Image-Only Real-Time Incremental UAV Image Mosaic for Multi-Strip Flight

Fangbing Zhang, *Student Member, IEEE*, Tao Yang, *Member, IEEE*, LinFeng Liu, *Student Member, IEEE*, Bang Liang, *Student Member, IEEE*, Yi Bai, *Student Member, IEEE*, and Jing Li, *Member, IEEE*

***Abstract*—Limited by aircraft flight altitude and camera parameters, it is necessary to obtain wide-angle panoramas quickly by stitching aerial images, which is helpful in rapid disaster investigation, recovery after earthquakes, and aerial reconnaissance. However, most existing stitching algorithms do not simultaneously meet practical real-time, robustness, and accuracy requirements, especially in the case of a long-distance multistrip flight. In this paper, we propose a novel image-only real-time UAV image mosaic framework for long-distance multistrip flights that does not require any auxiliary information, such as GPS or GCPs. The framework has a complete structure, mainly consisting of the three tasks of automatic initialization, current frame tracking, and real-time mosaic generation. The stitching plane is determined in the initialization process, the homography transformation of the current image is estimated in the tracking task, and the image is mapped to the stitching plane to generate and update the panorama in the real-time mosaic process. The core idea is that, in the tracking task, we introduce and develop a keyframe insertion strategy to generate a keyframe list and, on this basis, design a homography matrix estimation based on a local optimization strategy to reduce the accumulated error when continuously stitching image sequences collected online by UAVs and to realize real-time, effective UAV image mosaic construction. In addition, this framework has good scalability, which is not limited to a specific algorithm. To evaluate the effectiveness of the proposed framework, we carry out a large number of experiments on the AirSim simulation platform and present an exhaustive evaluation in some sequences from a popular dataset. Qualitative and quantitative experimental results in simulation and real environments demonstrate that our algorithm can obtain an effective and robust mosaic image in real-time. Through strategy comparison experiments, it is proven that the keyframe insertion strategy and the local optimization strategy both improve the stitching performance. Compared with five state-of-art image stitching approaches, the mosaic effect of the proposed method is comparable or better. In terms of algorithm speed, its performance is superior to them. Additionally, experiments of illumination change and feature replacement in the framework verify the good adaptability and scalability of the algorithm.**

***Index Terms*—UAV images, image mosaic, local optimization, homography estimation.**

I. Introduction

As SOON as unmanned aerial vehicles (UAVs) were developed, they attracted widespread attention in many countries [1]–[3] and have since been widely used in the military field (e.g., for aerial reconnaissance and border patrol), as well as in civilian fields (e.g., for mapping, disaster monitoring, and traffic monitoring) [4]. Due to the limitations of flight altitude and camera focal length, the images acquired by UAVs have many undesirable characteristics, such as large numbers, a small image range, a high overlap degree, and multiple strips. To obtain more comprehensive information and expand the scope of the field of view in many specific tasks, it is very important to obtain a full panoramic view using fast image mosaic techniques [5].

Image mosaic technology has become a research hotspot, which has been widely used in multimedia applications where the viewing device has a small field of view but the desired view is large [6]–[9], such as in remote sensing image processing [10], [11], virtual reality [12], [13], and video surveillance [14]–[16]. Many software tools exist on the market for users to create panoramas, such as Autopano [17] and Panorama Photo Stitcher [18]. In general, image mosaic approaches are classified into four categories, according to the registration strategy. The first is based on pixel intensities or colors [19], [20]; this method is simple but has poor noise resistance and low stitching efficiency. The second is based on using the transform domain [21], which has good antinoise ability, but the number of calculations required for stitching is large and the quality of stitching is poor when the image has a change in view angle or zoom. The third is based on features [22], [23], for which algorithms are typically invariant to image scaling, translation, and rotation and have high stitching efficiency, robustness, and accuracy. Traditional feature extraction algorithms are Harris [24], SIFT(Scale-invariant feature transform) [25], and ORB(Oriented fast and rotated brief) [26]. The last is based on deep learning. Many feature extraction- and matching method-based deep learning algorithms [27], [28] have been

Manuscript received October 2, 2019; revised February 22, 2020 and April 25, 2020; accepted May 12, 2020. Date of publication May 28, 2020; date of current version April 23, 2021. This work was supported by the National Natural Science Foundation of China under Grants 61672429 and 61502364. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Chang-Su Kim. *(Corresponding author: Tao Yang.)*

Fangbing Zhang, Tao Yang, LinFeng Liu, Bang Liang, and Yi Bai are with the School of Computer Science, National Engineering Laboratory for Integrated Aero-Space-Ground-Ocean Big Data Application Technology, SAIIP, Northwestern Polytechnical University, Xi'an 710129, China (e-mail: fangbing_zhang@mail.nwpu.edu.cn; tyang@nwpu.edu.cn; linfengliu@mail.nwpu.edu.cn; knl000b@mail.nwpu.edu.cn; seafire@mail.nwpu.edu.cn).

Jing Li is with the School of Telecommunications Engineering, Xidian University, Xi'an 710071, China (e-mail: jinglixd@mail.xidian.edu.cn).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author. The material includes a video demo to show the experimental results in detail. This material is 24.1 MB in size.

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2020.2997193

developed to achieve image registration. Yi *et al.* [29] presented a novel deep network architecture that implements a full feature point handling pipeline, that is, detection, orientation estimation, and feature description. Tian *et al.* [30] proposed a high-performance descriptor based on neural networks, which performs with high efficiency on multiple datasets. SuperPoint [31] uses a self-supervised network to extract features and calculate descriptors, which has advantages in pose calculation. Luo *et al.* [32] proposed a unified framework based on deep learning that expands the details of existing local feature descriptors by summarizing contextual information. In addition, some researchers have attempted to use neural networks to directly learn geometric transformations to align two images. DeTone *et al.* [33] proposed Regression HomographyNet, a VGG-style model that learns the homography relating two images. Nguyen *et al.* [34] proposed an unsupervised learning algorithm that trains a deep convolutional neural network to estimate planar homographies.

Although image mosaic technology has emerged in recent years [15], [35], [36], these methods do not simultaneously meet the real-time, robustness, and accuracy requirements of UAV image mosaics. First, UAV images have the characteristics of a large quantity of data, small phase amplitude, and higher overlapping degree. The large amount of data in a UAV video image leads to an unbearable time cost in the mosaic procedure. Second, UAVs are typically small-sized and have poor self-stability and wind resistance. Although they are equipped with an autopilot and a stabilization gyroscope, there will inevitably be some tilting in the process of taking photos. The camera also has geometric distortion induced by the lens, so images taken by UAVs have large affine distortion compared with the real scene. There have been many proposed algorithms for constructing UAV image mosaics [37]. For example, some mosaic algorithms based on the idea of structure-from-motion (SFM) have been proposed. Wang *et al.* [38] presented an automated image mosaicing method based on SFM without ground control points. This method can not only obtain a panoramic image of a wide area but also the corresponding three-dimensional terrain model. Zhou *et al.* [39] utilized the SFM method to estimate the relative camera poses and stitch multiple UAV images into one seamless image. The study [40] validated that the SFM method in UAV image mosaic construction has low accuracy in geographic positioning due to the low-precision POS. However, area measurement is typically highly accurate, which meets the requirements of agricultural science. Generally, stitching algorithms based on SFM can generate accurate orthoimages, but SFM methods typically take hours to generate the final orthoimage, which is not suitable for real-time and incremental use. Several mosaic algorithms based on simultaneous localization and mapping (SLAM) have been developed. Bu *et al.* [41] presented a real-time approach to stitch large-scale aerial images incrementally. This method uses a monocular SLAM system to generate a 3D point cloud map and can output high-quality mosaics in most circumstances. However, this method requires GPS data to optimize the stitching effect, and the mosaic image quality may decrease for nonplanar environments.

Some 2D methods have been proposed recently for image mosaic construction. Ge *et al.* [42] proposed a fast mosaicking method for an image sequence acquired by a small UAV. This method applies bundle adjustment with a few ground control points (GCPs) to estimate the homography transformation between images and ground, which is used to orthorectify and merge the images together to generate a mosaic of the whole scene. Liu *et al.* [43] developed a new adjustment model for mosaicing based on homographic transformation, which is especially suitable for low-overlap images with a long focal length captured by a medium-altitude UAV, and can effectively eliminate misalignments in the seam lines. Avola *et al.* [44] presented a small-scale UAV-based system for real-time incremental generation of georeferenced mosaics at low altitudes. The system uses an ROI to speed up the stitching stage and uses a rigid transformation to build a mosaic at low altitude, partially mitigating the artifacts due to the parallax error. Zhang *et al.* [45] proposed an improved dynamic programming algorithm to search for the optimal seam lines to complete a UAV image mosaic, which adopted an adaptive energy aggregation and traversal strategy to find a more ideal splicing path for adjacent UAV images and avoid ground objects better. The paper [46] presented an mgraph approach to generate an incremental mosaic in real time from a UAV swarm. The algorithm is based on the generation and fusion of multiple submosaics represented by homography graphs, taking advantage of parallel processing and UAV metadata to generate a georeferenced mosaic over a geographic information system. Ye *et al.* [47] proposed an aerial panorama generation system using camera-equipped drones, which can generate aerial panoramas in a short time with appropriate aspect ratios and good visual quality, by measuring the pixel movement between consecutive frames. However, the aerial panorama generated by the system may have poor visual defects at turning points. Chen *et al.* [48] designed an image mosaic method based on robust feature matching and a new bundle adjustment strategy to make the mosaic system suitable for the UAV image panoramic mosaic effect.

Although considerable progress has been achieved in this area to date, a fast, robust, and efficient aerial image mosaic system in an unknown environment is still worthwhile to study, especially in long-distance multistrip flight conditions. As shown in Fig. 1, a drone often scans an entire scene through a multistrip flight to obtain images that participate in the stitching, and the flight distance is generally long. This paper proposes a novel image-only real-time UAV image mosaic framework, which does not require any auxiliary information, such as GPS or GCPs. The main contributions are as follows:

- First, we propose a novel image-only UAV image mosaic framework for a long-distance multistrip flight that operates in real time. The framework has a complete structure comprised of three tasks: automatic initialization, current frame tracking, and real-time mosaic generation. The initialization process is used to determine the stitching plane. In the tracking task, the transformation of the current image is estimated through keyframe insertion and local optimization. The panorama is generated and updated in the real-time mosaic process. In addition, this framework has good scalability, which is not limited to a specific algorithm, and provides a new research idea for aerial image mosaic technology.

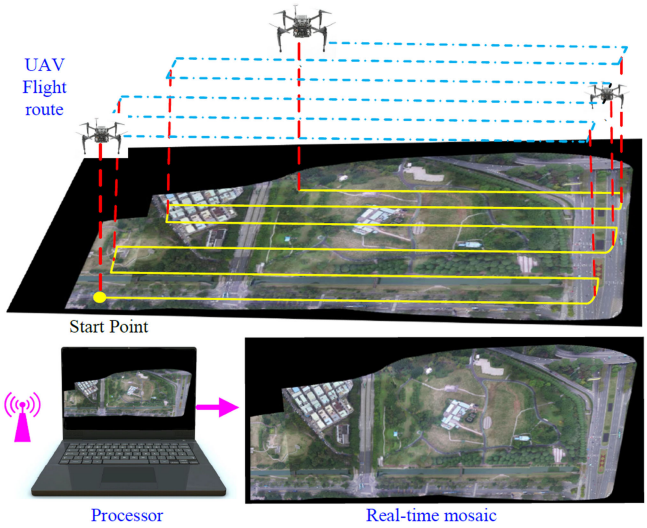

Fig. 1. Demonstration that our system can realize real-time mosaic construction in long-distance flight using only a single image. Obviously, when the UAV flight route is long and there are many multistrip flights, it is easy for accumulated stitching errors to occur. Our system framework can reduce these cumulative errors, to a certain extent, and obtain a better mosaic in real-time.

- Second, in the current frame tracking task, we introduce and develop a keyframe insertion strategy. The keyframe list is constructed by setting the insertion conditions and calculating the keyframe feature points, homography matrix, and mapping positions of vertices. On this basis, we design a local optimization strategy to estimate the homography matrix. In this strategy, we quickly determine the effective keyframes matching the current frame by using the keyframe list to search for the keyframes that overlap the previous frame. The remapping error of the matching feature points between the effective keyframe and the current frame is used as the error term to optimize the homography matrix in this step. These two strategies can effectively reduce the error accumulated when continuously stitching image sequences collected online by UAV, as well as improving the validity and robustness of stitching algorithm.
- Finally, we conduct many experiments in simulation and real environments. For the simulation, we build a simulation environment on the AirSim platform and verify the effectiveness and robustness of the algorithm under different flight conditions, especially in a long-distance multistrip flight. For the real experiments, we choose some sequences from popular datasets to further validate that the algorithm can obtain effective and robust panoramas in various environments. Moreover, we carry out comparison experiments to prove that the keyframe insertion strategy and local optimization strategy can improve the stitching performance. Comparison with five state-of-the-art stitching approaches demonstrates that the algorithm is comparable, or even better, in terms of stitching effect and speed. Additionally, experiments of illumination change and feature replacement in the framework verify the good adaptability and scalability of the algorithm.

II. The Proposed Method

The proposed image-only real-time UAV image mosaic framework is illustrated in Fig. 2. From the figure, we can see that when the framework operates, there is a preparation step to extract and describe the features of the images captured by the UAV for image registration. This paper adopts the CUDA-SIFT algorithm with GPU acceleration to allow real-time performance and provide good invariance in terms of image rotation, image scaling, and illumination changes. The framework is mainly divided into three parts: automatic initialization, current frame tracking, and real-time mosaic generation. For the first frame, the system automatically determines the mapping plane and initializes the keyframe and keyframe list. When processing subsequent images, the current frame tracking process, including keyframe insertion and homography matrix estimation based on local optimization, is responsible for solving the homography mapping matrix H between current frame and mapping plane. At the same time, the image is mapped to the stitching plane by its matrix H to generate the mosaic in real-time. These three parts are described in detail in the following subsections.

A. Preparation and Automatic Initialization

Considering that UAV images have the characteristics of high coincidence and a large quantity of data, feature-based registration methods are more suitable. From the two aspects of speed and performance, the system utilizes the CUDA-SIFT algorithm to extract and describe the features of the images in preparation, which are used for image registration. The CUDA-SIFT code used is from the fourth version of the SIFT implementation using CUDA for Nvidia GPUs. This version has been optimized for the Kepler and later generations of GPUs. One advantage is that the algorithm speed is fast: on a GTX 1080 GPU, the code requires approximately 1.7 ms on a 1920×1080 pixel image, and brute-force matching of features requires approximately 2.2 ms for two sets of approximately 1,900 SIFT features each. Another advantage is that SIFT features have good invariance to image translation, rotation, scale changes, and brightness changes. Therefore, when the brightness of the image to be stitched changes, CUDA-SIFT can still extract and match the SIFT feature points stably, ensuring that the stitching algorithm can work normally and avoid the preprocessing of image brightness. The extracted feature points in the preparation are shown in Fig. 2. While CUDA-SIFT is used in our current system, the proposed framework is not limited to this feature point. In fact, this feature can be replaced by other features in the proposed framework, such as deep learning features.

When the first image is captured by the UAV, the system first carries out an automatic initialization process. In this process, we identify the first frame as the stitching plane; subsequent images are mapped to this plane. In the process of UAV flight, real-time acquired aerial images cannot always be registered with the first frame. At this time, a pairwise matching strategy can be used for image mosaic generation, but the cumulative error will be very large. Referring to the keyframe strategy in SLAM, our system appropriately uses some images as keyframes in the stitching process, and the feature points, mapping matrix, and mapping

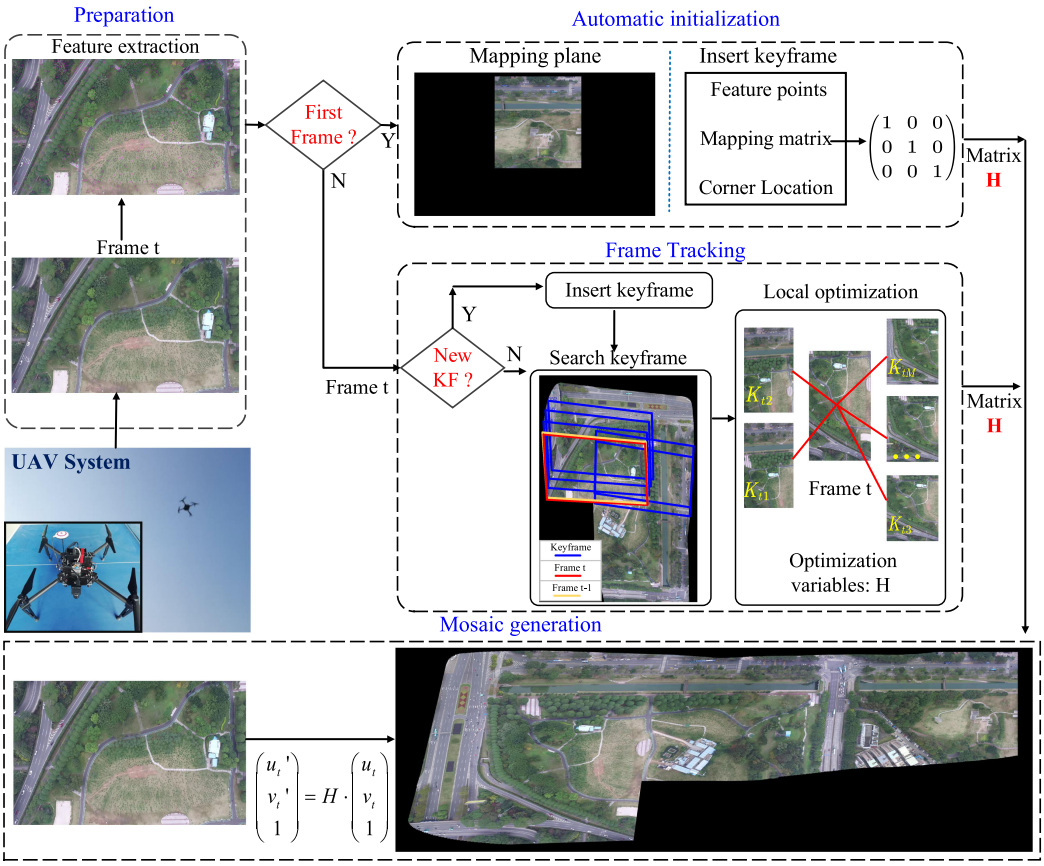

Fig. 2. An overview of the proposed image-only real-time incremental UAV image mosaic framework for long-distance multistrip UAV flight. The framework realizes UAV image mosaic construction by estimating the homography transformation between images. The framework mainly includes three tasks: automatic initialization, current frame tracking, and real-time mosaic generation.

location of the four image corners are saved, as shown in Fig. 2. The mapping matrices of subsequent images are obtained by registration with keyframes, which can reduce the cumulative error, to a certain extent. In the initialization process, we insert the first image into the list as the first keyframe, and its mapping matrix is a unit matrix, as shown in Fig. 2.

B. Current Frame Tracking

As seen in Fig. 2, after the automatic initialization is completed, the framework needs to track the currently acquired aerial image. The tracking process is responsible for solving the mapping matrix of the current frame to locate this image on the stitching plane and deciding when to insert a new keyframe into the list, as shown in Fig. 3.

1) Keyframe Insertion: In the framework, we use some images as keyframes in the stitching process, and the keyframe insertion strategy is very important. If the keyframes are sparse, frame tracking failure will likely result; if the keyframes are very dense, although tracking basically will not fail, it will lead to a rapid increase in information redundancy and computational complexity. The extent to which this redundant information will improve the accuracy of the system is very limited, or will not even be improved while consuming more computing resources.

In this framework, new keyframes are created with a generous policy to ensure the success of tracking and to reduce the cumulative error of the pairwise matching strategy for mosaic construction. Meanwhile, the insertion of these new keyframes can properly reduce the information redundancy and the loss of computational resources. More specifically, as shown in Fig. 3, the keyframe insertion conditions of the framework include two aspects. One is that the number of interior points (N) matching the current frame t and the latest keyframe K_t is less than a certain value, $N1$, and the other is that the proportion of the overlap area (P) between the current frame and the latest keyframe is greater than a threshold $P1$. In our system, $N1 = 300$ and $P1 = 0.75$. When both conditions are satisfied, we insert the frame $t-1$ into the keyframe list as the keyframe K_{t+1} and store the feature points, mapping matrix, and the mapping location of the four corners of this new keyframe.

2) Local Optimization: To map the current frame to the stitching plane, the homography mapping matrix H can be calculated directly by matching the current frame with the latest keyframe. Although this strategy is simple, there are cumulative errors that lead to worse stitching results in later stages. In this part, we propose a local optimization strategy to estimate the homography matrix H of the current frame to partially eliminate the accumulated error and improve the validity and robustness of stitching.

In this strategy, we first need to quickly determine the effective keyframes that match the current frame t. It generally requires considerable time to match the current frame with all

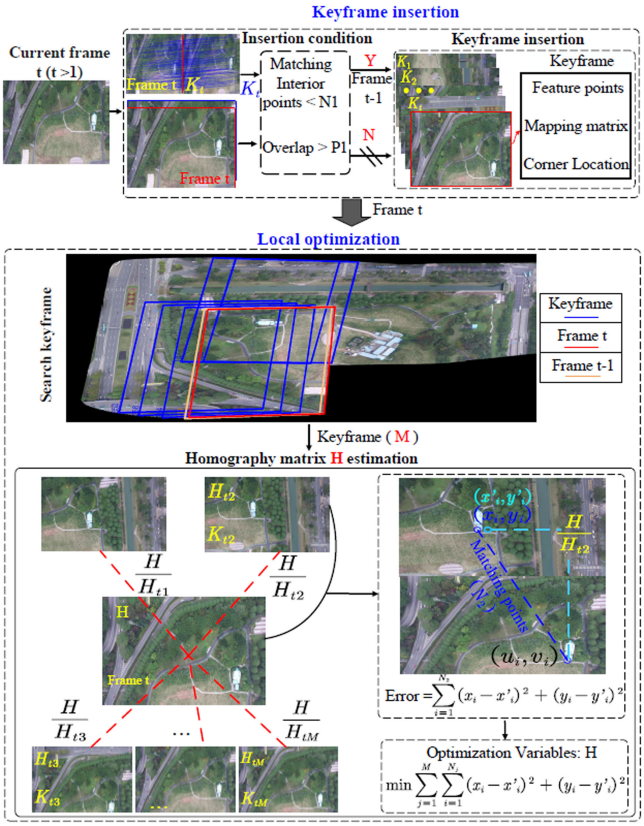

Fig. 3. The tracking process of the current frame. The framework solves the homography mapping matrix H of the current frame to locate the image on the stitching plane in the tracking process, which includes two sections: keyframe insertion and local optimization.

the keyframes pairwise and select the effective keyframes from them. Moreover, with the increase in UAV flight time and number of keyframes, the search time will increase rapidly. Considering that the moving distance between the current frame t and the previous frame $t-1$ is limited when a UAV collects images in real time, we can easily and quickly find keyframes overlapping with frame $t-1$ on the stitching plane. In theory, these keyframes overlap with the current frame t and can match each other. In this framework, we define the proportion of overlapping areas of a frame $t-1$ and a keyframe K_i as

$$Overlap = \frac{area(t-1) \cap area(K_i)}{area(t-1)}, \tag{1}$$

where the function $area(t-1)$ represents the area of frame t in the stitching plane and $area(K_i)$ represents the area of keyframe K_i in the stitching plane. When $Overlap > 0.3$, we consider frame t and keyframe K_i as likely to match. Thus, not only can the effective keyframes be quickly found but also the error matching caused by scene similarity can be reduced, to a certain extent, by utilizing the temporal information and spatial location information of the image. The search results are shown in Fig. 3, where the orange box represents the position of the previous $t-1$ and the blue box represents the keyframes found.

Then, we match the current frame t with the found keyframes. Assume that there are M effective keyframes, $K_{t1}, K_{t2}, \ldots, K_{tM}$, and their corresponding respective homography matrices are $H_{t1}, H_{t2}, \ldots, H_{tM}$. To optimize the homography matrix H of frame t, we set it as an optimization variable. The homography matrix between the current frame t and the keyframe K_{tj} can be expressed as $\frac{H}{H_{tj}}$. Assume that the number of matching points between the current frame t and the keyframe K_{tj} is N_j, $j = 1, 2, \ldots, M$. Moreover, the matching point of pixel (u_i, v_i) in frame t is (x_i, y_i) of keyframe K_{tj}, and $i = 1, 2, \ldots, N_j$. Using the homography matrix $\frac{H}{H_{tj}}$ between the current frame t and the keyframe K_{tj}, we know that the pixel (x_i', y_i') of the remapping of pixel (u_i, v_i) onto the keyframe K_{tj} satisfies the following formula:

$$\begin{pmatrix} x_i' \\ y_i' \\ 1 \end{pmatrix} = \frac{H}{H_{tj}} \times \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix}. \tag{2}$$

We define $(x_i - x_i')^2 + (y_i - y_i')^2$ as a remapping error of pixel (u_i, v_i). Then, the remapping error of all matching points between the keyframe K_{tj} and the current frame t is

$$\sum_{i=1}^{N_j} ((x_i - x_i')^2 + (y_i - y_i')^2). \tag{3}$$

We use the remapping error of the matching feature points between the effective keyframes $K_{t1}, K_{t2}, \ldots, K_{tM}$ and the current frame t as the error term to optimize the homography matrix, as shown in Fig. 3. The optimization problem can be expressed as follows:

$$\min \sum_{j=1}^{M} \sum_{i=1}^{N_j} ((x_i - x_i')^2 + (y_i - y_i')^2). \tag{4}$$

The framework uses the Ceres solver to solve this optimization problem quickly and obtain the homography matrix H of the current frame t. This optimization strategy not only utilizes the matching information with the latest keyframe, but also the information of other matching keyframes. For example, from the keyframes found in Fig. 3, in addition to the time-approaching keyframes (top), there are also some position-approaching keyframes (right). This strategy utilizes these keyframes to optimize the matrix H, to ensure the accuracy and robustness of the current frame in the local stitching and to reduce the cumulative error caused by relying only on information of the latest keyframe. The effect of this strategy is particularly evident when the UAV carries out a multistrip flight.

C. Real-Time Mosaic Generation

When a UAV flies at high altitude, the scene in the aerial image is considered planar (or approximately planar). Therefore, aerial images obtained from different perspectives in the same scene can be mapped from one image to another through a homographic transformation model. A homography matrix is a 3×3 homogeneous matrix with eight unknowns. The framework selects the first frame as the stitching plane, and subsequent images are mapped to this plane by the homography transformation. After obtaining the homography matrix H of the current frame t,

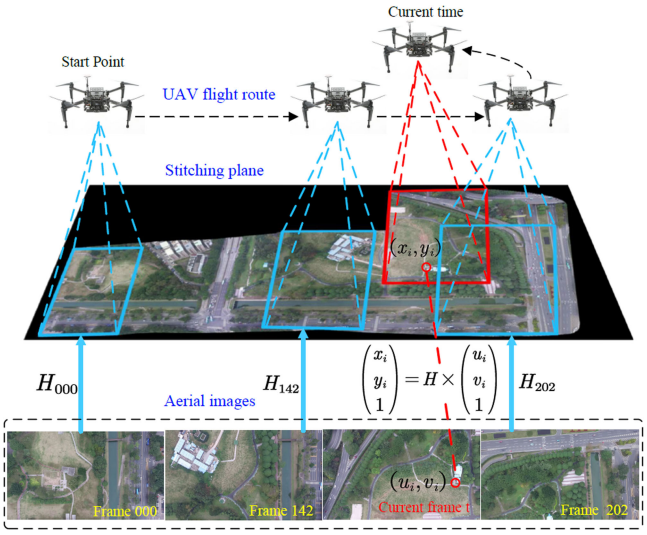

Fig. 4. Real-time mosaic generation process of the proposed framework. The framework selects the first frame as the stitching plane. When the UAV flies, this process uses the estimated matrix H to perform the homography transformation on the current frame and map it to the stitching plane.

we can quickly map the image to the stitching plane and obtain the current mosaic in real time. The real-time mosaic generation process is shown in Fig. 4.

More specifically, to generate mosaic images, we first define a canvas of fixed size $(W * W)$; W is set to 3,000 in this framework. Assuming that the pixel coordinates of the current frame t are $(u_1, v_1), (u_2, v_2) \ldots (u_i, v_i), \ldots, (u_N, v_N)$ and the corresponding homography matrix is H, mapping the current image to the stitching plane, the coordinates (u_i', v_i') of each pixel on the stitching plane satisfy:

$$\begin{pmatrix} u_i' \\ v_i' \\ 1 \end{pmatrix} = H \times \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{pmatrix} \times \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} \quad i = 1, \ldots, N. \tag{5}$$

Then, we draw the transformed image onto the canvas according to the coordinates (u_i', v_i') of the mapped pixels, as shown in Fig. 4. For the first frame, its homography matrix is a unit matrix. Therefore, the pixel coordinates should be unchanged after the homography transformation, and $u_i' = u_i, v_i' = v_i$. To facilitate the expansion of subsequent images onto the canvas, the first frame is placed in the center, which means that the coordinates of a pixel $(u_i', v_i'), i = 1, 2, \ldots, N$ on the canvas are $(u_i' + W/2, v_i' + W/2), i = 1, 2, \ldots N$. We can obtain the homography matrix H for frame t, through the tracking process and obtain the coordinates (u_i', v_i') of each pixel after the homography transformation. These pixel coordinates are relative to the first frame image coordinate system. Similarly, the mapped image is added onto the canvas, where the coordinates of a pixel $(u_i', v_i'), i = 1, 2, \ldots, N$ on the canvas are $(u_i' + W/2, v_i' + W/2), i = 1, 2, \ldots N$. Whenever the canvas is not large enough for a new image to be added, it expands outward automatically.

III. Experiments

In this section, to evaluate the performance of the real-time UAV image mosaic framework presented in this paper, we conduct a large number of simulations and real experiments. First, in the simulations, with the help of the AirSim platform, we simulate the process of aerial image mosaic construction in a real environment, and perform many experiments to evaluate the stitching performance under different flight conditions. Then, in real experiments, we present an exhaustive evaluation in some sequences from the most popular datasets. Thereafter, quantitative experiments and analyses are carried out under some image sequences of real and simulation environments to further evaluate the performance of the algorithm. Finally, the algorithm in this paper is compared with other state-of-art image stitching approaches in terms of both time and stitching performance. In addition, experiments of illumination change and feature replacement in the framework are conducted in both real environments and the simulation environment, which are used to verify the adaptability to the environment and scalability of the proposed algorithm.

A. Simulation Experiments

In the simulation, we utilize the AirSim platform to simulate the process of online image acquisition and real-time UAV image mosaic construction in a real environment. AirSim is a simulator for drones, cars, and more built on the Unreal Engine. It is open-source, cross-platform, and supports hardware-in-loop with popular flight controllers for physically and visually realistic simulation. Our simulation environment included a small town, a factory, and mountain racing tracks, as shown in Fig. 5(a). It can be seen from the figure that there is a town in the simulation environment, including houses, trees, cars, and other objects. Then, we added a UAV to the environment. The platform can provide us with various visual data, such as RGB images, depth maps, and color segmentation maps, which can be acquired in all directions of the UAV. As the proposed system only needs a monocular camera to acquire images, we selected RGB images as the input of the simulation system and set the image resolution to 640 × 480 at 10 fps. The input image is shown in Fig. 5(a). The components of the simulation system are shown in Fig. 5(b), including two computers and an ethernet crossover cable. One computer (the client) was used to build the simulation environment and output scene images. The other computer (the server) acted as a processor to implement the system and obtain the UAV mosaic image.

In this simulation, we designed five flight routes in different environments to test the performance of our system, including a straight line, a circle, Z-shaped, spiral, and multistrip flights. We conducted a quantitative analysis of the UAV image stitching results obtained in these five flight conditions; the experimental results are shown in Fig. 6. The first row in Fig. 6 displays the UAV flying in a straight line above the town and factory. In the

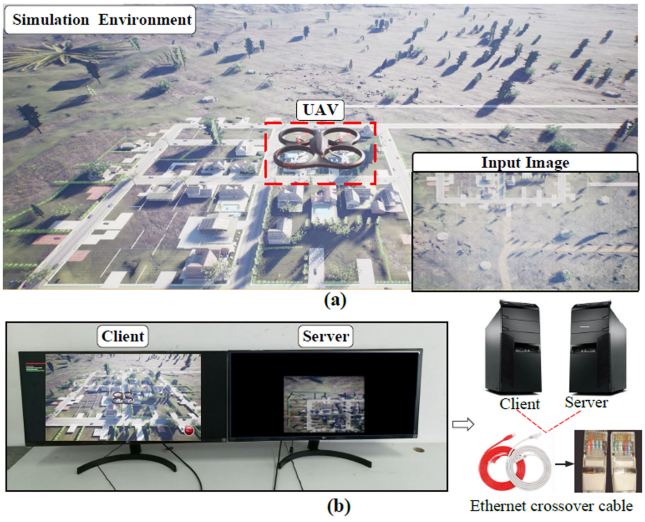

Fig. 5. Demonstration of the simulation platform. (**a**) The simulation environment, including several different scenarios and a UAV. Scene details include house, trees, cars, and other objects. (**b**) The components of the simulation system, including two computers and an ethernet crossover cable.

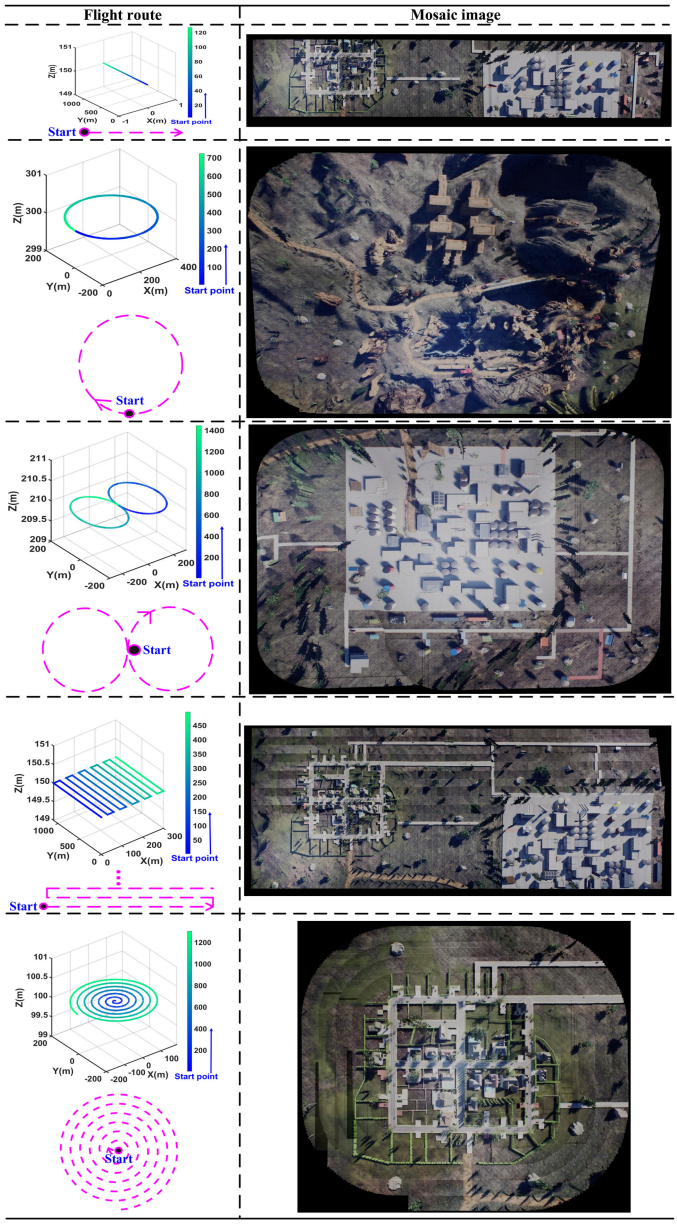

Fig. 6. UAV flight routes and stitching results under five flight conditions in different scenarios, including straight-line flight over a small town and a factory, circular flight over a mountain racing track, an 8-shaped flight over a factory, a spiral flight over a small town, and a multistrip flight over a town and a factory. In the flight route, dark blue indicates the starting point of flight. As the flight continues, the route becomes lighter and gradually turns dark green. The flight route on the mosaic is shown by the rose-red line.

flight route, the flying height of the drone was 100 m, and it flew more than 1,000 m in the y direction, collecting more than 100 aerial images. For the mosaic image, the UAV flew along a line from the town to the factory. Due to passing through some sections without buildings, such as near roads, there was less texture acquired in some of the aerial images. Additionally, the presence of some tall buildings caused shadows in the image. These factors increased the difficulty of stitching. In the system, we used the CUDA-SIFT algorithm to detect SIFT features, which has higher accuracy and more robust results than the ORB or SURF features. In addition, the selection of keyframes and the local optimization strategy reduced some accumulated stitching errors. We can see that the mosaic image obtained has little error and the stitching effect is good. For example, the road in the horizontal direction on the mosaic is relatively straight and there was basically no stitching misalignment.

From the second row in Fig. 6, we can see that the UAV flew 360 degrees above the mountain racing track along a circle with a radius of 200 m and collected more than 700 aerial images. As shown in the third row of Fig. 6, the drone flew above the factory along an 8-shaped path and collected more than 1,400 images. The mosaic images show the complexity of these two scenes. In the mountainous scene, there was a race track in the valley with a lake in the center surrounded by high and low mountains and plains. In the factory scene, there were mostly tall buildings, and there was considerable flat land with less texture and tall trees around the outside. Therefore, many of the aerial images collected have less texture and large shadows in these two scenarios, which affect the mosaic effect. The mosaic images of the two scenes are shown in Fig. 6. Overall, although the UAV flew along an arc and there were one or two closed loops, the proposed system generally obtained good stitching results, with basically no obvious stitching errors. For some small seam misalignments, it was generally possible to enhance the stitching effect by using further image fusion at the seam. In more detail, as the system assumes that the scene is planar and uses the homography transformation for image stitching, in places where there were large differences in height—such as where trees existed in the factory scene—the homography transformation estimation was prone to errors, resulting in image stitching errors. In addition, when there are large shadows in the image, the number of feature points is insufficient, and stitching errors may easily occur. For example, there is a large shadow in the

mountain scene (the upper right part of the mosaic image). In these cases, the proposed framework optimizes the homography matrix by a local optimization strategy, not only relying on the previous keyframe but also matching all the surrounding keyframes, which enhances the stitching robustness and reduces the cumulative error. In Fig. 6, it can be seen that although there are slight seam misalignments in these places, the stitching effect is acceptable. Moreover, as the images participating in the optimization are all the adjacent keyframes at the same geographical location, the stitching errors in these places have only a small influence on the mosaic effect of the subsequent images and the error is not directly accumulated. For instance, in the mosaic image of the factory scene, the stitching error at the bottom has little effect on the image stitching in the upper left. This is because when stitching the aerial images in the upper left, the keyframes around it include some early keyframes that are basically free of stitching errors. When using these images for local optimization, some accumulated errors can be eliminated, such that the stitching effect in this area is not significantly reduced.

From the fourth row of Fig. 6, we can see that the UAV's flight altitude was 150 m, and it flew above the town and factory. There are 11 strips, each with a distance of more than 1,000 m, and more than 450 images were collected. The fifth row of Fig. 6 shows the drone flying in a spiral above the town at a height of 100 m in six loops, with more than 1,200 images collected. These two routes were characterized by long-distance flights and many closed loops in the process. As seen from the fourth row mosaic image in Fig. 6, there are large plains with less texture between the town and the factory, and some trees above the ground, which pose some challenges for image stitching. The mosaic result shows that our system can obtain good stitching results when the UAV carries out long-distance multistrip flights. More specifically, in the horizontal flight direction, the stitching accuracy is high. There is basically no obvious stitching misalignment in the horizontal direction of the road. In the vertical direction, because there are some tall buildings and many trees in the factory scene, we can see some stitching misalignment in the mosaic image. However, these stitching errors are not directly accumulated in the proposed system. Instead, through the local optimization strategy, all surrounding keyframes are matched to optimize the homography matrix. For example, when the drone flew from the factory (right) to the town (left), the image homography matrix was solved by local optimization, and the stitching error was greatly reduced by using the early keyframes with small errors. The stitching error of the town on the left is smaller than that of the factory on the right. From the fifth row mosaic image in Fig. 6, we can see that when the UAV flew continuously along a spiral, the outer ring image needed not only an accurate mosaic with the adjacent frame but also a good mosaic with the adjacent inner ring image. In this case, the effectiveness and importance of the local optimization strategy can be particularly reflected. The local optimization strategy determines the images participating in the optimization by finding adjacent keyframes in the stitching plane, including the adjacent keyframe and the adjacent keyframes on the inner ring. The homography matrix was obtained by solving the optimization problem, which greatly improved the accuracy and robustness of image mapping to the stitching plane. It can be seen that under such complicated flight conditions, although there were some small stitching errors at the road with some trees, most places had a good stitching effect. This mosaic image covered the entire town, expanded the scope of monitoring, and facilitated the display of the entire town's landscape, housing distribution, traffic conditions, and other information.

TABLE I
PARAMETERS OF SEQUENCES FROM THE NPU DRONE-MAP DATASET

Sequence	Location	H-Max (m)	V-Max (m/s)	Length (km)
inspire1-rail	Luoyang, Henan	\	\	\
phantom3-highflower	Gucheng, Shaanxi	162.4	16.43	3.549
phantom3-huangqi-kfs	Hengdong, Hunan	222.3	16.57	6.945
phantom3-centralPark	Shenzhen, Guangdong	161.8	16.61	4.536

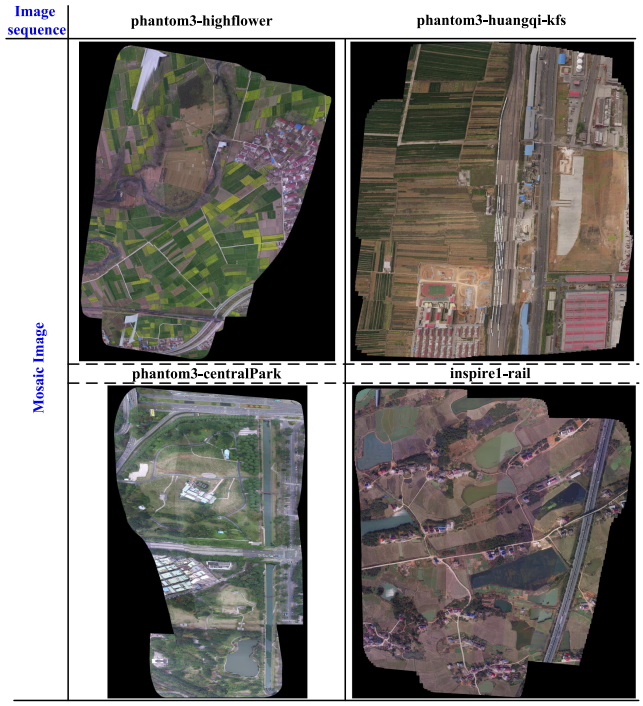

Fig. 7. Aerial image stitching results of the proposed system in the actual scene. The test dataset includes the image sequences phantom3-highflower, phantom3-huangqi-kfs, phantom3-centralPark, and inspire1-rail in the NPU Drone-Map dataset.

B. Real Experiments

In this section, we conduct experiments in real scenes to further validate the effectiveness and robustness of the proposed framework. The NPU Drone-Map dataset contains several aerial video sequences captured with different terrains and heights and has been used to evaluate aerial image stitching systems. We chose four aerial image sequences from this popular dataset to test the performance of the proposed framework. The parameters of the image sequences are shown in Table I. These four scenarios include city center parks, factories, fields in the north, and villages in the south. The mosaic images of these scenes are shown in Fig. 7.

From Fig. 7, we can see the stitching results of the sequences phantom3-huangqi-kfs and phantom3-highflower. Sequence phantom3-huangqi-kfs was collected by a drone flying over a factory in Luoyang, Henan. The mosaic image is shown in the first row (right) of Fig. 7. We can see that there are high buildings in many places in the scene, which results in the stitching accuracy at these places being affected. For example, there are some stitching misalignments near the railway in the figure. However, when subsequent images are stitched, the error is not directly accumulated but is corrected by the local optimization strategy, so the stitching error of subsequent images is likely to not increase but decrease. It can be seen that the roads in the figure (above the railway) are basically stitched correctly, and that the stitching error of the right side of the railway is gradually reduced. The first row (left) of Fig. 7 displays the mosaic image of the sequence phantom3-highflower, which was taken by a UAV over a field in Gucheng, Shaanxi. From Table I, the maximum flight altitude of the UAV was 162.4 m, its maximum speed was 16.43 m/s, and the flight distance was 3.549 km. There were more than four long-distance strips. From the stitching results of this sequence, it can be seen that a large area of the scene is a wheat field, and that the whole scene can be regarded as planar. We can see that, in this case, even when the drone continuously flies along four long-distance strips, the mosaic effect of the panorama obtained by our system is still good and there is almost no obvious stitching error; only in the vicinity of the riverside is there some stitching misalignment, which is due to less image texture in these places. Some places in the picture are slightly white, which was caused by the drone's wing when the drone captured the image.

The sequence inspire1-rail shows a village in Hengdong, Hunan; its mosaic image is shown in the second row (right) of Fig. 7. Table I shows that when the sequence was acquired, the UAV had a maximum flying height of 222.3 m and a total flight distance of 6.945 km. The flight distance was far, and there were more than 6 strips. It can be seen, from the mosaic, that this scene is of a typical southern village, including scattered houses, large fields, woods, small lakes, and intersecting paths. Due to the high flight altitude, the whole scene in the image can also be regarded as planar. Although the flight path was long, the overall effect of the mosaic image is good and it can be seen that the highway and the numerous intersecting paths have no substantial stitching misalignment. For some lakes with less texture, the stitching effect may be reduced, but the stitching errors are not directly accumulated and the framework mitigates their adverse effect on subsequent image stitching by the local optimization strategy. The last sequence is phantom3-centralPark, which was captured by a UAV above a city center park in Shenzhen, Guangdong. In this scenario, the maximum flight altitude of UAV was 161.8 m, and the length of the flight trajectory was 4.536 km, including four long-distance strips. The panorama of the city central park obtained by this system is shown in the second row (left) of Fig. 7. The stitching effect of this panorama is good. For example, many roads and a long river in the figure have no obvious stitching misalignment. The problems of minor misalignment and chromatic aberration at the stitching seam can be solved by further fusion. There are some stitching errors near the tall buildings, as these areas do not meet the planarity assumptions for the scene in the framework. However, the error is reduced due to the local optimization strategy, and it can be seen that the effect of the subsequent image stitching is not greatly affected. This panorama clearly shows the distribution of the entire park and the surrounding traffic conditions, greatly expanding the scope of monitoring.

TABLE II
QUANTITATIVE EVALUATION PARAMETERS IN DIFFERENT SCENES

Metric		Image Sequence		
		phantom3-highflower	inspire1-rail	spiral flight
Average stitching error	Position(pixels)	0.3159	0.1810	**0.1640**
	Angle(degrees)	0.0080	**0.0071**	0.0104
SSIM		0.9616	**0.9759**	0.9676
PSNR(dB)		32.2297	**39.4350**	36.1856
Cosine similarity		0.9991	**0.9999**	0.9996

In short, regardless of whether a simulation environment or real scene is considered, our experimental results have proven that the proposed framework can obtain effective and robust mosaic images under various flight conditions, whether it is a long-distance multiple strip flight or a continuous steering flight. Moreover, in the framework, the keyframe insertion and local optimization strategies can greatly reduce the stitching error and improve the overall stitching effect.

C. Quantitative Experiments

For quantitative analysis of the algorithm, we first took a large-resolution aerial image as a benchmark and simulated the image acquisition process when a UAV flies with multiple strips, dividing the original image into many small pieces with the same resolution. At the same time, we recorded the positions of the small images on the original image, including the image center coordinate and the central axis angle. In this way, we obtained the image sequence to be stitched and the corresponding position ground truth. Then, based on this sequence, image stitching was carried out, and the quantitative evaluation parameters were calculated. Image stitching accuracy was measured from the average position error and the average angle error. The average position error and the average angle error of the mosaic image were estimated by calculating the distance and angle difference between the mapping position of each small image and its true position. Additionally, we measured the cosine similarity, SSIM (structural similarity) and PSNR (peak signal to noise ratio) between the stitching image and the original image to further evaluate the stitching effect of the algorithm.

In this experiment, we took the mosaic image as the original large-resolution image and calculated the quantitative parameters of the proposed framework in three sequences in simulation and real environments, including sequence spiral flight, phantom3-highflower and inspire1-rail. The parameters are shown in Table II. As seen from Table II, this framework has robust and good stitching performance in different scenes. First, the average position error is as low as 0.1640 pixels, and the maximum is no more than 0.4. The average angle error is

as low as 0.0071 degrees, and the maximum is approximately 0.1 degrees. This means that the mapping position of the small image to be stitched is close to its true position, and the stitching accuracy of the algorithm is high. Then, for the cosine similarity and SSIM, the cosine similarity is as high as 0.9999, and the highest value of SSIM is 0.9759, which indicates that the mosaic image is very close to the original image in terms of brightness, contrast and structural similarity, and the stitching effect is good. Finally, for the PSNR, the larger the value, the smaller the image distortion. Generally, image quality between 30 dB and 40 dB is good. From the table, we can see that the PSNR is between 32 dB and 39 dB, which shows that the quality of the mosaic image is good and the stitching result is effective.

D. Comparative Experiments

The comparison experiment included two parts: comparison of strategies and comparisons with other stitching algorithms. In the first part, we test the mosaic performance for the keyframe insertion strategy and the local optimization strategy to prove the importance of these two strategies. In the second part, we chose the mature stitching software Autopano, state-of-art stitching algorithms Opencv-stitcher, Openpano, NISwGSP [49] and fast panorama stitching for high-quality-panoramic mages on mobile phones (HQPI) [6] for comparative experiments to verify the performance of the proposed system in terms of speed and stitching effects.

1) Comparison of Strategies: In this part, to prove that the keyframe insertion and local optimization strategy in this framework can improve the stitching performance, we compared this framework with the successive-stitching method without keyframe insertion and a stitching algorithm with keyframe insertion but no local optimization. The experimental results are shown in Figs. 8 and 9.

In the simulation environment, the selected two image sequences collected by the drone during a spiral flight over a small town and a multistrip flight over a town and a factory, as shown in Fig. 8. In these two sequences, the drone had a long flight distance and there were many closed loops in the process. It can be seen from the mosaic image in Fig. 8 that the successive-stitching method has a good stitching result initially, but as the flight distance increased, the cumulative error continued to increase, resulting in a worse stitching effect in the later image. For example, in the multistrip flight, the stitching error of the later image was large, and the mosaic effect was poor. For the stitching method of keyframe insertion, compared with the previous method, the stitching effect greatly improved. In this scheme, by inserting keyframes, the frequency of the cumulative error iteration was reduced and the stitching accumulation error was reduced. For example, in the spiral-flight sequence, the stitching result was good, and there were only some small stitching misalignments in the periphery of the mosaic image. Although this method can obviously reduce the accumulated stitching error, it cannot eliminate the error in essence and so the stitching error becomes larger and the stitching effect becomes worse as the flight distance increases. In the multistrip

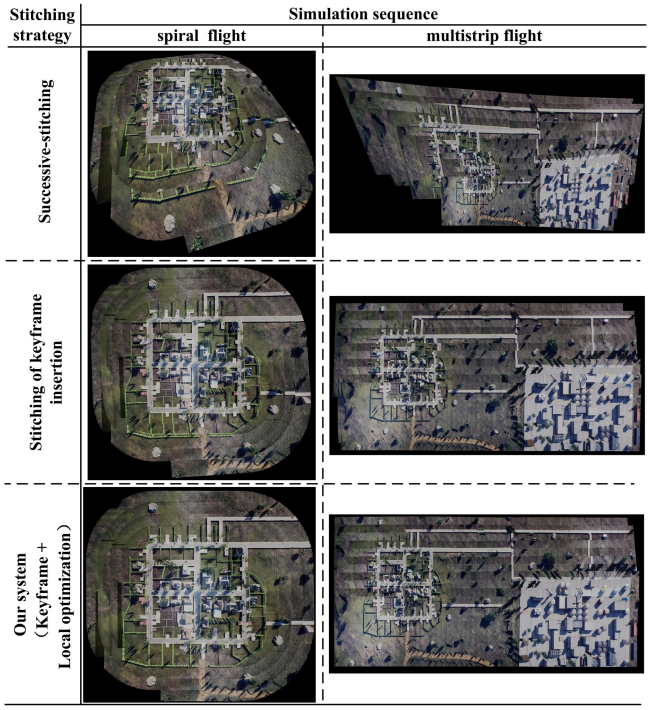

Fig. 8. Experimental results of strategy comparison in two simulation image sequences. Experimental scenarios include spiral flight over a small town and a multistrip flight over a town and a factory. In the experiment, we compared the proposed framework with the successive-stitching method without keyframe insertion and a stitching algorithm with keyframe insertion but no local optimization.

flight with a long flight distance, we can see that the stitching error became increasingly larger, and the stitching effect of the later image was poor, as shown in Fig. 8. For the proposed framework, a local optimization strategy was added, based on the second method. The stitching effect of the framework is the best in the figure. This strategy can mitigate cumulative errors by using the surrounding keyframes to optimize the homography matrix of the current frame, rather than just using the latest keyframe. Therefore, this strategy has a certain robustness for aerial image stitching, especially during long-distance flights, and can obtain good stitching results. In the spiral flight, it can be seen that not only was the stitching effect of the image center good, but there was basically no stitching error on the road in the periphery of the mosaic image. In the multistrip flight, although there were some stitching misalignments, compared with the above methods, the stitching effect was greatly improved.

The real experimental image sequences included phantom3-highflower and inspire1-rail from the NPU Drone-Map dataset, as shown in Fig. 9. In these two sequences, the flight distance was far, and there were more than 4 long-distance strips. From the mosaic image of Fig. 9, we can see that the proposed system had the best mosaic effect and the smallest mosaic error. In the case of a long-distance multistrip flight, the successive-stitching method had a large image mosaic error in the later stage due to the accumulation of errors. For instance, in phantom3-highflower, when the drone performed the fourth strip flight, the stitching error became increasingly obvious, as shown in the left part

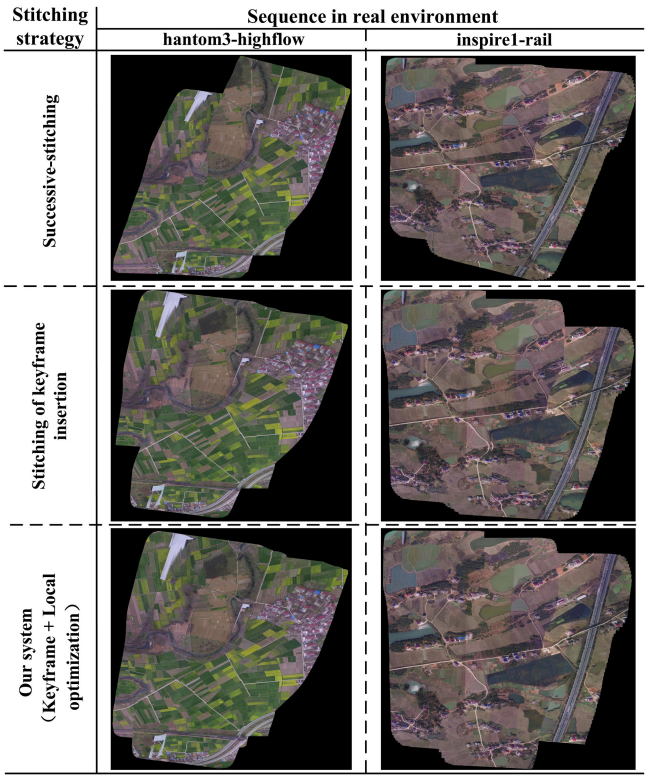

Fig. 9. Experimental results of strategy comparison in two real environmental image sequences. Experimental image sequences include phantom3-highflower and inspire1-rail from the NPU Drone-Map dataset.

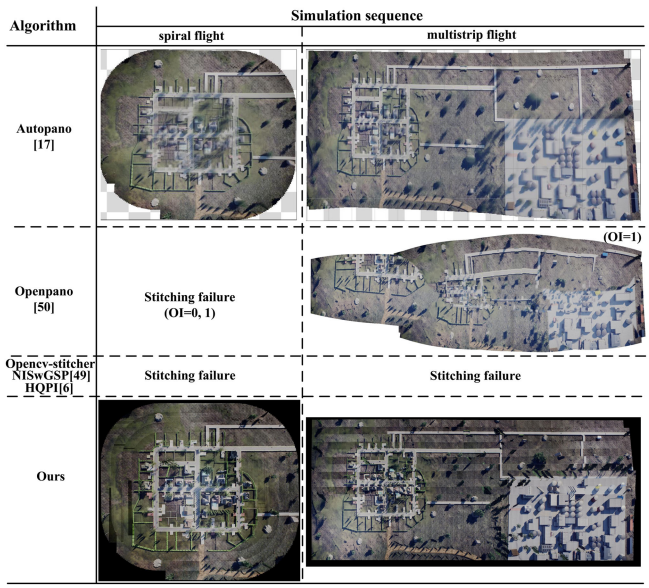

Fig. 10. Experimental results of six stitching algorithms in two simulation image sequences. Experimental scenarios include spiral flight over a small town and multistrip flight over a town and a factory. In the experiment, we compared the proposed framework with the mature stitching software Autopano, as well as the stitching algorithms Opencv-stitcher, Openpano, NISwGSP, and HQPI. Among them, the Openpano algorithm includes two modes: ordered stitching (OI = 1) and disordered stitching (OI = 0).

of Fig. 9. In inspire1-rail, after the fourth strip flight, although the image stitching effect in the flight direction was good, the stitching error between adjacent strips was large, as seen from the lake at the bottom of the mosaic image. Compared with the first method, the stitching effect of the method with keyframe insertion improved, and the cumulative error increased slowly. However, as the cumulative error could not be fully eliminated, it is still not suitable for long-distance flights. Moreover, there were often errors in the splicing of adjacent strips. For example, the road at the bottom of the mosaic image in sequence phantom3-highflower and the village on the right of the mosaic image in sequence inspire1-rail had some obvious stitching misalignments. In the proposed framework, the keyframe insertion and local optimization strategies not only reduced the cumulative error iteration frequency but also reduced the cumulative error by searching the surrounding keyframes for homography matrix optimization, such that robust and effective stitching results could be obtained in long-distance flights.

2) Comparisons With Other Stitching Algorithms: For comparison, we chose a mature stitching software Autopano, as well as state-of-the-art stitching algorithms Opencv-stitcher, Openpano, NISwGSP [49], and HQPI [6] for comparative experiments to demonstrate the performance of the proposed framework in terms of speed and stitching effect.

Autopano is a product of the virtual reality software company Kolor [17]. It is a powerful, easy-to-use, practical, and professional image stitching software that supports more than 400 input file formats. Opencv-stitcher is a new module added in OpenCV 2.4.0. Its function is to achieve image stitching. The algorithm has many parameters, including feature point categories, coordinate transformation models, and so on. In the experiment, the fusion plane was cylindrical, and other parameters were set as default. Openpano is a panorama stitching program written in C++ from scratch (without any vision libraries) [50]. This algorithm supports two modes, ordered stitching (ORDERED_INPUT = 1, or OI = 1) and disordered stitching (ORDERED_INPUT = 0, or OI = 0), which can be set by parameters; there are other parameter selections in the algorithm, as well. In the experiment, the multiband fusion frequency band was set to 2 and other parameters were set as default. The code of the NISwGSP algorithm was obtained from the public website [51], which adopts the local warp model and adds the global similarity prior to reduce distortion and generate a more natural stitched image. The code of the HQPI algorithm was obtained from the public website [52], which is a fast stitching approach implemented on mobile phones. The method uses dynamic programming to find an optimal seam quickly, sequential panorama stitching is used to reduce memory needs, and color correction is utilized to improve the quality of image labeling and blending.

As with the strategy comparison experiment, the four image sequences we selected were spiral flight and multistrip flight in the simulation, phantom3-highflower and inspire1-rail from the NPU Drone-Map dataset. The computer used had an Intel i7-7820X CPU with 64 GB RAM and an Nvidia Geforce gtx1080Ti GPU with 11 GB video memory. The experimental results are shown in Figs. 10 and 11, and the time performance of four stitching algorithms is shown in Table III. From Figs. 10

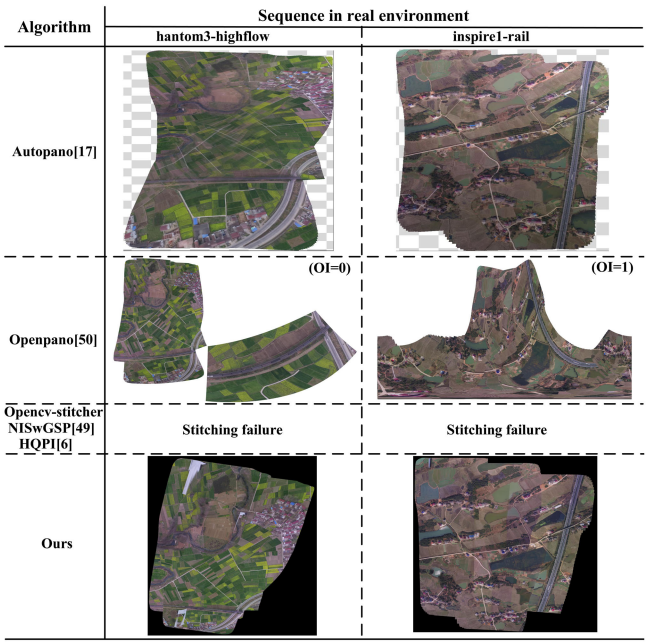

Fig. 11. Experimental results of six stitching algorithms on two real environment image sequences. Experimental image sequences included phantom3-highflower and inspire1-rail from the NPU Drone-Map dataset.

TABLE III
TIME PERFORMANCE OF SIX STITCHING ALGORITHMS

Algorithm / Sequence	Autopano [17]	Openpano [50]	Opencv-stitcher NISwGSP [49] HQPI [6]	Ours
spiral flight (1300 frames)	1.3 h	failure (OI=0,1)	failure	**57 s**
multistrip flight (495 frames)	300 s	2.5 h (OI=1)	failure	**39 s**
hantom3-highflow (601 frames)	427 s	12 h (OI=0)	failure	**93 s**
inspire1-rail (393 frames)	245 s	1.5 h (OI=1)	failure	**59 s**

and 11, we can see that the Autopano software generally obtained good stitching results. For example, the mosaic image obtained from the sequence inspire1-rail in Fig. 11 had almost no obvious stitching errors. However, if there were tall buildings in the scene, the mosaic image became blurred after fusion, as the images involved in the stitching come from different angles of the building, such as the mosaic image in Fig. 10. In addition, as the algorithm does not use the information that the input image is ordered; when there are different geographical locations in the image sequence but the appearance of the images is similar, the stitching is prone to errors, as seen in the mosaic image of the sequence phantom3-highflower in Fig. 11. In terms of time performance, the Autopano processes the image sequence offline and the stitching time is related to the image resolution and the number of images. As seen from Table III, for the image sequence inspire1-rail with a resolution of 1920×1080 and a number of 393, the stitching time was 245 s. For the sequence multistrip flight with a resolution of 640×480 and a number of 495, the stitching time was 300 s. When the number of images increased, the stitching time greatly increased. For example, for the sequence inspire1-rail with a resolution of 640×480 and a number of 1,300, the stitching time was 1.3 h. In summary, when the number of images was small, the algorithm generally completed the image stitching in a few minutes, and the stitching effect was good; however, it is easy to make mistakes for scenes with similar landscapes.

As the aerial image sequence acquired by a UAV is sequential, the ordered stitching mode is preferable for Openpano when testing the stitching effect. However, when Openpano performs ordered image stitching, it is likely that the stitching will fail due to the matching failure of the adjacent frames, such as the sequence inspire1-rail in Fig. 11 and the sequence phantom3-highflower in Fig. 11. For image sequences with successful stitching in order, the algorithm can generally stitch correctly for adjacent frames; however, in the case of long-distance multistrip flights, it is difficult to ensure the overall stitching effect. For example, in the mosaic image of the multistrip flight in Fig. 10, we can see that adjacent frames obtained by lateral flight could generally be correctly stitched, but images in the vertical direction were easily stitched incorrectly, which resulted in large mosaic errors overall. In addition, ordered stitching is prone to scale differences when estimating the image conversion model, resulting in deformation of the mosaic image, such as the sequence inspire1-rail in Fig. 11. For image sequences with ordered stitching failure, we reran the performance test using the disordered stitching mode. For the sequence spiral flight in Fig. 10, due to the large number of images, stitching failure occurred due to out-of-memory errors during the process. For the sequence phantom3-highflower in Fig. 11, the mosaic image obtained by the algorithm consists of two parts, as the two parts were not matched. The stitching effect of the two separate parts was good, and the error was small. From Table III, it can be seen that ordered stitching was faster than disordered stitching, and that the stitching time was related to the image resolution and number of images. In ordered stitching, for the image sequence inspire1-rail with a resolution of 1920×1080 and a number of 393, the stitching time was 1.5 h. For the image sequence of the multistrip flight with a resolution of 640×480 and a number of 495, the stitching time was 2.5 h. In disordered stitching mode, for the image sequence phantom3-highflower with a resolution of 1920×1080 and a number of 601, the stitching time was approximately 12 h. The ordered stitching mode of Openpano generally took approximately 3 hours when the number of images was approximately several hundred, and the stitching effect was unstable. In the case of disordered stitching, the algorithm could generally produce a good mosaic image, but its speed is too slow, more than 10 hours.

The Opencv-stitcher algorithm is generally applied to image sequence mosaics with a small number of images (i.e., dozens). In our experiment, the number of images in the sequences was large (several hundred), and stitching failures were often caused by insufficient memory when optimizing. Therefore, this algorithm is not suitable for constructing aerial image mosaics from long-distance flights. Similarly, the NISwGSP and HQPI algorithms failed to stitch on the four image sequences chosen. For these two algorithms, we tested the performance

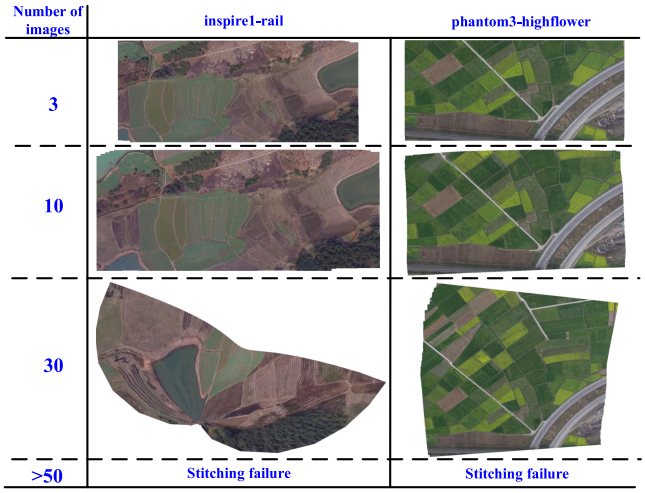

Fig. 12. Experimental results of the NISwGSP on two image sequences (phantom3-highflower and inspire1-rail) in a real environment. In this experiment, different numbers of images are selected from the sequence to obtain the stitching results of the NISwGSP.

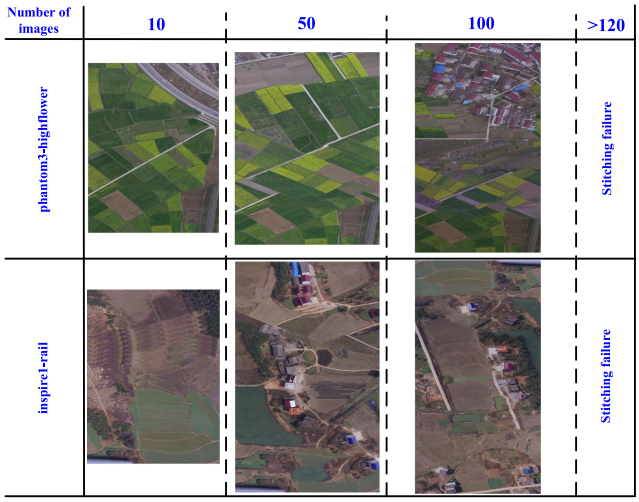

Fig. 13. Experimental results of the HQPI on two image sequences (phantom3-highflower and inspire1-rail) in the simulation environment. In this experiment, different numbers of images are selected from the sequence to obtain the stitching results of the HQPI algorithm.

and speed of the algorithm by selecting a different number of images in each sequence; some stitching results are shown in Figs. 12 and 13, and the time performance is shown in Table IV. Fig. 12 shows the stitching results of NISwGSP on the sequences phantom3-highflower and inspire1-rail in the real environment. When there were few images (e.g., 3, 10, or 50), the algorithm could construct the mosaic image, and the stitching effect was good. From the figure, we can see that there are no obvious stitching errors in the road in the sequence phantom3-highflower. However, with an increase in the number of images and more similar areas in the scene, the stitching effect may be reduced; for example, the stitching result with 50 images from the sequence inspire1-rail is distorted. In terms of algorithm speed, the processing time of the algorithm was approximately 450 s for 50

TABLE IV
TIME PERFORMANCE OF THREE STITCHING ALGORITHMS

Sequence \ Algorithm		NISwGSP [49]	HQPI [6]
spiral flight	3 frames	34.01 s	\
	10 frames	145.02 s	8.10 s
	30 frames	446.48 s	63.25 s
	100 frames	failure	483.66 s
	450 frames	failure	6478.96 s
	$\geq$ 480 frames	failure	failure
multistrip flight	3 frames	21 s	\
	10 frames	80.1 s	8.10 s
	30 frames	failure	60.63 s
	100 frames	failure	350.14 s
	450 frames	failure	10919.23 s
	$\geq$ 480 frames	failure	failure
hantom3-highflow	3 frames	32 s	\
	10 frames	130.86 s	98.18 s
	30 frames	436.31 s	640.68 s
	50 frames	failures	1114.21 s
	100 frames	failures	3242.20 s
	$\geq$ 120 frames	failure	failure
inspire1-rail	3 frames	29.77 s	\
	10 frames	135.33 s	105.79 s
	30 frames	389.19 s	572.07 s
	50 frames	failures	780.62 s
	100 frames	failures	3072.82 s
	$\geq$ 120 frames	failure	failure

images. The processing speed was mainly affected by the number of feature points. For example, sequence multistrip flight had fewer feature points, which takes less time, but the stability is poor, and it fails with 50 images. In general, this algorithm is suitable for processing a small number of images but not for long-distance real-time image stitching. Fig. 13 shows the stitching results of the HQPI on the sequences phantom3-highflower and inspire1-rail in the real environment. It can be seen that the algorithm can process more images than NISwGSP, but its memory occupation increases with an increase in the number of images; therefore, its processing power is still limited. The simulated sequence with a resolution of 640×480 can splice nearly 450 images; for the real environment sequence with a resolution of 1920×1080, it can splice almost 100 images. From the stitching effect in this figure, the stitching error of the algorithm is obvious, and it is difficult to obtain effective results for sequences collected by UAVs in multistrip flights. Referring to the speed of this algorithm in Table IV, we can see that the time required for the algorithm to process 100 images with a resolution of 640×480 was approximately 450 s, and the time required to process 100 images with a resolution of 1920×1080 was approximately 1,000 s. The processing speed of this algorithm is affected by image resolution: the higher the resolution, the slower the speed. In general, this algorithm is often used to process single-strip image sequences and is not suitable for the real-time splicing of UAV multistrip flight images.

Compared with these five algorithms, we can see that the proposed algorithm can achieve effective and robust stitching results in the four chosen image sequences. Although there were some errors, they were acceptable. In addition, our framework has obvious advantages in speed compared to the other five algorithms. For the image sequence inspire1-rail, with a resolution of 1920×1080 and a number of 393, the stitching time was

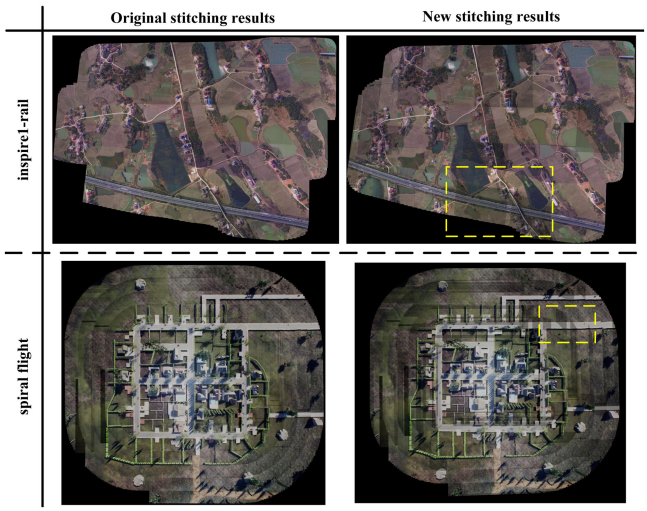

Fig. 14. By randomly changing the brightness of some images in the spiral flight and inspire1-rail sequences, two sets of new image sequences with brightness differences were generated. Based on these two image sequences, the original and new stitching results were obtained using the proposed algorithm.

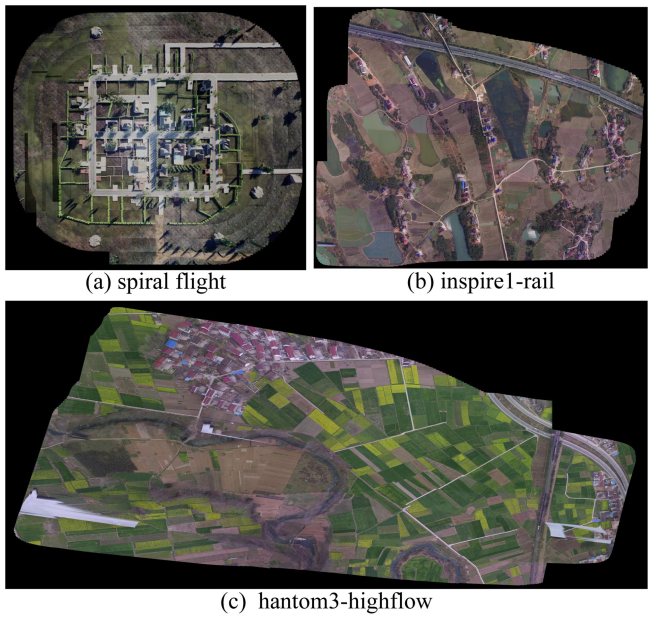

Fig. 15. Experimental results on three sequences in the simulation and real environments after replacing the SuperPoint features with SIFT features in the proposed framework. Three sequences include (a) spiral flight in the simulation, (b) phantom3-highflower and (c) inspire1-rail in real environments.

59 s. For the image sequence spiral flight, with a resolution of 640×480 and a number of 1,300, the stitching time was 57 s. In short, for image sequences with a resolution of 640×480, the processing speed of this algorithm is approximately 17 fps, which is basically real-time.

E. Supplementary Experiments

In this section, to further verify the adaptability to the environment and scalability of the proposed algorithm, we carried out experiments of illumination change and feature replacement in our framework in both real environments and the simulation environment.

1) Experiments of Illumination Change: To verify the robustness of the proposed algorithm to image brightness change, we generated two sets of new image sequences based on two sets of image sequences (inspire1-rail and spiral flight), with brightness differences introduced by randomly changing the brightness of some images in the sequences, and then retested the stitching effect of the algorithm. The original and new stitching results are shown in Fig. 14.

From the first line of Fig. 14, we can see that in the original image sequence (inspire1-rail) acquired from a real environment, when the UAV flew in a one-way strip, the image brightness basically did not change, and the image could be well stitched. When the UAV flew in a reverse strip, the brightness of the image changes. However, the proposed algorithm does not cause the error of horizontal stitching due to the brightness change between the strips. After changing the brightness of some of the images in the sequence, the brightness of the image changed when flying in a one-way strip, as shown in the yellow box on the first line of Fig. 14. However, the proposed algorithm still achieved effective stitching results, and compared with the original stitching results, the accuracy was not significantly affected. From the second line of Fig. 14, we can see that in the original image sequence (spiral flight) acquired from the simulation environment, due to the fixed lighting conditions, the brightness of the image changed slightly, and the overall stitching effect was good. After changing the brightness of some of the images in the image sequence, we can see that some of the images dimmed, such as the yellow box in the new stitching results, and, compared with the original stitching results, there were some small stitching errors in the road in the upper right corner; however, the overall stitching effect was not significantly affected. Overall, the proposed image stitching algorithm is robust to brightness changes in images. Generally, it can avoid image brightness adjustment preprocessing and thus reduce the computational burden.

2) Experiments of Feature Replacement in the Proposed Framework: In the proposed UAV image mosaic algorithm, to achieve the tracking task, we chose to extract SIFT features for image registration and calculate the homography transformation. In fact, in the proposed framework, this feature can be replaced by other features, such as deep learning features. Then, we used SuperPoint instead of SIFT to realize image registration in the tracking task and experimented on three image sequences in the simulation and real environments to verify the effectiveness and robustness of the algorithm framework. The experimental results are shown in Fig. 15.

The sequence spiral flight in Fig. 15(a) was collected continuously by a UAV flying along a spiral in the simulation. From the stitching results, it can be seen that under such complex flight conditions, the proposed framework did not produce obvious stitching errors due to the keyframe insertion strategy and the local optimization strategy, and the roads in the scene were stitched well (basically without dislocation). The sequence

phantom3-highflower and inspire1-rail in Fig. 15(b) and (c) were acquired by UAVs carrying out multistrip flights in real environments, with complex scenes such as wheat fields, lakes, villages, and many similar areas. From the stitching results, we can see that the stitching accuracy for a single strip was high. Then, due to the local optimization strategy proposed in the framework, although there were a few small stitching dislocations between the strips, the overall error was not large, and the stitching effect was good. Moreover, with the increase in the flight strips, the stitching error did not accumulate over time. Thus, from the experimental results of these three image sequences, we know that it is feasible to use SuperPoint for feature extraction and matching in the framework to obtain effective and robust stitching results.

In other words, the proposed framework has good scalability and is not limited to a specific algorithm, which provides a new research idea for aerial image mosaic technology. Moreover, the scalability also allows and supports the addition of new functional modules to increase the adaptability of the algorithm to harsh environments. For example, it is convenient to add an image enhancement [53]–[55] preprocessing module to this framework to improve the image quality when the UAV flies under very poor conditions.

IV. Conclusion

In this paper, we proposed a novel image-only real-time UAV image mosaic framework for long-distance multistrip flights. The framework has good scalability and only uses the image information of a single camera to implement three main tasks—automatic initialization, current frame tracking, and real-time mosaic generation—without any auxiliary information, such as GPS. More specifically, the initialization process is used to determine the stitching plane, and the tracking task is designed with a keyframe insertion strategy and homography matrix estimation based on a local optimization strategy. It can be noted that these two strategies not only reduce the cumulative error iteration frequency but also reduce the cumulative error, to obtain effective stitching results in long-distance multistrip flights. Simultaneously, to the tracking task, the image is mapped to the stitching plane to generate and update the stitching image in real time.

To evaluate the effectiveness of the proposed framework, we carried out a large number of experiments on the AirSim simulation platform and present an exhaustive evaluation in some sequences from a popular dataset. Qualitative and quantitative experimental results in simulation and real environments demonstrated that our algorithm can obtain an effective and robust panorama under different flight conditions in various environments, even under long-distance multistrip flight situations. Through strategy comparison experiments, we have proven that the keyframe insertion strategy and the local optimization strategy can improve the stitching performance. Compared with five state-of-the-art image stitching approaches, the stitching effect of the proposed algorithm is comparable or better. In terms of speed, its performance was superior to them, and for image sequences with a resolution of 640×480, the processing speed was basically real-time. Additionally, experiments of illumination change and feature replacement were carried out to verify that the proposed framework has good adaptability and scalability, which is not limited to a specific algorithm, and provides a new research idea for aerial image mosaic technology. In fact, this work has great application prospects in the UAV field, such as mapping, agricultural patrols and traffic monitoring, and it appears especially important in the situations such as aerial emergency rescue and rapid postearthquake surveys where the time is constrained and the high stitching accuracy is required. In the future, we will improve the local optimization part of the system to further reduce stitching errors. In addition, subsequent processing such as seam fusion of the mosaic image to make its color transition more natural and the overall effect better provides another direction for future research.

References

[1] F. Fang, T. Wang, Y. Fang, and G. Zhang, "Color blending for seamless image stitching," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 7, pp. 1115–1119, Jul. 2019.

[2] T. Yang *et al.*, "Hybrid camera array-based UAV auto-landing on moving UGV in GPS-denied environment," *Remote Sens.*, vol. 10, no. 11, p. 1829, 2018.

[3] J. Li *et al.*, "An adaptive framework for multi-vehicle ground speed estimation in airborne videos," *Remote Sens.*, vol. 11, no. 10, p. 1241, 2019.

[4] D. Avola *et al.*, "A UAV video dataset for mosaicking and change detection from low-altitude flights," *IEEE Trans. Syst., Man Cybern.: Syst.*, vol. 50, no. 6, pp. 2139–2149, Jun. 2020.

[5] P. Tsao, T. Ik, G. Chen, and W. Peng, "Stitching aerial images for vehicle positioning and tracking," in *Proc. IEEE Int. Conf. Data Mining Workshops*, Nov. 2018, pp. 616–623.

[6] Y. Xiong and K. Pulli, "Fast panorama stitching for high-quality panoramic images on mobile phones," *IEEE Trans. Consum. Electron.*, vol. 56, no. 2, pp. 298–306, May 2010.

[7] G. Zhang, Y. He, W. Chen, J. Jia, and H. Bao, "Multi-viewpoint panorama construction with wide-baseline images," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 3099–3111, Jul. 2016.

[8] P. Bajpai, A. Upadhyay, S. Jana, J. Kim, and V. K. Bandlamudi, "High quality real-time panorama on mobile devices," in *Proc. IEEE Int. Conf. Multimedia Expo. Workshops*, Jul. 2018, pp. 1–4.

[9] S. Liu and Q. Chai, "Shape-optimizing and illumination-smoothing image stitching," *IEEE Trans. Multimedia*, vol. 21, no. 3, pp. 690–703, Mar. 2019.

[10] L. Yu, Y. Zhang, M. Sun, and Y. Lu, "Automatic reference image selection for color balancing in remote sensing imagery mosaic," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 5, pp. 729–733, May 2017.

[11] W. Zhang, X. Li, J. Yu, M. Kumar, and Y. Mao, "Remote sensing image mosaic technology based on surf algorithm in agriculture," *EURASIP J. Image Video Process.*, vol. 2018, no. 1, p. 85, Sep. 2018.

[12] T. Bertel, N. D. F. Campbell, and C. Richardt, "Megaparallax: Casual 360 panoramas with motion parallax," *IEEE Trans. Visualization Comput. Graph.*, vol. 25, no. 5, pp. 1828–1835, May 2019.

[13] P. C. Madhusudana and R. Soundararajan, "Subjective and objective quality assessment of stitched images for virtual reality," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5620–5635, Nov. 2019.

[14] V. R. Gaddam, M. Riegler, R. Eg, C. Griwodz, and P. Halvorsen, "Tiling in interactive panoramic video: Approaches and evaluation," *IEEE Trans. Multimedia*, vol. 18, no. 9, pp. 1819–1831, Sep. 2016.

[15] J. Li, Z. Wang, S. Lai, Y. Zhai, and M. Zhang, "Parallax-tolerant image stitching based on robust elastic warping," *IEEE Trans. Multimedia*, vol. 20, no. 7, pp. 1672–1687, Jul. 2018.

[16] H. Yong, J. Huang, W. Xiang, X. Hua, and L. Zhang, "Panoramic background image generation for ptz cameras," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3162–3176, Jul. 2019.

[17] Autopano, [Online]. Available: http://www.kolor.com/autopano-download/, Accessed on: Jul. 4, 2019.

[18] Hugin, "Panorama photo stitcher, [Online]. Available: http://hugin.sourceforge.net/, Accessed on: Jun. 4, 2019.

[19] M. Lee, M. Shen, and C. J. Kuo, "Color matching techniques for video mosaic applications," in *Proc. IEEE Conf. Integr. Comput. Mater. Eng.*, Jun. 2004, vol. 1, pp. 121–124.
[20] C. Chailloux, J. Le Caillec, D. Gueriot, and B. Zerr, "Intensity-based block matching algorithm for mosaicing sonar images," *IEEE J. Ocean. Eng.*, vol. 36, no. 4, pp. 627–645, Oct. 2011.
[21] Z. Fu and Y. Han, "An algorithm of color image mosaic based on fourier-mellin transform," in *Proc. Artif. Intell. Comput. Intell.*, 2011, pp. 273–279.
[22] J. Chen, L. Luo, S. Wang, and H. Wu, "Automatic panoramic UAV image mosaic using ORB features and robust transformation estimation," in *Proc. 37th Chin. Control Conf.*, Jul. 2018, pp. 4265–4270.
[23] M. S. Patel, N. M. Patel, and M. S. Holia, "Feature based multi-view image registration using surf," in *Proc. Int. Symp. Adv. Comput. Commun.*, Sep. 2015, pp. 213–218.
[24] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. 4th Alvey Vision Conf.*, 1988, pp. 147–151.
[25] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.
[26] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to sift or surf," in *Proc. IEEE Conf. Int. Conf. Comput. Vision*, Nov. 2011, pp. 2564–2571.
[27] H. Deng, T. Birdal, and S. Ilic, "PPFNet: Global context aware local features for robust 3 d point matching," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 195–205.
[28] I. Melekhov *et al.*, "DGC-Net: Dense geometric correspondence network," in *Proc. IEEE Conf. Workshop Appl. Comput. Vision*, 2019, pp. 1034–1042.
[29] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "LIFT: Learned invariant feature transform," in *Proc. IEEE Conf. Eur. Conf. Comput. Vision*, 2016, pp. 467–483.
[30] Y. Tian, B. Fan, and F. Wu, "L2-Net: Deep learning of discriminative patch descriptor in Euclidean space," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 661–669.
[31] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. Workshops*, 2018, pp. 224–236.
[32] Z. Luo *et al.*, "ContextDesc: Local descriptor augmentation with cross-modality context," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 2527–2536.
[33] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Deep image homography estimation," 2016, *arXiv:1606.03798*.
[34] T. Nguyen, S. W. Chen, S. S. Shivakumar, C. J. Taylor, and V. Kumar, "Unsupervised deep homography: A fast and robust homography estimation model," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2346–2353, Jul. 2018.
[35] X. Meng and B. Wang, and W. Leong, "SkyStitch: A cooperative multi-UAV-based real-time video surveillance system with stitching," in *Proc. 23 rd ACM Int. Conf. Multimedia*, 2015, pp. 261–270.
[36] J. Zheng, Y. Wang, H. Wang, B. Li, and H. Hu, "A novel projective-consistent plane based image stitching method," *IEEE Trans. Multimedia*, vol. 21, no. 10, pp. 2561–2575, Oct. 2019.
[37] Y. Yang and X. Lee, "Four-band thermal mosaicking: A new method to process infrared thermal imagery of urban landscapes from UAV flights," *Remote Sens.*, vol. 11, no. 11, p. 1365, 2019.
[38] H. Wang, J. Li, L. Wang, H. Guan, and Z. Geng, "Automated mosaicking of UAV images based on SFM method," in *Proc. IEEE Geosci. Remote Sens. Symp.*, Jul. 2014, pp. 2633–2636.
[39] H. Zhou, D. Zhou, K. Peng, R. Guo, and Y. Liu, "Seamless stitching of large area UAV images using modified camera matrix," in *Proc. IEEE Int. Conf. Real-Time Comput. Robot.*, Jun. 2016, pp. 561–566.
[40] H. Pei *et al.*, "Accuracy analysis of UAV remote sensing imagery mosaicking based on structure-from-motion," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2017, pp. 5904–5907.
[41] S. Bu, Y. Zhao, G. Wan, and Z. Liu, "Map2DFusion: Real-time incremental UAV image mosaicing based on monocular slam," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2016, pp. 4564–4571.
[42] Y. Ge, G. Wen, and X. Yang, "A fast mosaicking method for small UAV image sequence using a small number of ground control points," in *Proc. 8th Int. Conf. Intell. Human-Mach. Syst. Cybern.*, Aug. 2016, vol. 2, pp. 90–94.
[43] J. Liu, J. Gong, B. Guo, and W. Zhang, "A novel adjustment model for mosaicking low-overlap sweeping images," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 4089–4097, Jul. 2017.
[44] D. Avola *et al.*, "Real-time incremental and geo-referenced mosaicking by small-scale UAVs," in *Proc. Int. Conf. Image Anal. Process. Image Anal. Process.*, 2017, pp. 694–705.
[45] W. Zhang, B. Guo, M. Li, X. Liao, and W. Li, "Improved seam-line searching algorithm for UAV image mosaic with optical flow," *Sensors*, vol. 18, no. 4, p. 1214, 2018.
[46] J. J. Ruiz, F. Caballero, and L. Merino, "MGRAPH: A multigraph homography method to generate incremental mosaics in real-time from UAV swarms," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 2838–2845, Oct. 2018.
[47] J. Ye, H. Chen, and W. Tsai, "Panorama generation based on aerial images," in *Proc. IEEE Int. Conf. Multimedia Expo. Workshops*, Jul. 2018, pp. 1–6.
[48] J. Chen, Q. Xu, L. Luo, Y. Wang, and S. Wang, "A robust method for automatic panoramic UAV image mosaic," *Sensors*, vol. 19, no. 8, p. 1898, 2019.
[49] Y. Chen and Y. Chuang, "Natural image stitching with the global similarity prior," in *Proc. IEEE Conf. Eur. Conf. Comput. Vision*, 2016, pp. 186–201.
[50] Openpano, [Online]. Available: https://github.com/ppwwyyxx/OpenPano, Accessed on: Jul. 4, 2019.
[51] Niswgsp, [Online]. Available: https://github.com/nothinglo/NISwGSP, Accessed on: Feb. 4, 2019.
[52] Fast-panorama-stitching-for-high-quality-panoramic-images-on-mobile-phones. [Online]. Available: https://github.com/additya1998/Fast-Panorama-Stitching-for-High-Quality-Panoramic-Images-on-Mobile-Phones, Accessed on: Feb. 4, 2020.
[53] S. Liu and Y. Zhang, "Detail-preserving underexposed image enhancement via optimal weighted multi-exposure fusion," *IEEE Trans. Consum. Electron.*, vol. 65, no. 3, pp. 303–311, Aug. 2019.
[54] S. Park, B. Moon, S. Ko, S. Yu, and J. Paik, "Low-light image enhancement using variational optimization-based retinex model," in *Proc. IEEE Int. Conf. Consum. Electron.*, 2017, 70–71.
[55] Y. Zhang and S. Liu, "Non-uniform illumination video enhancement based on zone system and fusion," *J. Comput.-Aided Des. Comput. Graph.*, vol. 29, no. 12, pp. 2317–2322, 2017.

Fangbing Zhang (Student Member, IEEE) is currently working toward a doctorate degree with the School of Computer Science and Technology, Northwestern Polytechnical University, Xi'an, China. Her research interests include UAV visual perception and navigation and visual SLAM.

Tao Yang (Member, IEEE) is a full professor at the School of Computer Science at Northwestern Polytechnical University. His research interests include intelligent visual perception and visual SLAM.

Linfeng Liu (Student Member, IEEE) is working toward the M.S degree with the School of Computer Science and Technology, Northwestern Polytechnical University, Xi'an, China. His research interests include simultaneous localization and mapping based on monocular vision systems.

Bang Liang (Student Member, IEEE) is working toward the M.S. degree with the School of Computer Science, Northwestern Polytechnical University, Xi'an, China. His research interests include visual navigation control.

Yi Bai (Student Member, IEEE) is working toward the M.S degree at the School of Computer Science, Northwestern Polytechnical University, Xi'an, China. Her research interests include simultaneous localization and mapping.

Jing Li (Member, IEEE) is an associate professor in the School of Telecommunications Engineering, Xidian University, Xi'an, China. Her research interests include image registration, matching and retrieval, and video content analysis and understanding.