (a) Pipeline of the cascade filtering. (b) Filtering results.

From (3), the score of each component ranges from 0 to 1, which indicates the probability of the corresponding component to be the sky part. Fig. 4 shows the calculated scores of the collected domains obtained in Section III-B. As shown in Fig. 4, the score of the sky-component equals 1, while scores of the non-sky ones are less than 0.3. In this case, we can directly determine the sky-component by finding the one with the maximum score. However, sometimes the segmentation results may be contaminated by occludes or suffer from severe weather conditions, it's difficult to decide the sky-component only depending on the calculated scores. To ensure the correctness of the results and speed up the algorithm, we filter the segmentation results in a cascaded manner. Fig. 5 illustrates the pipeline of the cascade filtering and the results of this module.

As shown in Fig. 5(a), the components that are not connected to the image boundary are first removed by the boundary filter. Second, the size filter removes components whose sizes are less than threshold $\mathrm{Thr\_size}$. Then, the intensity filter removes components, the average intensities of which are less than the average intensity of the image. Finally, the scores filter removes components whose scores are less than the threshold $\mathrm{Thr\_score}$ and choose the connected domain with the highest score as the desired sky-component. In this module, we use $\mathrm{Thr\_score} = 0.6$, $\mathrm{Thr\_size}$ equals one-tenth of the image size, and the filtering results are shown in Fig. 5(b).

D. Horizon Line Extraction

Once the sky-component is determined, we apply LSD [29], clustering, and length-weighted average to further extract the horizon line. First of all, we extract the boundary between the sky-component and the ground-component and remove the spurious edges caused by undesirable segmentation. Second, we employ the LSD method to detect line segments in the

Algorithm 1: Horizon Line Extraction Algorithm.

```
Inputs: The sky-component determined by cascade filtering.
1: Extract the contours of the sky-component and remove the parts that coincide with the image boundary.
2: Label the contours, exclude spurious ones, and choose the one with max size as the sky-ground boundary.
3: Employ the LSD method to detect line segments in the binary mask where the sky-ground boundary is located.
4: Calculate the angle $A$, length $L$, and the middle point $(mx, my)$ of each line segment.
5: For $p = 1, \ldots, m$ :
    Take $A_p$ as the clustering center, let the cluster threshold $A_{\mathrm{thr}} = 10^o$ , and clustering score Scor e$_p =$ 0.
    For $q = 1, \ldots, m$ :
        If $\mathrm{abs}(A_p - A_q) < A_{\mathrm{thr}}$ :
            Calculate the contribution of line segment $q$ to cluster center $A_p$, based on the length and angle of $q$,
            $\mathrm{Score}_p + = (1 - \mathrm{abs}(A_p - A_q)/(2{*}A_{\mathrm{thr}})) *L_q$.
            Add the index $q$ to the cluster Lines$_p$.
6: Choose the cluster with the highest score as the candidate set, and then apply RANSAC to further eliminate the outliers in the candidate set.
7: Calculate the slope angle of the horizon line and estimate the point which lies on the line, adopting the lengths of the candidates as weights.
Outputs: The slope angle of the horizon line and the point through which the horizon line passes.
```

image area where the sky-ground boundary is located. To handle the nonlinear horizon Line scene, we cluster the angles of the detected line segments, calculate the score of each cluster, and choose the cluster with the highest score as candidates, the specific processes are shown by Algorithm 1 Steps 5 and 6 in pseudocode. Finally, we adopt the lengths of the candidates as weights to calculate the slope angle of the horizon line and the point through which the horizon line passes. As shown in Algorithm 1, the complete horizon line extraction procedure is detailly presented in pseudocode.

To exclude the pseudoboundaries, we first extract the contours of sky-component and remove the pseudocontours that have a father contour. Then, we remove the parts that coincide with the image boundary and label the remain boundaries by an eight-neighborhood connected domain labeling algorithm. Finally, we choose the one with max size as the sky-ground boundary.

To further reduce the execution time and ensure the correctness of horizon line extraction, all steps are performed utilizing the binary image nearby the sky-ground boundary, as shown in Fig. 6. The results shown in this figure correspond to the *horizon line extraction* module in Fig. 1. Fig. 6(a) illustrates the sky-ground boundary by culling the spurious edges and the points that coincide with the image boundary. Fig. 6(b) shows the extracted line segments by LSD, and about 20 segments are found. Through applying clustering and RANSAC techniques, the remained candidates are shown in Fig. 6(c), it illustrates that all the extracted segments are preserved. Fig. 6(d) demonstrates the output horizon line and points of Algorithm 1.

E. UAV Attitude Estimation

1) Attitude Estimation: For simplicity, we assume that the axes of the camera coordinate system are parallel to the axes of the UAV coordinate system (see Fig. 7). Then, the roll and pitch angles of the UAV can be directly determined from the extracted horizon line. We define that the roll angle is positive when the UAV rolls right and the pitch angle is positive when the UAV raises head, as shown in Fig. 7.

The origin of the image reference is defined as the upper left corner of the image. Let θ be the slope angle of the horizon line in the image coordinate, S_1 be the size of the sky-component segmented by the horizon line, (x_1, y_1) be the centroid of the sky-component, S_0 be the size of the image, and F_A be the field angle of the camera at the vertical orientation. Since the horizon line is orthogonal to the camera's rotation axis, the roll angle of the UAV is invariant to all other motions. We employ the positional relationship between the sky-component and the horizon line to eliminate the ambiguity of the roll angle through (4) and (5). If the horizon line is orthogonal to the x-axis, denoted by $x = d$, then the roll angle is determined by

$$\mathrm{roll} = \begin{cases} +90 \; if \; x_1 < d \\ \pm 90 \; if \; x_1 = d \\ -90 \; if \; x_1 > d \end{cases} \tag{4}$$

Otherwise, the equation of the horizon line can be denoted as $y = k * x + b$, and then the roll angle is determined by

$$\mathrm{roll} = \begin{cases} -\theta & if \; y_1 - k * x_1 - b < 0 \\ -\theta \; or - \theta + sgn\,(\theta) * 180 & if \; y_1 - k * x_1 - b = 0 \\ -\theta + sgn\,(\theta) * 180 & if \; y_1 - k * x_1 - b > 0 \end{cases} \tag{5}$$

where $\theta = \tan^{-1}(k)$ and $\mathrm{sgn}(\theta)$ is the sign function. Besides, the uncertainty of the roll angle in formulas (4) and (5) can be eliminated according to the continuity of the aircraft attitude in the time domain. Besides, the pitch angle is proportional to the size of the sky-component in case of no rolls. Based on the sizes of sky-component and ground-component, the pitch angle of the UAV is approximated by

$$\mathrm{pitch} \approx F\_A{*}\left(S_1 - S_0/2\right)/S_0. \tag{6}$$

Due to the limitation of the camera's field of view, the measurable range of pitch angle is $(-F\_A/2, +F\_A/2)$. Fig. 8 shows that when the pitch angle is fixed, the changes in the roll angle lead to the changes of the sky-component area.

The red parts indicate the equivalent increment of the sky-component size caused by the rolls and the black parts denote the equivalent reduction of the sky-component with respect to the corresponding rolls. Thus, the size of the red part subtracts the size of the black part equal to the increment of the sky-component caused by the rolls. Generally, the width of the image is greater than the height of the image. As shown in the second row of Fig. 8, the greater the roll angle, the greater the error.

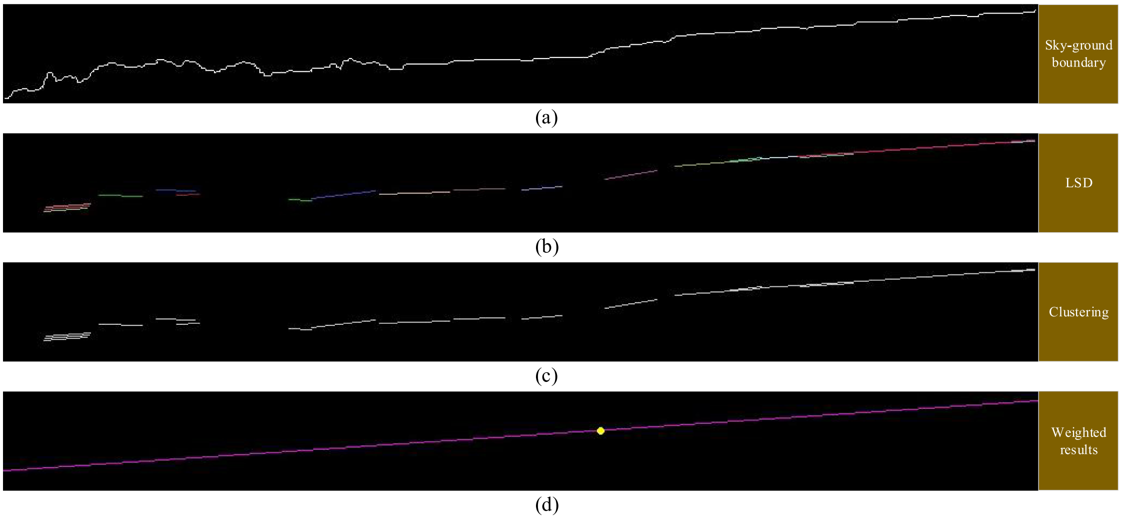


Fig. 6. Results of the horizon line extraction module. (a) Extracted sky-ground boundary. (b) Candidates by LSD. (c) Results of clustering and RANSAC. (d) Length-weighted horizon line.

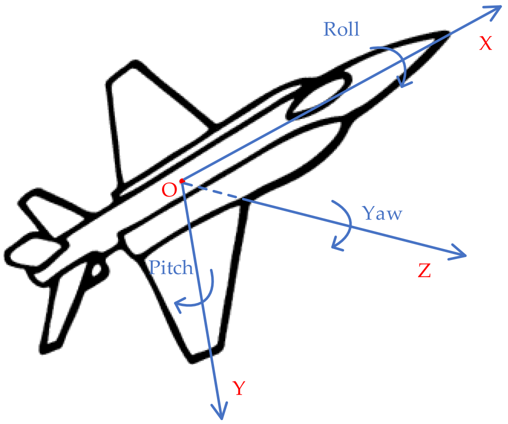


Fig. 7. Body coordinate system and triplet of Euler angels for UAV.

A large roll angle will increase the size of the sky-component, resulting in a smaller calculated pitch angle compared to the actual one. The quantitative results in Fig. 13 also illustrate this point. In addition, using a square image area to calculate the pitch angle will further reduce errors.

2) Attitude Rectification: For actual application, the axes of the UAV reference system are not exactly parallel to the ones of the camera coordinate system. Consequently, the attitude angles estimated above cannot be straightly applied for the UAV navigation task. To eliminate the installation error between the camera and the

UAV, we have to calibrate the transformation from the UAV reference system to the camera coordinate system. The calibration consists of the following three subtasks.

1) Calibrate the internal parameters of the camera, including principal point, focal length, and the distortion coefficients.
2) Calibrate the noise values of the IMU, including the noise density and random walk coefficients of the accelerometer and gyroscope, respectively.
3) Calibrate the joint transformation between the camera and the body, including rotation and translation.

In this article, we employ the open-source software imu_utils [30] to calibrate the IMU parameters and utilize the open-source visual-inertial calibration toolbox Kalibr [31] to calibrate the camera internal parameters and the camera-body joint transformation. For the specific details, see [30] and [31]. Assuming that the calibrated rotation matrix denotes R and the translation vector denotes T. R and T indicate the transformation from the UAV reference system to the camera reference system. An alternative parametrization to the rotation matrix R is a triplet of roll, pitch, and yaw. We decompose the rotation matrix R in the order *of x-y-z,* that is

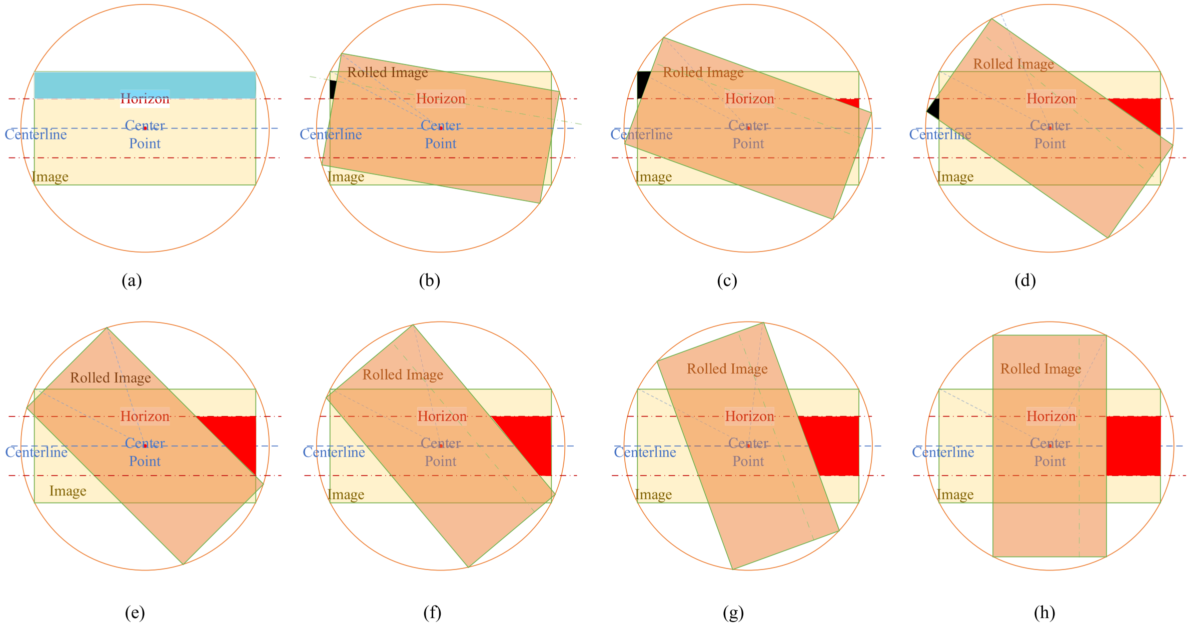
$$R \ = R_x\ (\gamma) * R_y\ (\alpha) * R_z\ (\delta) \tag{7}$$

where γ, α, δ denote the roll, pitch, and yaw angles respectively. $R_x(\gamma)$, $R_y(\alpha)$, $R_z(\delta)$ are the basic rotation matrixes along x-axis, y-axis, and z-axis respectively. Therefore, we can exploit the calibration results to rectify the estimated roll and pitch angles by

$$\begin{cases} \mathrm{roll}_r = \mathrm{roll}_e \ - \gamma \\ \mathrm{ptich}_r \approx \mathrm{pitch}_e - \alpha \end{cases} \tag{8}$$

where roll_e and pitch_e denote the above estimated results, roll_r and ptich_r denote the rectified ones.

To ensure the reliability of the detection results, we perform consistent filtering on the detection results based on the continuity of the physical events in the temporal and spatial dimensions to eliminate the outliers. If the detection results of the succeeding frame change a lot, the detection results are considered to be unreliable. In the time domain, we use the second-order Kalman filter and Gaussian weighted filter to process the detection results and remove the high-frequency part of the results. In addition, we exploit the detection results to eliminate outliers and supervise the selection of cluster centers and candidate sets, in the subsequent horizon line extraction task.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 8. Changes of the sky-component concerning the roll angles. (a)–(h) Roll angle increases from 0° to 90°.

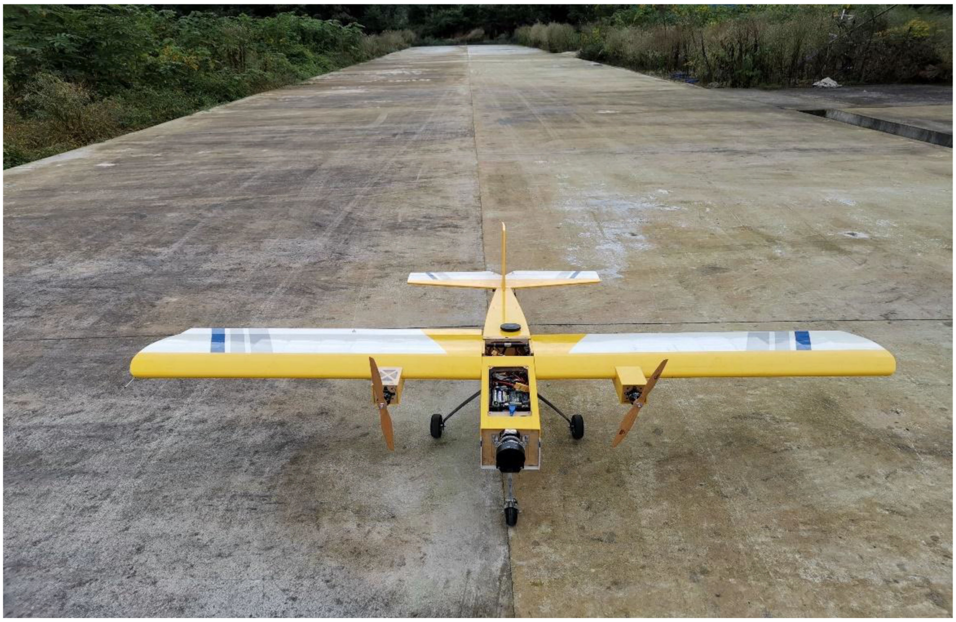


Fig. 9. Fixed wing UAV testbed.

IV. Experiments

A. UAV System

To validate the feasibility of the proposed algorithm, a UAV system as shown in Fig. 9 was designed. This fixed-wing UAV has a wingspan of 2.2 m, an empty weight of 5.6 kg, and a gross weight of 7.5 kg. Images were captured by a forward looking OSG230-150UC industrial camera at 60 fps, a focal length of $6\ \mathrm{mm}$, $1920 * 1200$ resolution, and $73.3^{o} * 57.5^{o} * 86.6^{o}(H * V * D)$ field of view, shown in Fig. 10(a). We mounted the camera on the nose of the UAV and made the axes of the camera coordinate system parallel to the ones of the UAV coordinate system as much as possible through high-precision alignment. The UAV using an INTEL NUC8i7BEH with 16 GB of RAM and a CPU with a speed of 3.0 GHz, and vision processing is performed onboard. In order to reduce the weight, we disassembled the shell of the NUC, as shown in Fig. 10(b). We employed a Pixhawk 4 as the flight controller, which integrates IMU, magnetometer, barometer, and other sensing devices, shown in Fig. 10(c). To facilitate communication between different programs, we have established this system based on the robot operating system. The NUC8i7BEH communicates

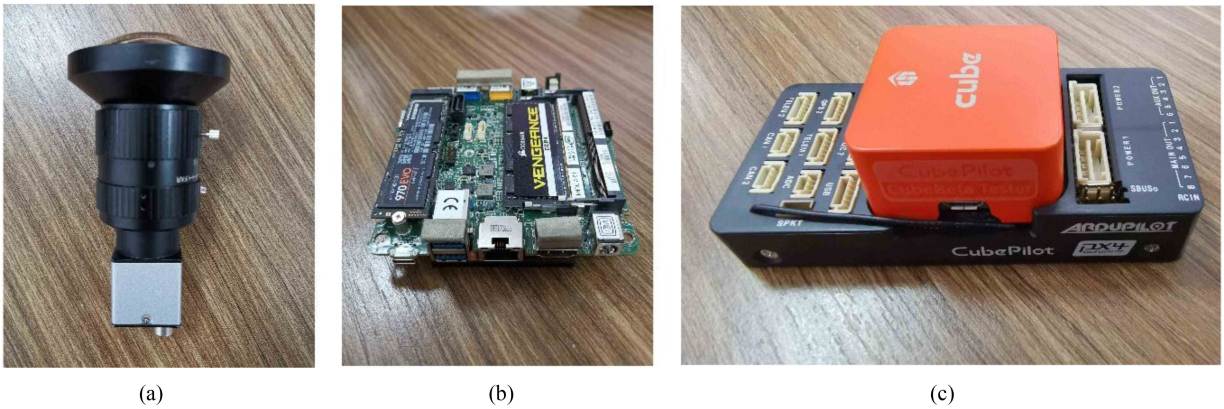


Fig. 10. Devices integerated in the UAV system. (a) OSG230-150UC industrial camera. (b) NUC8i7BEH vision processor. (c) Pixhawk4 flight controller.

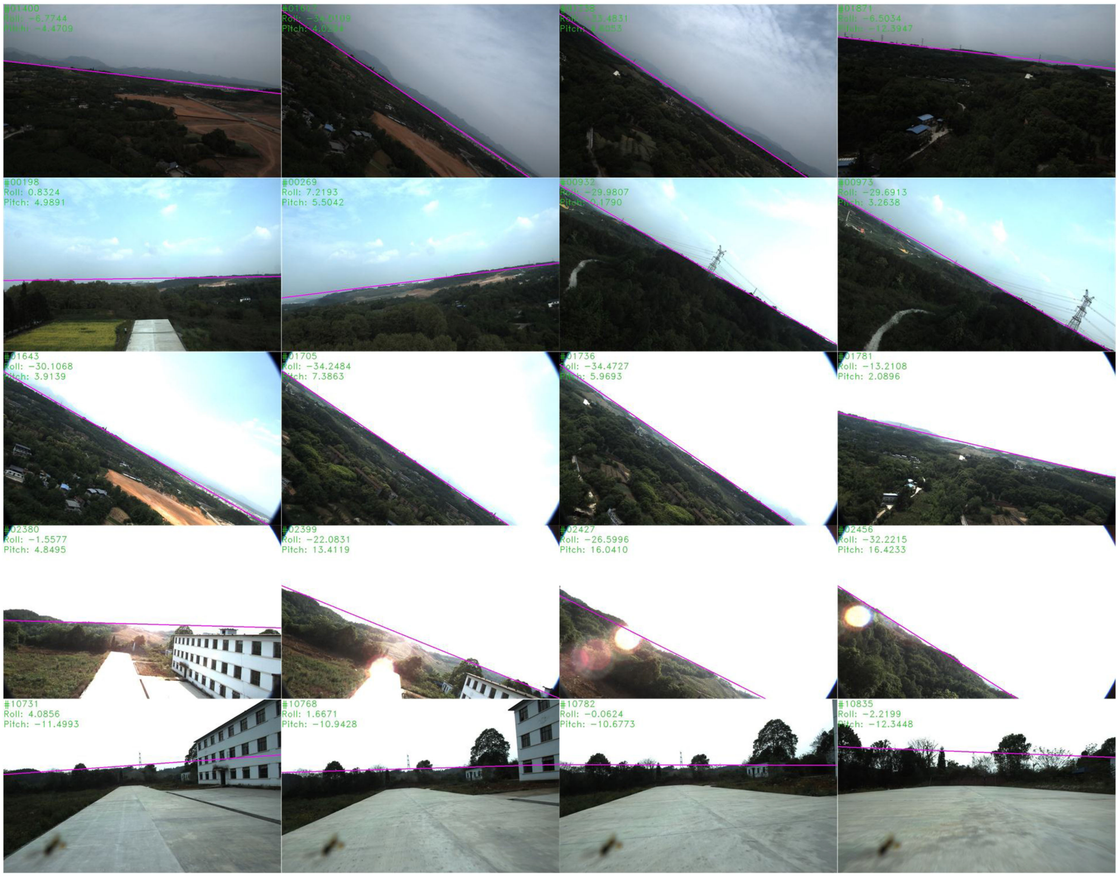

#01400 Roll: −6.7744 Pitch: −4.4709
#01617 Roll: −34.0109 Pitch: 4.0284
#01738 Roll: −33.4831 Pitch: 4.4053
#01871 Roll: −6.5034 Pitch: −12.3947
#00198 Roll: 0.8324 Pitch: 4.9891
#00269 Roll: 7.2193 Pitch: 5.5042
#00932 Roll: −29.9807 Pitch: 2.1790
#00975 Roll: −29.6913 Pitch: 3.2638
#01643 Roll: −30.1068 Pitch: 3.9139
#01705 Roll: −34.2484 Pitch: 7.3863
#01736 Roll: −34.4727 Pitch: 5.9693
#01781 Roll: −13.2108 Pitch: 2.0896
#02380 Roll: −1.5577 Pitch: 4.8495
#02399 Roll: −22.0831 Pitch: 13.4119
#02427 Roll: −26.5996 Pitch: 16.0410
#02456 Roll: −32.2215 Pitch: 16.4233
#10731 Roll: 4.0856 Pitch: −11.4993
#10768 Roll: 1.6671 Pitch: −10.9428
#10782 Roll: −0.0624 Pitch: −10.6773
#10835 Roll: −2.2199 Pitch: −12.3448

Fig. 11. Qualitative horizon line detection results with different flight conditions. The First row: low illumination. The second row: Cloudy. The third row: Sunny. The fourth row: Backlighting. The last row: Landing.

with the Pixhawk 4 through a serial port. The current settings support the NUC8i7BEH to transmit computed roll and pitch angles to the Pixhawk 4 and receive aircraft telemetry, including differential GPS coordinates, barometric altitudes, airspeed, and the reference roll, pitch, and yaw estimated by Pixhawk 4.

During actual flight tests, extensive automatic flights with different conditions were conducted and comparisons were also made to the IMU estimation. To test the influence of flight altitudes on the presented method, the flights are performed under various designed altitudes, including 50, 20, and 10 m. To verify the adaptability with respect to illumination, flights are performed in the presence of dim illumination, cloudy, and backlighting. To test the robustness of our algorithm corresponding to terrains and obstacles, we implemented the test in the hilly areas with rich scenes. Besides, the designed speed of the UAV was 20 $\mathrm{m/s}$, and the images were downsampled to $480 * 300$ to ensure real-time performance. Moreover, because the field of view might be occluded by houses and trees during takeoff and

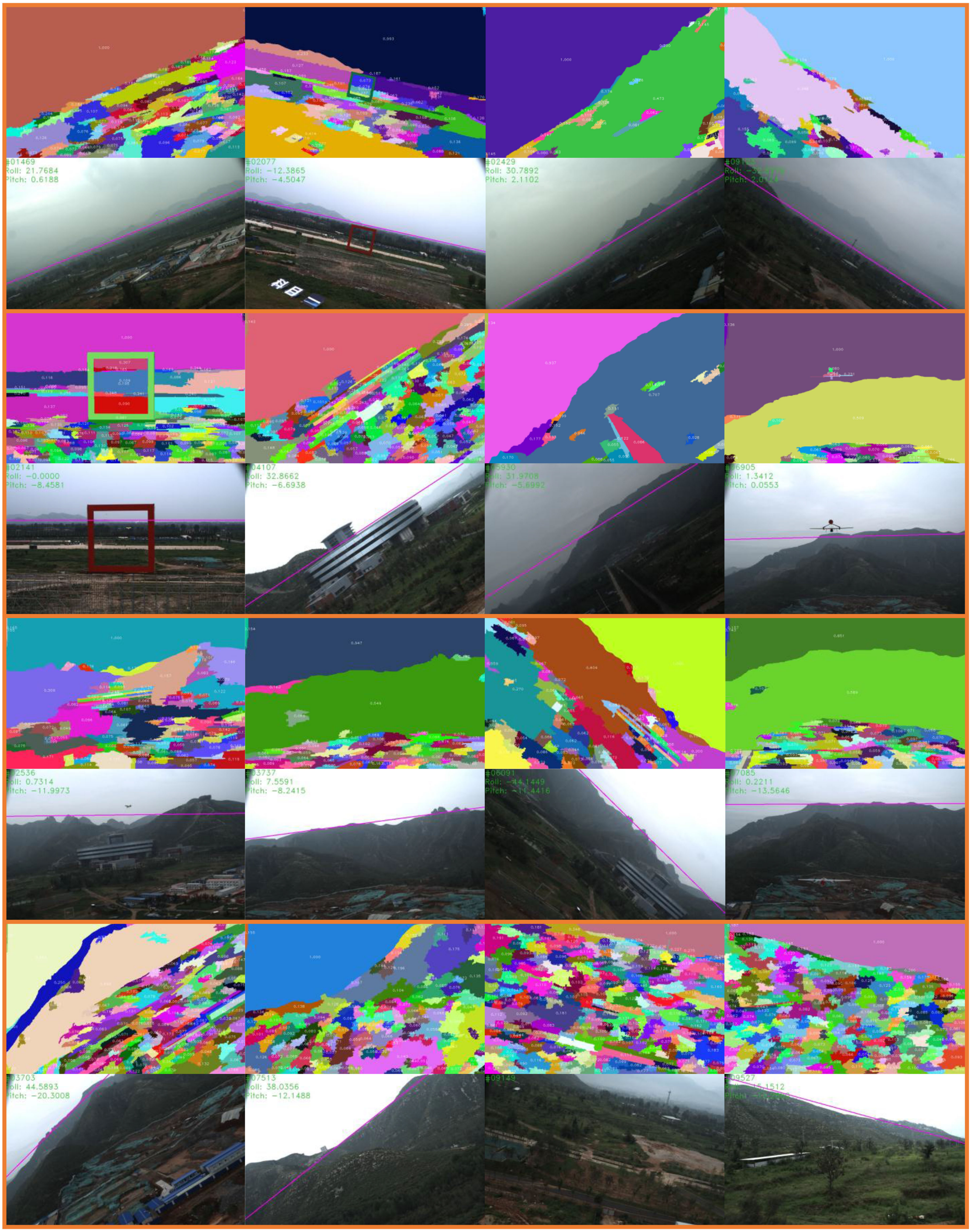
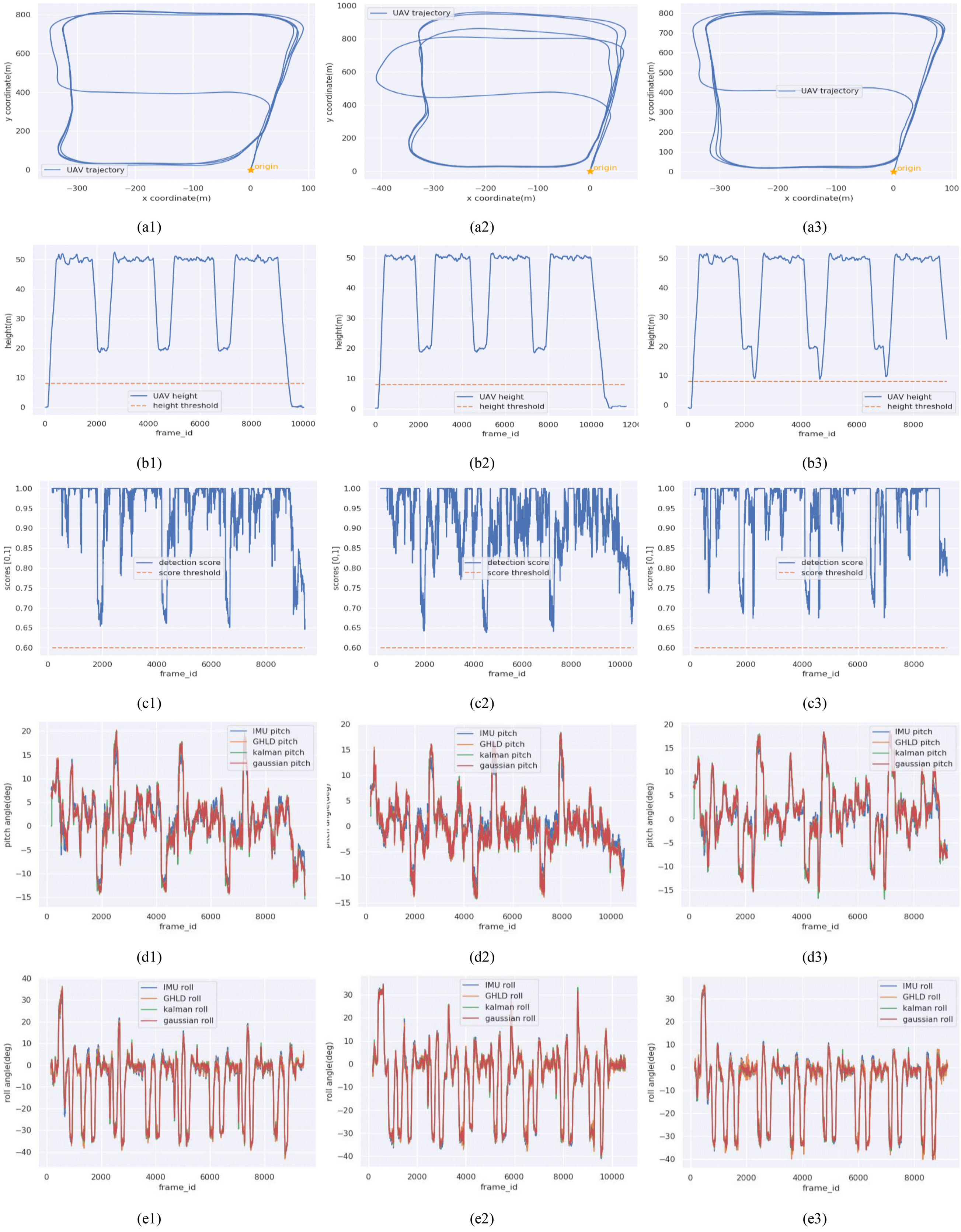


Fig. 12. Horizon line detection results in nonlinear scenes. The first group: simple nonlinear scenes. The second group: difficult nonlinear scenes. The third group: difficult nonlinear scenes when the UAV flew close to the high mountains. The fourth group: challenging nonlinear scenes.



(a1) (a2) (a3)

(b1) (b2) (b3)

(c1) (c2) (c3)

(d1) (d2) (d3)

(e1) (e2) (e3)

Fig. 13. Quantitative results of three flight tests. Left column: Flying on a sunny day. Middle column: Flying on a dark and windy day. Right column: Flying in the evening with backlighting. (a*) UAV trajectories in *x*-*y* plane. (b*) Flying altitude of the UAV. (c*) Sky-component scores. (d*) Comparison of the pitch angles. (e*) Comparison of the roll angles.

landing, to ensure the reliability of detection, we set the altitude threshold equal to 8 meters, that was, the horizon line detection was performed only when the flying altitude of the UAV was greater than the threshold. Qualitative results are illustrated in Fig. 11, and quantitative comparison results are shown in Fig. 13.

B. Qualitative Results

Fig. 11 exhibits the results of the proposed algorithm performed on five groups of images. Each row of images corresponds to one flight test, the illumination of which increases from the first row to the fourth row, and the last row shows the detection results during landing. As shown in the first row, the proposed GHLD can deal with the distant mountains and irregular sky-ground boundaries, in spite of low illumination. For the second row, the tests were performed on a cloudy day, it shows that our algorithm achieves favorable results in the presence of a high-voltage line. The tests in the third row were performed on a sunny day, in this case, the sky part was overexposed, which is beneficial to the graph-based image segmentation and results in satisfactory results. The images shown in the fourth row were captured when the UAV flew over the ground station at an altitude of about 10 m and meanwhile, the sunshine came in face of the camera. The results show that the presented GHLD found the correct horizon line despite backlighting. The last row in Fig. 11 illustrates the results when the UAV was landing. In this case, the results of image segmentation may severely suffer from the ground clutter, such as buildings and trees, which leads to performance degradation of the horizon line detection algorithm. Results shows that the estimated roll angles are favorable. However, the estimated pitch angles obviously depart from the actual ones, which are almost 0° during landing. Thus, for safety considerations, we did not apply the detected angles for navigation.

To further demonstrate the feasibility and reliability of the proposed fixed-wing UAV system and horizon line detection technology, we participated in the 2021 UAV Challenge Competition in a formation of 4 UAVs for the simulation docking race and in a formation of 7 UAVs for the speed crossing race. The racing field is a complex scene with mountains and buildings, resulting in nonlinear scenes. The nonlinear here mainly refers to that the sky-ground boundaries are nonlinear and it is mainly caused by the ups and downs of the terrain. In most cases, the flight scenes are nonlinear, especially when there are mountains or buildings in the field of view, as shown in Fig. 12. Corresponding horizon line detection results are shown in Fig. 12 and the results are divided into four groups by bold orange lines. For each group, the upper row exhibits the segmentation results with scores and the bottom row shows the corresponding detection results. The images in the first group were captured when the UAV flew at an altitude of about 20 m and the mountains were far away from the UAV. The results illustrate that the proposed GHLD can handle these simple nonlinear scenes. The second group shows the more difficult nonlinear scenes. The test images were captured when the UAV was crossing the gate, flying nearby the buildings, flying toward the mountains, and chasing after the simulation refueling aircraft. The results show that our algorithm can deal with these scenarios despite the highly nonlinear sky-ground boundaries. For the third group, the test images were captured when the UAV flew close to the high mountains. As shown in the images, the mountains occupy most part of the images and the sky-ground boundaries are also nonlinear. Results show that the presented GHLD detects the satisfactory lines which can be applied to compute the rolls and pitches. However, in this case, the estimated pitch angles obviously depart from the actual ones, because of the serious occlude of the mountains, same with the landing case shown in Fig. 11. The fourth group shows the challenging nonlinear scenes that the proposed GHLD fails to detect the horizon lines. In this case, the UAV was too close to the ground or the mountains, causing that most of the images were occupied by the ground part. In addition, the local boundaries of high mountains will also affect the detection of the horizon. The processing method to these challenging nonlinear horizon line scenes fall into the third category discussed in Section II, which utilizes the deep learning approaches to infer the horizon lines.

To sum up, the proposed GHLD is robust to both linear and most nonlinear scenes and adapts to dynamic changes in flight conditions such as topography, lighting conditions, and altitude.

C. Quantitative Analysis

To further analyze the performance of GHLD, we adopt the UAV attitude values output by IMU as the reference values, which are estimated inside the Pixhawk 4 by integrating GPS and magnetometer. We test the performance of the proposed algorithm by comparing the difference between the detected values and the reference ones. Fig. 13 shows the results of our approach applied on three different flights.

The left column illustrates the results with strong light conditions. The middle column illustrates the results corresponding to low illuminance and windy conditions. The right column shows the results in the evening with the presence of backlighting. Fig. 13(a1)–(a3) illustrate the projection of the UAV trajectories on the *X*–*Y* plane, Fig. 13(b1)–(b3) show the altitude of the flights, and Fig. 13(c1)–(c3) illustrate the scores of the sky-components calculated using (3). Fig. 13(d1)–(d3) and (e1)–(e3) illustrate the estimated pitch and roll angles, respectively. In both cases, the estimated angles are presented for IMU, GHLD, Kalman filter, and Gaussian filter.

The Kalman filter adopts a second-order motion model to predict the corresponding angle, angular rate, and angular acceleration. The initial values of these state variables are computed by differentiating the recent measurements. Besides, the measurement noise of our GHLD is modeled by an additive white Gaussian noise and the variance of the measurements noise is obtained by offline statistics. As for the Gaussian filter, we employ a set of Gaussian coefficients to weight the recent K measurements and the coefficients are precomputed by function $\mathrm{Coef} = \exp(-0.5 * (k/\mathrm{sigma})^2)$ followed by normalization. In this article, we set $K = 10$ and $\mathrm{sigma} = 3.0$, thus the coefficients are [0.0026, 0.0067, 0.0154, 0.0318, 0.0586, 0.0966, 0.1426, 0.1882, 0.2223, 0.2350].

TABLE I
STATISTICS OF EXPERIMENTAL RESULTS

Attitude	Flight No.	Mean of GHLD error (°)	Variance of GHLD error (°)	Mean of KALMAN error (°)	Variance of KALMAN error (°)	Mean of Gaussian error (°)	Variance of Gaussian error (°)
Roll	1	0.1187	2.5397	**0.1069**	2.4898	0.1192	**2.4530**
	2	0.0465	1.4968	**0.03932**	1.4668	0.0472	**1.3445**
	3	0.1661	3.6781	**0.1537**	2.8565	0.1662	**2.4625**
Pitch	1	**-0.0071**	**1.8407**	0.0116	2.4341	-0.0117	1.9185
	2	0.0576	**1.9706**	0.0636	2.5833	**0.0550**	2.1575
	3	-0.2233	2.0220	**-0.2089**	2.4464	-0.2267	**1.9356**

The results in Fig. 13(b∗) and (c∗) demonstrate that the scores of the sky-components are disturbed by the flight altitude. The higher the altitude is, the higher the score is, and vice versa. Because the higher the flight altitude is, the boundary between the sky and ground is closer to a straight line in the image, and the detection algorithm is less disturbed by the ground clutter. Comparison with Fig. 13(c1)–(c3), we also obtain that the scores of sky-components are affected by flight conditions. As shown in Fig. 13(c2), (d2), and (e2), the windy weather resulted in fluctuating results, because the attitude of the UAV is fluctuated by the wind. Fig. 13(d*) and (e*) illustrate that the pitch and roll angles produced by GHLD, Kalman filter, and Gaussian filter are almost consistent with the measured values of IMU. The specific statistical results are given in Table I.

Table I gives the error statistics of the roll and pitch angles estimated by GHLD, Kalman, and Gaussian relative to the IMU reference values. The error means and error variances of the estimated values relative to the reference ones are counted. It is manifest from the table that the mean value of the error is about 0, which shows the unbiasedness of the detection results; the variance of the error is about 2 degrees, which shows the accuracy of the proposed algorithm.

In engineering practice, for high-frequency requirements and limited airborne resources, we can downsample the image to a size of 240×150. Image downsampling may have some slight impact on the sky-ground boundaries, but this process will not degrade the accuracy of the horizon line extraction, since the horizon line represents the global semantic information and we extract the horizon line in a clustering and length-weighted average way. By the means of engineering optimization, our algorithm can run at approximately 60 fps. Meanwhile, by decreasing the exposure time of the camera, we find that the intensity difference between the sky-component and ground-component changes to indistinct, which results in performance degradation of the graph-based segmentation algorithm. Thus, slightly increasing the camera's exposure time is conducive to image segmentation. However, to avoid blurring of images caused by high-speed motion, the exposure time should be set to no more than 40 ms. To reduce the influence of the backlight, we install a hood and filter on the lens. Besides, to ensure the reliability of navigation tasks, the horizon detection algorithm runs after the UAV reaches a certain altitude to avoid ground objects blocking the camera's view.

V. DISCUSSION

A. Result Analysis

Herein, we propose a GHLD method to promote the task of UAV navigation. The results reveal that the angles produced by our method are almost consistent with the reference values by the IMU and our method adapts to the dynamic changes of illumination, terrain, and altitude. Besides, the optimized algorithm executes at the speed of 60 fps, which fulfills the dynamic requirements of the fixed-wing UAV navigation. Although the study of this article achieves satisfactory results in the actual flight tests, some limitations are also noteworthy.

1) Only the roll and pith are estimated by the detected horizon line, the yaw and absolute positions of the UAV are not determined. Thus, the proposed method can only measure part of attitude angles and exploit them to stabilize the aircraft.
2) The measurable range of pitch angle is limited by the camera's field of view, which means that the pitch is unmeasurable when the drone severely pulls up or dives. Besides, the pitch angle calculated by formula (6) is an approximation and affected by the roll, which decreases the accuracy of estimated pitch angle.
3) Our vision system is composed of a visible light image sensor, which may hinder by the atmospheric and illumination conditions. For example, on foggy days or at night, due to low visibility, our system cannot work.
4) Although results in Fig. 11 show that the proposed method achieves acceptable rolls during landing. For reliability concerns, we apply the detected angles for navigation only when the UAV reached a certain altitude, to reduce the influence of the ground clutter.

B. Future Works

Our future work will strive to address the above limitations of the proposed horizon detection method, particularly in the case of dark and obstacles. Meanwhile, we will apply the proposed method on infrared images to alleviate the hamper of atmospheric and illumination conditions. For some difficult situations, we intend to study a fast and robust artificial intelligence method to recognize the horizon line or even infer the unseen horizon line.

Furthermore, to accomplish fully autonomous navigation in the absence of the navigation satellites, we will study the hybrid between the SALM algorithms and horizon line-based navigation methods, which supplement and promote each other. As presented in [7], [9], and [10], the SALM algorithms can estimate the accumulated positions and attitudes of the UAV, which are required by UAV navigation. However, the horizon-based navigation methods can directly provide the absolute roll and pitch orientations, which can assist to mitigate the drifts in the SLAMs.

VI. Conclusion

The horizon line has various applications for UAVs, such as obstacle detection, attitude stabilization, path planning, and autonomous navigation. However, it's pretty difficult to detect the horizon line, especially for fixed-wing UAV applications, due to the remotely sensed big data, the dynamic changes in flight, and the serious consequences of failure. To fulfill the requirement of horizon line detection for fixed-wing UAV navigation, an effective horizon line detection method, GHLD, was proposed in this article. We divide the horizon line detection task into four subtasks. First, we improve the graph-based image segmentation algorithm by modifying the weight calculation of the edges and the merging criteria of connected domains. Second, we determine the sky-component by cascading the boundary filter, size filter, intensity filter, and scores filter. Then, we extract the horizon line by employing LSD, clustering, RANSAC, and length-weighted average on the sky-ground boundary. Moreover, we eliminate the ambiguity of the roll and pitch angles by using the positional relationship between the sky-component and the horizon line. Through engineering optimization, our proposed method runs at approximately 60 fps on the fly.

To verify our algorithm, we designed a fixed-wing UAV system and flew with various lighting conditions, terrains, speeds, attitudes, and altitudes. Extensive experimental results illustrate the robustness of the presented algorithm. The comparison results show that the angles produced by our method are almost consistent with the reference values by the IMU, which demonstrates the effectiveness of the proposed algorithm.

Although we have demonstrated the validity of the proposed approach, however, we did not tackle some difficult situations, such as in the case of dark, obstacles, and challenging nonlinear scenes. Thus, in future work, we will further improve the adaptivity and precision of the developed method. Moreover, we will integrate the horizon detection method with the SLAMs to accomplish a more reliable autonomous navigation system.

References

[1] S. Park, D. H. Won, M. S. Kang, T. J. Kim, H. G. Lee, and S. J. Kwon, "RIC (robust internal-loop compensator) based flight control of a quadrotor type UAV," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2005, pp. 3542–3547.

[2] J. Li and L. Xiao-Min, "Vision-based navigation and obstacle detection for UAV," in *Proc. Int. Conf. Electron., Commun. Control*, 2011, pp. 1771–1774.

[3] B. N. Sepehri, S. A. Etemad, and A. Whitehead, "Robust horizon detection using segmentation for UAV applications," in *Proc. 9th Conf. Comput. Robot Vis.*, 2012, pp. 346–352.

[4] R. Jan, "Accelerometers and an aircraft attitude evaluation," in *Proc. IEEE Sensors Conf.*, Nov. 2005, pp. 784–789.

[5] S. Thurrowgood, R. J. D. Moore, and D. Bland, "UAV attitude control using the visual horizon," in *Proc. Australia Conf. Robot., Autom.*, Dec. 2010, pp. 1–8.

[6] E. Khoramshahi, R. A. Oliveira, N. Koivumki, and E. Honkavaara, "An image-based real-time georeferencing scheme for a UAV based on a new angular parametrization," *Remote Sens.*, vol. 12, no. 19, p. 3185, 2020.

[7] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[8] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. Robot.*, vol. 33, no. 2, pp. 249–265, Apr. 2017.

[9] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM," *IEEE Trans. Robot.*, to be published, doi: 10.1109/TRO.2021.3075644.

[10] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[11] S. Timotheatos, S. Piperakis, A. Argyros, and P. Trahanias, "Vision based horizon detection for UAV navigation," in *Advances in Service and Industrial Robotics, RAAD 2018. Mechanisms and Machine Science*, vol. 67, N. Aspragathos, P. Koustoumpardis, and V. Moulianitis Eds., Cham, Switzerland: Springer, pp. 181–189.

[12] T. Ahmad, G. Bebis, E. Regentova, and A. Nefian, "A machine learning approach to horizon line detection using local features," in *Advances in Visual Computing, ISVC 2013*, vol. 8033, G. Bebis *et al.*, Eds., Berlin, Heidelberg: Springer, pp. 181–193.

[13] T. Ahmad, G. Bebis, M. Nicolescu, A. Nefian, and T. Fong, "Fusion of edge-less and edge-based approaches for horizon line detection," in *Proc. 6th Int. Conf. Inf., Intell., Syst. Appl.* , 2015, pp. 1–6.

[14] D. K. Prasad, D. Rajan, C. K. Prasath, L. Rachmawati, E. Rajabally, and C. Quek, "MSCM-LiFe: Multi-scale cross modal linear feature for horizon detection in maritime images," in *Proc. IEEE Region Conf. TENCON*, Nov. 2016, pp. 1366–1370.

[15] D. Liang and Y. Liang, "Horizon detection from electro-optical sensors under maritime environment," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 1, pp. 45–53, Jan. 2020.

[16] X. Zou, W. Zhan, C. Xiao, C. Zhou, Q. Chen, T. Yang, and X. Liu, "A novel vision-based towing angle estimation for maritime towing operations," *J. Mar. Sci. Eng.*, vol. 8, no. 5, pp. 356, 2020.

[17] S. Fefilatyev, V. Smarodzinava, L. O. Hall, and D. B. Goldgof, "Horizon detection using machine learning techniques," in *Proc. 5th Int. Conf. Mach. Learn. Appl.*, 2006, pp. 17–21.

[18] T. G. McGee, R. Sengupta, and K. Hedrick, "Obstacle detection for small autonomous aircraft using sky segmentation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2005, pp. 4679–4684.

[19] N. S. Boroujeni, S. A. Etemad, and A. Whitehead, "Robust horizon detection using segmentation for UAV applications," in *Proc. 9th Conf. Comput. Robot Vis.*, 2012, pp. 346–352.

[20] Y. Sun and L. Fu, "Coarse-Fine-Stitched: A robust maritime horizon line detection method for unmanned surface vehicle applications," *Sensors*, vol. 18, no. 9, 2018, Art. no. 2825, doi: 10.3390/s18092825.

[21] W. Zhan *et al.*, "Autonomous visual perception for unmanned surface vehicle navigation in an unknown environment," *Sensors*, vol. 19, 2019, Art. no. 2216, doi: 10.3390/s19102216.

[22] W. Zhan *et al.*, "Adaptive semantic segmentation for unmanned surface vehicle navigation," *Electronics*, vol. 9, 2020, Art. no. 213, doi: 10.3390/electronics9020213.

[23] S. Workman, M. Zhai, and N. Jacobs, "Horizon lines in the wild," in *Proc. Brit. Mach. Vis. Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 19–22.

[24] L. Porzi, S. R. Bulò, and E. Ricci, "A deeply-supervised deconvolutional network for horizon line detection," in *Proc. Proc. 24th ACM Int. Conf. Multimedia*, Oct. 2016, pp. 137–141.

[25] M. Zhai, S. Workman and N. Jacobs, "Detecting vanishing points using global image context in a non-Manhattan world," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5657–5665.

[26] C. Y. Jeong, H. S. Yang, and K. D. Moon, "Horizon detection in maritime images using scene parsing network," *Electron. Lett.*, vol. 54, no. 12, pp. 760–762, 2018.

[27] A. Carrio, H. Bavle, and P. Campoy, "Attitude estimation using horizon detection in thermal images," *Int. J. Micro Air Veh.*, vol. 10, no. 4, pp. 352–361, 2018.

[28] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vis.*, vol. 59, pp. 167–181, 2004,.

[29] R. Grompone von Gioi, J. Jakubowicz, J. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 4, pp. 722–732, Apr. 2010, doi: 10.1109/TPAMI.2008.300.

[30] Imu_Utils, Accessed: Jan. 5, 2021. [Online]. Available: https://github.com/gaowenliang/imu_utils

[31] Visual-Inertial Calibration Toolbox Kalibr, Accessed: Jan. 5, 2021. [Online]. Available: https://github.com/ethz-asl/kalibr

Yong Xu received the B.S. degree in space engineering and the M.S. degree in aerospace science and technology from the National University of Defense Technology, Changsha, China, in 2011 and 2013, respectively.

He is currently a Research Assistant with the Aerospace Technology Institute of CARDC, Sichuan, China. His research interests include vision-based navigation, state estimation for robotics, deep learning, image target detection and tracking.

Hongtao Yan received the master's degree in aerospace technology from the National University of Defense Technology, Changsha, China, in 2017.

He was with the China Aerodynamics Research and Development Center, Mianyang, China, in February 2017. He is currently an Engineer with the Aerospace Technology Institute of CARDC, Sichuan, China. His research interests include UAV guidance and control.

Yue Ma received the B.S., M.S., and Ph.D. degrees in information and communication engineering from the National University of Defense Technology University, Changsha, China, in 2010, 2013, and 2017 respectively.

He is currently an Engineer with China Aerodynamics Research and Development Center, Sichuan, China. His interests include UAV swarm system, cooperative communication, and signal processing.

Pengyu Guo received the master of computer science and technology, and doctor of aerospace science and technology degrees from the National University of Defense Technology, Changsha, China, in 2010 and 2015, respectively.

He has experience working for China Xi'an Satellite Control Center and is currently an Associate Research Fellow with National Innovation Institute of Defense Technology, Academy of Military Science. His current research interests include devise algorithms based on computer vision and machine learning to enable unmanned platform's imaging systems for detection, tracking, recognition and relative pose estimation.