electronics

Article

Monocular Vision SLAM-Based UAV Autonomous Landing in Emergencies and Unknown Environments

Tao Yang [1,2,*], Peiqi Li [1], Huiming Zhang [3], Jing Li [4,*] and Zhi Li [1]

1 SAIIP, School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China; page_7026@mail.nwpu.edu.cn (P.L.); zLeewack@mail.nwpu.edu.cn (Z.L.)
2 Research & Development Institute of Northwestern Polytechnical University in Shenzhen, Shenzhen 518057, China
3 National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China; mf1733077@smail.nju.edu.cn
4 School of Telecommunications Engineering, Xidian University, Xi'an 710071, China
* Correspondence: tyang@nwpu.edu.cn (T.Y.); jinglixd@mail.xidina.edu.cn (J.L.); Tel.: +86-150-0291-9079 (T.Y.); +86-139-9132-0168 (J.L.)

Received: 17 April 2018; Accepted: 11 May 2018; Published: 15 May 2018

Abstract: With the popularization and wide application of drones in military and civilian fields, the safety of drones must be considered. At present, the failure and drop rates of drones are still much higher than those of manned aircraft. Therefore, it is imperative to improve the research on the safe landing and recovery of drones. However, most drone navigation methods rely on global positioning system (GPS) signals. When GPS signals are missing, these drones cannot land or recover properly. In fact, with the help of optical equipment and image recognition technology, the position and posture of the drone in three dimensions can be obtained, and the environment where the drone is located can be perceived. This paper proposes and implements a monocular vision-based drone autonomous landing system in emergencies and in unstructured environments. In this system, a novel map representation approach is proposed that combines three-dimensional features and a mid-pass filter to remove noise and construct a grid map with different heights. In addition, a region segmentation is presented to detect the edges of different-height grid areas for the sake of improving the speed and accuracy of the subsequent landing area selection. As a visual landing technology, this paper evaluates the proposed algorithm in two tasks: scene reconstruction integrity and landing location security. In these tasks, firstly, a drone scans the scene and acquires key frames in the monocular visual simultaneous localization and mapping (SLAM) system in order to estimate the pose of the drone and to create a three-dimensional point cloud map. Then, the filtered three-dimensional point cloud map is converted into a grid map. The grid map is further divided into different regions to select the appropriate landing zone. Thus, it can carry out autonomous route planning. Finally, when it stops upon the landing field, it will start the descent mode near the landing area. Experiments in multiple sets of real scenes show that the environmental awareness and the landing area selection have high robustness and real-time performance.

Keywords: UAV automatic landing; monocular visual SLAM; autonomous landing area selection

1. Introduction

Unmanned aerial vehicles (UAVs) are non-manned aircraft that are operated by radio remote control equipment or a self-contained program control device. Drones have wide applicability in military and civilian areas because they have advantages of simple and practical structure, convenient and flexible operation, and low cost. Furthermore, users do not have to worry about casualties that drones may cause. They are widely used in military missions such as tactical reconnaissance and territorial surveillance, target positioning, and so on. In civil use, drones can be used for field monitoring, meteorological exploration, highway inspection, etc. With the popularization and wide application of drones in military and civilian fields, drones' safety issues must be considered. Relevant data show that the number of failures in the recycling process of drones accounts for more than 80% of the total number of failures of drones. Therefore, research on the safe landing and recovery of drones has become an urgent task.

However, due to the complex application environment of drones (especially in the context of war), drones' landing research needs to consider many factors and versatility, with improved practicality. To be more specific, the main challenges cover the following points: (1) **Autonomous control without GPS signal**. The anti-jamming capability of GPS is extremely weak. If the on-board GPS signal receiver of drones malfunctions due to electronic interference, drones will lose their navigation and positioning function, and thereby fail to land safely. In the natural state, GPS signals can be easily interfered. The influencing factors are mainly divided into four categories: (a) weather factors and sunspots may reduce signal strength, but generally do not affect positioning; (b) electromagnetic interference, radio, and strong magnetic fields all generate different levels of interference; (c) GPS signals will decrease under shelters, such as buildings, vehicles, insulation paper, trees, and metal components; and (d) high-rise buildings and dense high-rise buildings will affect GPS signals. Therefore, it is very important to study the autonomous positioning and flight control of drones without GPS signals; (2) **Passive landing in an emergency**. Since the drone's compensation mechanism does not allow the failed drone to continue flying for a long time, it should begin to select a site for emergency landing. Although this is a helpless move, it is also an important measure to prevent the drone from falling into densely populated areas. The Federal Aviation Administration of the United States believes that, in the future, drones must not only guarantee their own secure flight, but also have the ability to interact safely with a variety of aircraft in their airspace in the event of an emergency. Such regulations still assume the ability of drones to maintain communication between the air and ground during emergencies. In fact, when some more serious failures occur, a drone is likely to completely lose contact with the ground. At that point, drones' abilities to autonomously plan routes, autonomously search for landing sites, and autonomously land become the last resort to save themselves; (3) **Autonomous landing in an unknown environment**. In the military field or in disaster relief situations, the place where drones need to perform tasks is mostly an unknown environment or a variegated environment. It is essential that drones can choose landing sites with proper strategies and land safely.

To address these problems, researchers have made their contributions to drones' autonomous flight and secure landing. Jung et al. [1] propose a four-rotor drone guided landing algorithm. This paper presents a framework for the utilization of low-cost sensors for precise landing in moving targets. Based on the previous paper, authors in [2] describes the tracking guidance for autonomous drone landing and the vision-based detection of the marker on a moving vehicle with a real-time image processing system. Falanga et al. [3] presents a quadrotor system capable of autonomously landing on a moving platform using only onboard sensing and computing. This paper relies on computer vision algorithms, multi-sensor fusion for localization of the robot, detection and motion estimation of the moving platform, and path planning for fully autonomous navigation. Authors in [4,5] propose drone landing technology by identifying a sign and then landing the drone on the marker. Therefore, the drone needs to place the landing mark in the landing area before landfall. Vlantis et al. [6] studies the problem of landing a quadrotor on an inclined moving platform. The aerial robot employs a forward-looking on-board camera to detect and observe the landing platform, which is carried by a mobile robot moving independently on an inclined surface. Kim et al. [7] propose a vision-based target following and landing system for a quadrotor vehicle on a moving platform. The system employs a vision-based landing site detection and locating algorithm using an omnidirectional lens. Measurements from the omnidirectional camera are combined with a proper dynamic model in order to estimate the position and velocity of the moving platform. Forster et al. [8] proposes a resource-efficient system for real-time three-dimensional terrain reconstruction and landing spot detection for micro aerial vehicles. This paper uses the semi-Direct monocular visual odometry (SVO) algorithm to extract the key points to create the terrain map. However, SVO is a visual odometer based on the semi-direct method, which inherits some drawbacks of the direct method and discards optimization and loop detection. Authors in [9] propose a fixed-wing drone landing method based on optical guidance, using the ground landing guidance system to optically guide the landing. A measuring camera is arranged on both sides of the runway, and a marker light is installed in front of the drone, and the drone is spatially positioned by binocular stereo vision. This method has many outstanding advantages, being a self-contained system with high measurement accuracy that is low-cost and has low power consumption. Furthermore, it is less susceptible to interference without time accumulation errors. However, it is a ground guidance system and is not suitable for fully-autonomous landing of quadrotor UAVs in emergencies and unknown environments.

Despite the above, these methods have their own drawbacks and limitations: (1) most existing methods focus on landing the drone on a marker; and (2) some methods use model-based approaches to deal with missing visual information. Alternative solutions are realized with the use of additional sensors attached to landing area. Among many, these sensors include inertial measurement units (IMUs), GPS receivers, or infrared markers; and (3) previous research has only been able to accomplish landing in a given environment. Additionally, GPS [7,10–12] or motion capture systems [13,14] are often used for state estimation, either only while patrolling or throughout the entire task. Conversely, this paper relies only on visual–inertial odometry for state estimation. These approaches do not work in many cases, such as an emergency landing of a UAV or landing in a stricken area. There are also many emergency situations during the flight of UAVs, such as low battery power, machine malfunctions, or some unexpected conditions where drones need to be landed in unmarked areas. Therefore, we need UAVs to be able to land autonomously in unstructured and natural environments.

In order to land in unknown environments, our approach is to use visual landing technology. The simultaneous localization and mapping (SLAM) algorithm is a hotspot of robot and computer vision research, and is considered as one of the key technologies for automatic navigation in unknown environments. In 2007, Professor Davison presented MonoSLAM [15], which is the first real-time monocular vision SLAM system. MonoSLAM was designed to expand the Kalman filter for the back-end, tracking very sparse feature points. Parallel tracking and mapping (PTAM) [16] is a well-known single-SLAM algorithm that proposes and implements the tracking and mapping into two separate thread modules, and greatly improves the efficiency of the algorithm so that the algorithm can run in real time. PTAM combined with augmented reality (AR) is used in augmented reality software. Nevertheless, it also has its own limitations. For example, it can only be applied in a relatively small working environment. Forster et al. propose a semi-direct monocular visual odometry (SVO) based on a semi-direct method [17]. It uses pixel brightness to estimate pose, resulting in the ability to maintain pixel-level precision in high-frame-rate video. However, SVO abandons the optimization and loop detection part in order to improve speed and make the system lightweight, which results in increased calculation error and inaccurate posture estimation under a long running time, and the loss is not easy to reposition. LSD-SLAM (large-scale direct monocular SLAM) [18] is an algorithm based on features and direct methods, proposed by Engel et al. It applies the direct method to semi-dense monocular SLAM, which can realize semi-dense scene reconstruction on a CPU. Since LSD-SLAM uses direct methods to track, it also inherits the disadvantages of direct methods. For example, LSD-SLAM is very sensitive to the intrinsic camera parameters and exposure, and it fails very easily in the process of fast motion. Mur-Artal et al. propose a feature-based monolithic SLAM system, ORB_SLAM2 [19], which can be applied to all scenes in real-time. The algorithm is divided into four modules: tracking, building, relocating, and closed-loop detection. The system is divided into three separate threads, which can successfully track and build a map. With the advent of the new sensor event-based camera, many SLAM studies based on event cameras [20,21] have emerged in recent years. However, the event-based camera is expensive and has a low spatial resolution, which limits the performance of the application.

In this paper, a vision-based UAV landing method is used. With the help of optical equipment and image recognition technology, the UAV is capable of autonomously identifying the landing zone and reconstructing three-dimensional terrain to accomplish automatic return and route planning. Before preparing for an autonomous landing, the drones scan the scene, analyze the appropriate location, and initiate an autonomous calibration of the initial landing site. Then, it carries out autonomous route planning for this point. When the appropriate landing path is calculated, the autonomous landing control mode is automatically switched on and the proper solution is adopted to approach the landing site. After reaching the landing field along with a correct landing route, it starts the descent mode. As drones continue to decline, the control system systematically identifies information such as altitude change rate and pose of the drone, and adjusts the altitude of the drone at any time until it lands in a predetermined landing zone. As shown in Figure 1, the experiments were performed in multiple scenarios. The results of the experiments show that the environmental awareness and the landing area selection have high robustness and real-time performance.

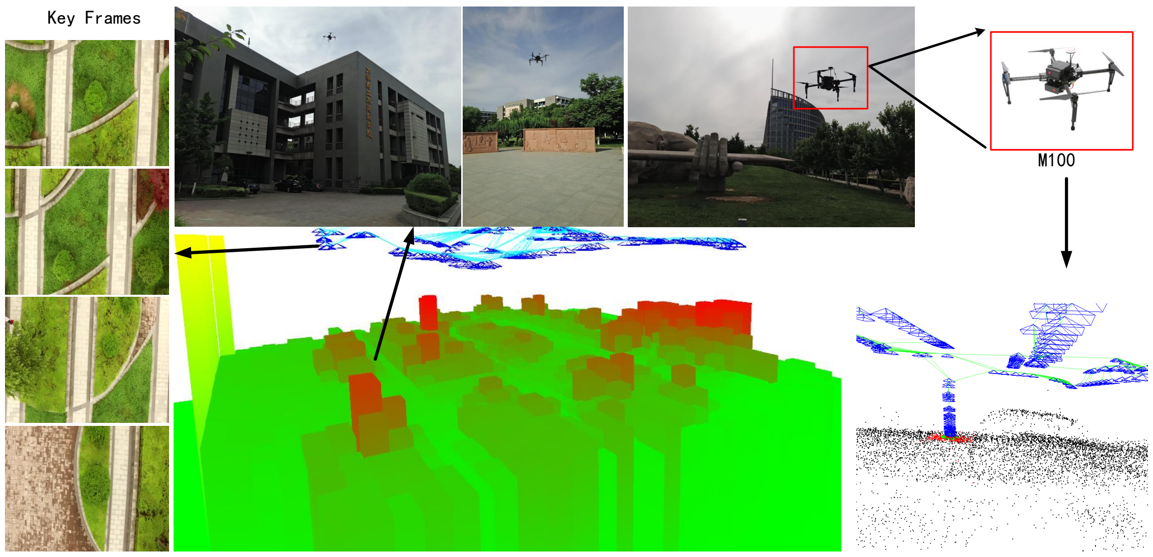


Figure 1. The M100, also known as the Matrice M100, is a quadrocopter drone for developers, released by DJI (Shenzhen, China), the world's largest consumer drone maker. The drone uses a monocular camera to scan the ground for key frames and a three-dimensional point cloud map. Then, it converts the three-dimensional point cloud map into a grid map, detects suitable landing areas, and carries out a landing. The map in the lower center of the picture shows a flat area in green and an area higher than the horizon in red. The depth of red indicates the height. The blue lines above the map show the keyframes created by the drone during the flight. The left column of the picture is part of the key frames.

When an airport fails to receive a control signal in some case (e.g., system failure or signal interference), the ability to land at the airport with a good strategy and land safely will greatly reduce the damage of the unmanned drone to the ground personnel. Advanced autonomous visual landing control technology can avoid the danger of drones facing emergency landings. With the use of optical and other detection equipment, the autonomous sensing capabilities of drones will be greatly improved, and evasion will be implemented before ground controllers are put in danger. An anti-collision algorithm that takes both safety and economy into consideration will automatically re-plan the route after the UAV has implemented collision avoidance maneuvers to continue the task.

This method can be applied not only to the passive landing of drones in complex scenarios or in emergency situations and the active landing of drones, but also to many areas such as the automatic driving of unmanned vehicles, augmented reality, and the autonomous positioning of robots.

The main contributions of this work can be summarized as follows:

- This paper proposes and implements the vision-based drone autonomous landing system in an unstructured environment. By combining existing technologies, they are improved to better meet the requirements of the system proposed herein.
- This paper proposes a novel map representation approach that combines three-dimensional features and a mid-pass filter. Each visible feature is converted into a grid map by utilizing the mid-pass filter to remove noise that is too high and too low in each grid. After constructing a grid map, feature points of different heights can be visualized as grids of different heights.
- This paper recommends a region segmentation to detect the edge of different-height grid areas. It smooths the areas with the same height based on a mean shift algorithm. An edge detector is used to identify obstacles and flat areas. By region segmentation, the speed and accuracy of the subsequent landing area selection are substantially improved.
- Based on a grid map and region segmentation, we present a visual landing technology to explore a suitable landing area for drones in emergencies and unknown environments. Furthermore, with the pose calculated by SLAM, drones can autonomously fulfill path planning and implement landing. To evaluate the proposed algorithm, we apply it in multiple sets of real scenes. Experimental results demonstrate that the proposed method achieves encouraging results.

The remainder of this paper is organized as follows. In Section 2, we propose a UAV autonomous landing approach based on monocular visual SLAM. The experimental results are presented in Section 3. Finally, we conclude the paper in Section 4.

2. The Approach

An overview of the proposed algorithm for the detection of landing sites is shown in Figure 2. When the drone begins the landing procedure, the approach can estimate the position and posture of the drone, build the grid map of the environment, and select the most suitable area for landing via the filtering algorithm. A selecting landing area and vision navigation method is demonstrated, which uses SLAM to estimate the current pose of the drone.

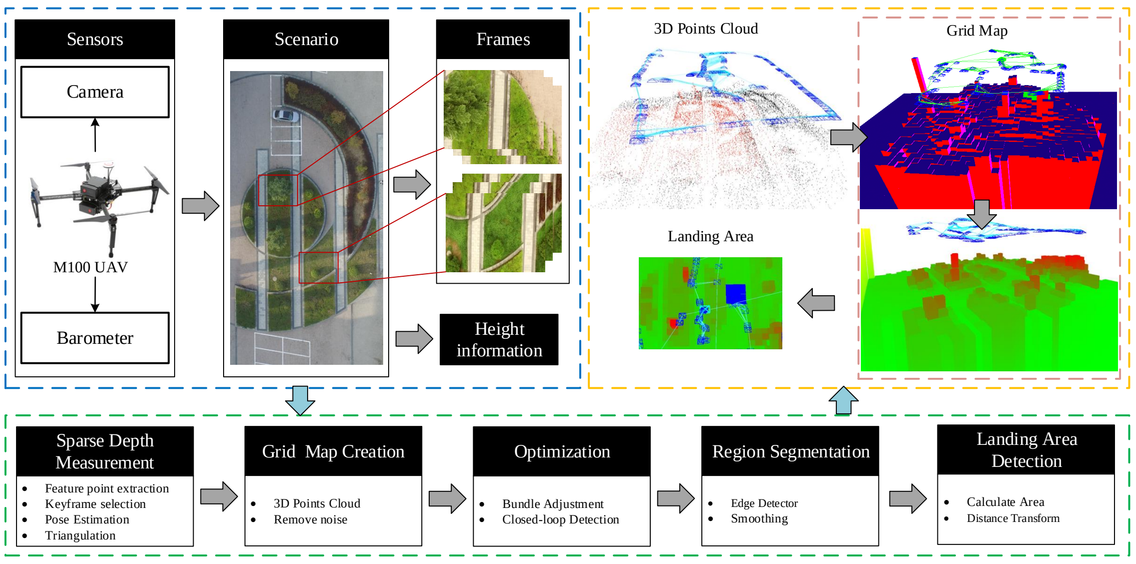


Figure 2. Overview of the main components and connection in the proposed approach.

This paper establishes a three-dimensional point cloud map of the environment by visual SLAM. Then, a two-dimensional grid map is set up by the three-dimensional point cloud of the feature points proposed by the SLAM algorithm. The height of each grid is calculated by projecting the map points of the graph into the corresponding grids. Then, the mean shift-based image segmentation algorithm is used to smooth the height of the grid map, divide the obstacles and ground, and combine the highly similar image blocks together. By calculating the space distance between the landing area and the obstacle, the algorithm selects the region which is the farthest from the obstacle as the filtered landing area. In this way, a suitable area for UAV landing is selected. The UAV finally lands on the safe area by following the descent program.

2.1. Sparse Depth Measurement

The camera pose is forwarded to the on-board computer, which associates the camera pose with the corresponding images based on the pre-calibrated camera. These camera pose estimates are used as priors in the bundle adjustment if an area is marked as a potential landing spot. Additionally, feature tracks are generated by the provided framerate. The ORB (oriented FAST and rotated BRIEF) [22] feature tracker is used to generate coarse depth measurements for region tracking in unknown terrain and in the bundle adjustment of the backend thread. The ORB feature tracker is made up of the oriented FAST (Features from Accelerated Segment Test) [23] corner detector, which detects corners with a description of scale and rotation, and BRIEF (Binary Robust Independent Elementary Features) [24], an efficient feature point descriptor.

Monocular vision SLAM is a feature-based system that can be applied to all scenes in real time. The algorithm divides the algorithm into four modules: tracking, building, relocating, and closed-loop detection. It divides the system into three separate threads, which can track well and build a map. The ORB_SLAM2 [19] algorithm can guarantee the global consistency of trajectory and map through its optimization and closed-loop detection. If the camera returns to the same scene as before, the algorithm can optimize posture and map by conducting closed-loop detection.

If the scene is a plane, or is approximated as a plane, or when the parallax is small, the motion estimation can be done by homography. The motion is restored by the basic matrix F in a non-planar scene with large parallax. Although the camera is facing the ground, the data captured by the drone may involve rugged terrain. The basic matrix F indicates the relationship between any two images of the same scene that constrains where the projection of points from the scene can occur in both images. At the same time, in order to improve the robustness of the system, the basic matrix F and the homography matrix H are estimated at the same time when the real data always contains some noise. The homography matrix H describes the mapping relation between two images of the same planar surface in space. Then, the smaller one is chosen as the motion estimation matrix by comparing the weight projection error. The method in the ORB_SLAM2 algorithm is to calculate the SH value and select the corresponding model according to the RH worth.

(1) Extract reference frame and current frame features p_r, p_c, and then match features between two frames $p_r \leftrightarrow p_c$. If the number of matching features are not enough, the reference frame is reset.

(2) Calculate the homography matrix H: $p_c = Hp_r$, and then calculate the fundamental matrix F: $p_c^T F p_r = 0$, $F = K^{-T}EK^{-1}$, where K is the internal matrix of the camera and E is the essential matrix. The essential matrix E can be seen as a precursor to the fundamental matrix, and its relationship with F is as above. The degrees of freedom of the homography matrix is 8, which can be computed by four pairs of matching features. The fundamental matrix F can be calculated by the classical eight-point-algorithm [25]. It is unavoidable that there is a large number of mismatches in feature matching, and we use random sampling consistency (RANSAC) to solve them.

(3) Restore motion from the fundamental matrix F or the homography matrix H.

We can get the motion T_{12} of the camera by the polar constraint. The depth information of the map points can be estimated by the motion of the camera by triangulation. p_1, p_2 are set to represent the coordinates after the features are normalized on two frames,

$$d_1 p_1 = d_2 T_{12} p_2, \tag{1}$$

where d_1, d_2 represent the depth of two-frame features, p_1, p_2 are the three-dimensional coordinates of the current frame and reference frame features, and T_{12} comprised of rotation matrix R and translation matrix t is the transformation matrix of the first graph to the second graph.

However, when solving the pose of the camera, because of the scale equivalence of the essential matrix E itself, there is also a scale equivalence of the t, R obtained by decomposing E. The normal method is to normalize the scale of t, which leads directly to the scale uncertainty of monocular vision. For t, after it is multiplied by an arbitrary constant, the polar constraint is always established. In order to solve this scale uncertainty, we compared the height information measured by the barometer with the flight altitude calculated by the SLAM system to obtain the scale factor.

The Matrice M100 comes with a barometer module. The barometer is based on the experimental principle of Evangelista Torricelli for measuring atmospheric pressure. Most of the aircraft's altitude measurement is achieved through a barometer, and GPS-equipped aircraft also generally have a barometer as a backup. For every 12 m of height raised, the mercury column is lowered by about 1 millimeter, so the height of the aircraft when it is flying in the air can be measured. In the experimental scenario, the drone's flying height is usually no more than 30 m, so the error of the height measured by the barometer is not very large and satisfies the experimental requirements. The experiment in this article has the feature that the monocular visual SLAM application environment is oriented to the ground. In addition, the camera ZENMUSE X3 (DJI, Shenzhen, China) used in the experiment has a PTZ (Pan/Tilt/Zoom) self-balancing function, which can ensure that the camera maintains its ground-facing state. A PTZ camera is a camera that supports all-around (up, down, left, and right) movement and lens zoom control. Because of the scale problem of monocular vision SLAM and the characteristics of the experiments in this paper, the height information obtained by the barometer measurement is sufficient to restore the SLAM scale factor on the z-axis:

$$\triangle H = \triangle h_v * s, \tag{2}$$

where $\triangle h_v$ represents the height difference in the z-axis measured by monocular visual SLAM, $\triangle H$ represents the height variation in world coordinate system, and s represents the scale factor. The value of s can be calculated by replacing $\triangle H$ with the height variation measured by a barometer. It is thus possible to continue to obtain true height information by scaling the visual pose.

After the success of the map initialization, PNP (Perspective-N-Point) is used to transform the three-dimensional motion into two-dimensional point pairs. Therefore, the position posture of the current frame can be obtained by using a PNP solution to calculate the three-dimensional motion to two-dimensional points by using the three-dimensional map point P in the reference frame and the two-dimensional keypoints p on the current frame.

Given a three-dimensional map point set P and two-dimensional matching on the set of points p, we can calculate the pose by minimizing the re-projection error pose:

$$\xi^* = argmin_{\xi} \frac{1}{2} \sum_{i=1}^{n} \left\| u_i - \frac{1}{s_i} K T P_i \right\|_2^2 . \tag{3}$$

The error is obtained by comparing the pixel coordinates (i.e., the observed projected position) with the position of the three-dimensional point projected according to the currently-estimated pose. This error is called the re-projection error. We minimize the re-projection error of the matching point by constantly optimizing the pose in order to obtain the optimal camera pose.

For each frame, when the map is initialized, the system estimates the position of the current frame in accordance with the previous frame. Hence, with successful tracking, it is relatively easy to get the posture information of each frame.

However, a sparse feature point map does not meet the requirements of the screening landing area. Thus, it is imperative to integrate other methods to optimize the map.

2.2. Grid Map Creation

First of all, as shown in Figure 3, we divide the plane into small grids. The size of the grid can be adjusted according to the actual situation. Then, the SLAM algorithm is applied to calculate the three-dimensional location of feature point in the world coordinate system and the pose of each key frame. Then, we will convert the three-dimensional point clout into a two-dimensional grid map. Furthermore, this article sets the point to be projected into the grid only when it is observed by multiple frames. If it is observed by merely one frame, the point will not be projected into the grid, which can avoid points that are noise. Thus, each grid has a pile of two-dimensional points with the height information. There is one final step needed to determine which grids are suitable for drones to land on based on these points.

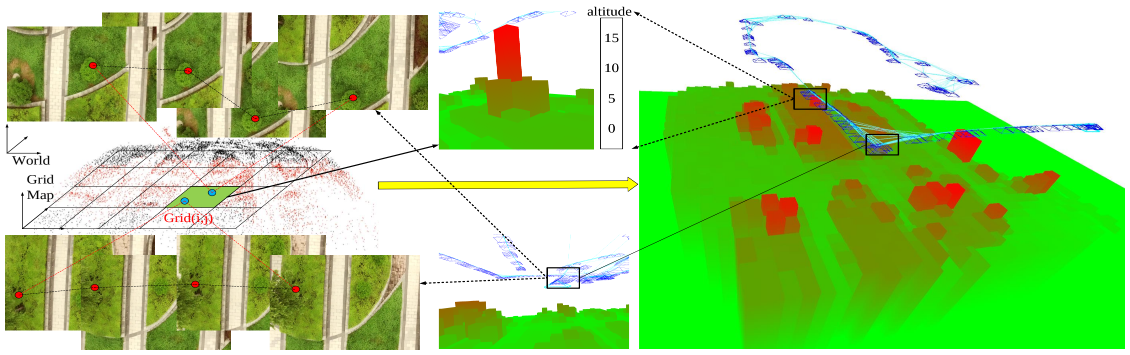


Figure 3. A feature point in real-world coordinates is observed by multiple key frames. Through triangulation, its three-dimensional position is obtained and converted into a three-dimensional point cloud. Then, the three-dimensional point cloud is projected into the two-dimensional grid map, and the height of the grid is obtained by calculating the height of all of the filtered three-dimensional points that fall in each grid. The premise of triangulation is to know the pose of each key frame, as shown in the blue block diagram.

First, we define the height of each grid:

$$h(i,j) = \frac{\sum_{k\epsilon N(i,j)} - h_{min} - h_{max}}{N(i,j) - 2}, \tag{4}$$

where h_k represents the map points in the grid $Grid(i,j)$, and the map points h_{min} and h_{max} represent the maximum and minimum values in the grid. The highest and lowest points in the grid are removed, and then the mean of the map points is computed to assign the value to the grid height $h(i,j)$.

The following formula defines whether the grid is suitable for landing:

$$T(i,j) = \sum_{m,n\epsilon R(i,j,r)} \|h(m,n) - h(i,j)\|^2, \tag{5}$$

where $h(m,n)$ is the height value of the map point (m,n) on the two-dimensional grid map and $h(i,j)$ represents the height of the grid (i,j) in the grid coordinates. r is the radius of the search, and is adjusted according to the size of the UAV. Through traversal of each grid, the drone can search for the landing area. Grids that do not have a projection point are regarded as unreliable and marked as non-landing areas. Finally, the threshold of the formula $T(i, j)$ is set according to the actual application to determine the grid suitable for the UAV landing, and at the same time the grid is marked out.

2.3. Pose and Map Optimization

There will be errors when the camera is calibrated and tracked, so it is necessary to do some optimization after the pose estimation. The estimate of the pose is obtained by tracking frames. By using this estimate as an initial value, we can model the optimization problem as a least-squares graph optimization problem and then use g2o (General Graph Optimization) [26] to optimize poses and maps. Even after optimization, there will be errors, and these tracking errors will continue to accumulate, which may lead to an increasingly growing rear frame pose estimation error, and eventually deviate from reality. Thus, long-term estimates of the results will be unreliable. Considering this, closed-loop testing, which is related to the correctness of estimated trajectory and maps after a long time, is particularly important.

Because the pose of key frames is estimated based on the previous reference frame, the error will be accumulated and result in increasingly inaccurate posture estimates. Therefore, we optimize the position and orientation using a closed-loop detection. When the camera captures the previously captured image, we can correct the position of the camera by detecting the similarity between the images. Closed-loop detection can be achieved through the word pocket model DBow3 [27]. DBoW3 is an open source C++ library for indexing and converting images into a bag-of-word representation. It implements a hierarchical tree for approximating nearest neighbours in the image feature space and creating a visual vocabulary. DBoW3 also implements an image database with inverted and direct files to index images and enabling quick queries and feature comparisons.

1. Feature extraction: select features based on the data set and then describe them to form feature data. For example, the sift key points in the image are detected, and then the feature descriptor is calculated to generate a 128-dimensional feature vector;
2. Learning the word bag: merge all of the processed feature data. Then, the feature words are divided into several classes by means of clustering. We set the number of these classes, and each class is equivalent to a visual word;
3. Use of a visual bag to quantify the image features: each image consists of many visual words. It can use statistical word frequency histograms to indicate which category an image belongs to.

With the dictionary, given any feature, the corresponding word can be found by looking up the dictionary tree layer-by-layer. When the new key frame is inserted, the distribution of the image in the word list or the histogram can be computed. This allows us to use the text search algorithm TF-IDF (term frequency-inverse document frequency) [28] and the approach in [29] to calculate the similarity between the two images. After detecting the closed loop, BA (bundle adjustment) is used to optimize some of the previous reference frames.

2.4. Region Segmentation-Based Landing Area Detection

It is necessary to divide the map according to the height before the screening of the landing area suitable for the UAV. The grid map of precise region segmentation based on height is helpful to improve the speed and accuracy of the subsequent landing area selection. In this paper, a method based on image segmentation to divide the height region of the grid map is proposed. This section introduces the algorithm flow in detail.

According to the experimental requirements, an image segmentation method based on mean shift [30] is used to segment the mesh map. In accordance with image segmentation based on the mean shift principle, the grid map obtained in the second chapter is smoothed and divided. Firstly, the size of the grid map and the height of each grid are input. Each grid is regarded as the smallest unit. Secondly, the mean shift algorithm is used to cluster the height of the grid to determine the total number of categories and the center of each category. Then, using these statistics as input, the final division of the grid map via the mean shift algorithm is obtained. Specific steps are shown in Algorithm 1.

Algorithm 1 Image Segmentation-Based Grid Map Partitioning Algorithm.

```
Input: grid map
 1: use the mean shift algorithm to smooth the created grid map. For each grid, initialize $j = 1$,
    $y_{i,1} = x_i$.
 2: while modulus point non-convergence do
 3:     calculate $y_{i,j+1}$
 4:     $z_i = (x_i^{(s)}, y_{i,c}^{(r)})$
 5: end while
 6: the grid map is smoothed with mean shift, and the convergence result is stored in $z_i$, $z_i = y_{i,c}$
 7: for $i = 0, 1, 2 \cdots N$ $z_i$ do
 8:     if grid space distance $< h_s$ && height distance $< h_r$ then
 9:         divided into different categories $C_{p_{p=1,\cdots m}}$
10:     end if
11: end for
Output: for each grid $i = 1, 2, \cdots, n$, the category logo $L_i = p|z_i \in C_p$.
```

After clustering the grid map, the ground condition without a priori environment information can be obtained. Therefore, the system gains understanding of the height distribution of the ground and the obstacle information to a certain extent, and is able to select an area suitable for the UAV to land. Then, the world coordinates in this district can be output to guide the landing of the UAV. Due to the skewing that may occur during the drone's landing, the UAV landing point needs to avoid obstacles in order to ensure a safe of landing. For the grid map, the algorithm sets the districts of all the altitude, except the landing height H, as obstacles. After districts with matching height and area are selected, it is necessary to calculate the integrated distance between the districts and all the obstacles. Specific steps are shown in Algorithm 2.

Algorithm 2 Choose the Best Landing Spot.

```
Input: the previous grid height categories $C_i$, the appropriate landing height $H$, the appropriate
    landing area $S$.
 1: for $i = 0, 1, 2 \cdots N$ grid categories $C_{i_{i=1,\cdots N}}$ do
 2:     if the height $h$ of $C_i = H$ && the area $s = S$ then
 3:         add grid $g_i$ to the landing zone candidate set and number $a_i$
 4:     else
 5:         add grid $g_i$ to the obstacle set and number $b_j$
 6:     end if
 7: end for
 8: for $i = 0, 1, 2 \cdots N$ the landing zone candidate set $a_i$ do
 9:     for $j = 0, 1, 2 \cdots N$ the obstacle set $b_j$ do
10:         calculate the distance $d_{ij}$ of each area $a_i$ from each obstacle $b_j$; that is, the distance from the
            candidate area to the nearest edge of the obstacle area.
11:         calculate the overall distance $d_i = d_i + d_{ij}$ of each area $a_i$ from all obstacles.
12:     end for
13: end for
Output: The area $a_i$ with the largest $d_i$ is the landing point, which makes it possible to stay away from
    existing obstacles.
```

The appropriate landing height H is selected from the previous grid height categories and the appropriate landing area S is set according to the size of the UAV. Then, the system can determine the best landing location through these two screening approaches.

3. Experiments

3.1. Experimental Platform

This paper chose the commercial UAV DJI Matrice 100 (M100) as a platform for the data acquisition offline processing stage and the UAV autonomous real-time control stage experiment. It includes a flight controller, power system, barometer module, GPS module, and other modules. This paper used the monocular camera, ZENMUSE X3, as a visual sensor to carry out the experimental data collection. The small ZENMUSE X3 monocular camera can guarantee high-quality video during high-speed movement with its wide-angle fixed focus lens, powerful performance, a shooting screen without distortion, and clear images. At the same time, we used the camera's PTZ self-balancing function to make the camera face the ground. We also used the barometer module on the UAV to measure the flight altitude for the monocular SLAM scale correction. In this paper, the image data resolution was 640×480. The experimental configuration environment was an Intel Core i7-8700K CPU@3.70 GHz processor, 16.0 GB memory, and 64-bit operating system.

If the camera used does not have a PTZ self-balancing function, the height difference on the z-axis cannot be used directly when restoring the scale factor. In the calculation of the scale factor stage, the aircraft only moves in height, and the difference in the three-dimensional space of the pose in this time period is calculated to replace the difference in the z-axis. In addition, during the initial phase of the aircraft, it is necessary to ensure that the camera is parallel to the ground. After this, the PTZ self-balancing function no longer affects system functionality.

The experimental platform interacted with the M100 through the image acquisition card and the wireless serial port to simulate the on-board processing, as shown in Figure 4. The laptop captured the image stream taken by the drone in real-time through the image capture card, and sent the control command to the drone in real-time through the wireless serial port.

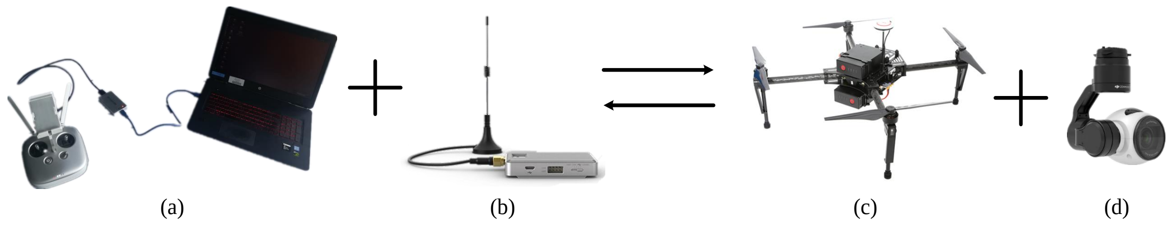


Figure 4. The image (**a**) is the experimental configuration environment. One end of the image acquisition card is connected to the computer, and the other end is connected to the remote control of the drone. The image capture card transmits the image to the computer via a remote control; The image (**b**) shows the wireless serial port module; The image (**c**) shows the M100 drone; and the image (**d**) shows the ZENMUSE X3 camera.

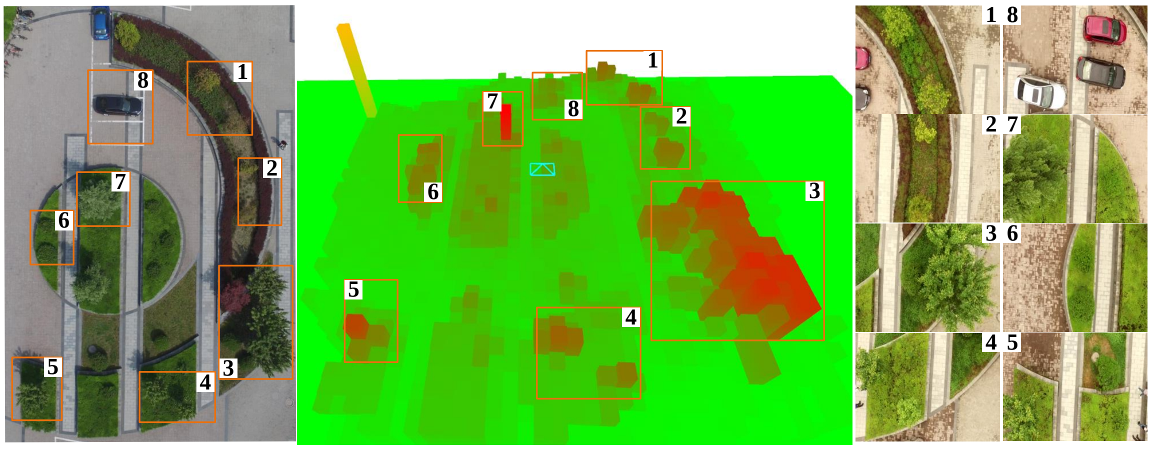
3.2. Real-Time Control Experiments

In Section 2, we can get the key frame data of monocular image sequence processing, and obtain the UAV pose information and the three-dimensional point cloud data according to the visual SLAM system. Then, we convert the three-dimensional point cloud into a two-dimensional grid. Next, the image segmentation algorithm based on mean shift is used to deal with the two-dimensional grid map. Finally, the two-dimensional grid map is filtered and the appropriate landing area is obtained.

At the beginning of the experiment, we first turned on the computer and started the program. The drone flew over the scene and began to simulate entering the fault state. It was necessary to start the autonomous landing procedure. The startup program controlled the drone to scan the landing environment. When the UAV started the data acquisition, the first step is to initialize. At the initial location, the aircraft pose must have a translation instead of a rotation change. In order to reduce the error of the pose information and the 3D point cloud data estimated by the SLAM system, after completing the initialization, the UAV started the closed-loop flight, followed by some closed loops of smaller radius. By forming a closed loop, the construction and pose estimation results are optimized. The candidate landing area is selected from within the construction result. The drone moved to the top of the candidate area with the shortest path, and focused on the candidate landing area (i.e., movement with small displacement). Finally, if the program still confirmed that the area was the final landing area, the landing mode was started. Otherwise, the candidate area was re-determined and scanned.

Figure 5 shows the creation of a two-dimensional height grid map and the specific meaning of each part of the grid map. The color represents the depth of the map. Green represents the lowest point, red represents the highest point, and the middle height is represented by a gradient color. Dark blue indicates a suitable landing site. Light blue indicates the flight path of the drone. The experimental scene is a circular flower bed surrounded by large semi-circular flower beds. There are three cars parked in front of the flower bed. There are big trees, shrubs and weeds on the circular flower beds and the semicircular flower beds on the periphery. The other half of the flower bed is an empty square which is suitable for landing.

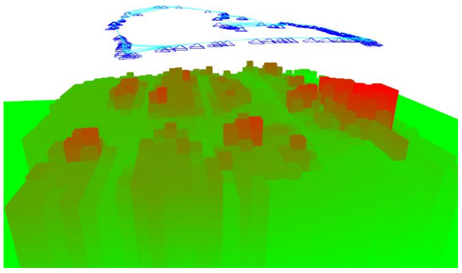
In order to prove the accuracy of the proposed UAV autonomous landing system, we carried out a real-time control experiment of UAV autonomous landing. The landing trajectory and the specific process are shown in Figure 6. It presents our study of the UAV autonomous landing area screening and the entire implementation process of the UAV autonomous landing system. When the UAV was ready to land, flight control switched from normal flight to autonomous landing. The autonomous landing system was started. Then, the system was initialized and the flight began. After the initialization was completed, the pose was estimated, and the closed-loop flight was carried out. Then, the area to be measured was screened. After the selection of the UAV landing area, it began to fall on the selected area.

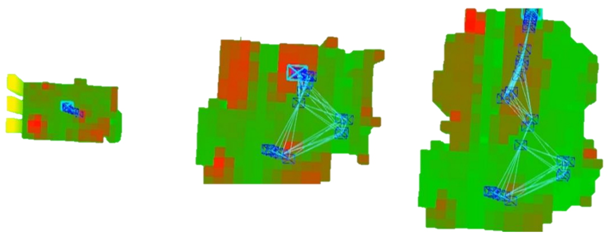


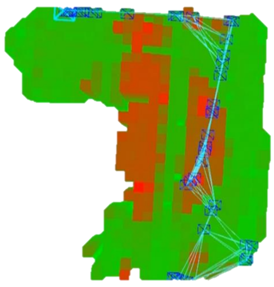
(a) Scenario overview (b) Grid map (c) Part of the scenario

Figure 5. *Cont.*

(d) Side view of scenario


(e) Side view of grid map


(f)Time 1 (g)Time 2 (h)Time 3


(i)Time 4

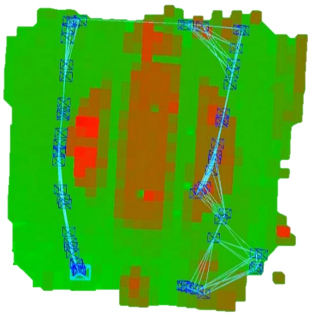

(j)Time 5

Figure 5. The images of (**a**–**c**) detail the correspondence between each part of the two-dimensional height map and the real scene; the image (**d**) is a real scene with different angles; The image (**e**) is the side view of the grid map; the images of (**f**–**j**) are the construction process of the grid map. The blue box above the map shows the trajectory of the drone and the pose of the drone when the key frame is generated.

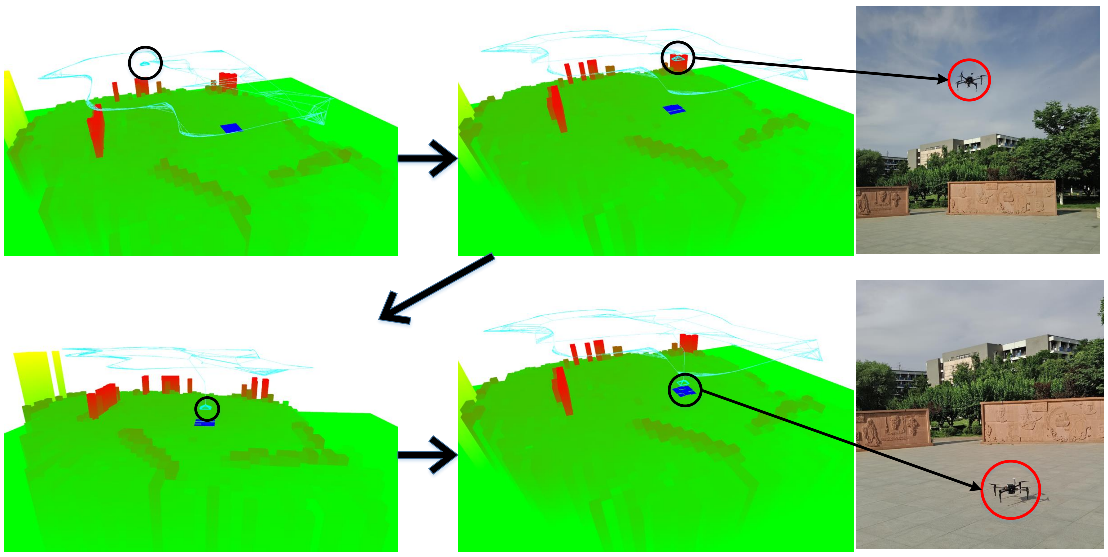
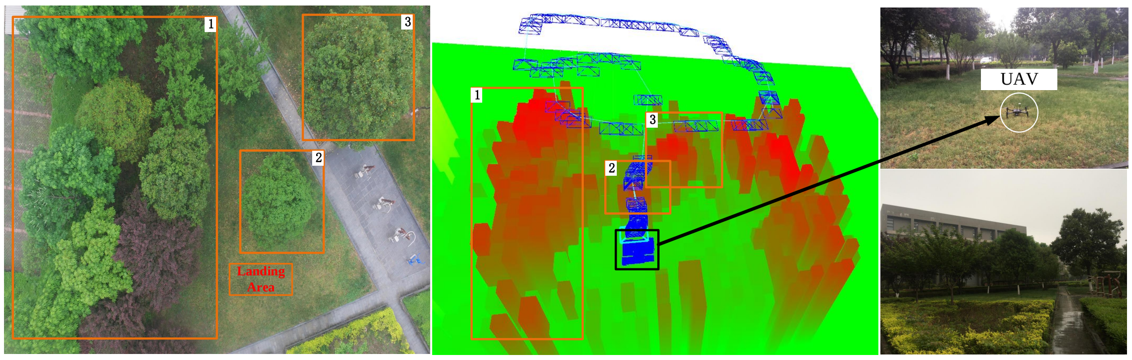


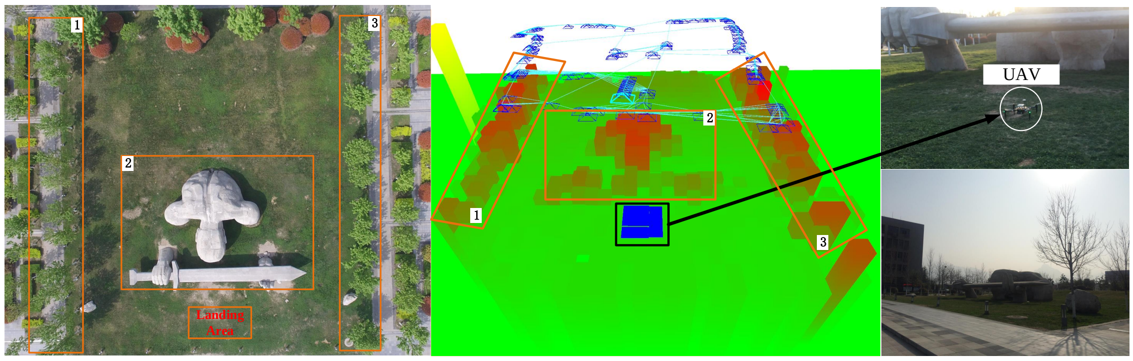
Figure 6. The landing process of a quadrotor and the real-world scene of the process. The color represents the depth of the map. Green represents the lowest point, red represents the highest point, and the middle height is represented by a gradient color. Dark blue indicates a suitable landing site. Light blue indicates the flight path of the aircraft. The black circle refers to the current position of the drone.

3.3. Landing Area Detection in Multi-Scenario

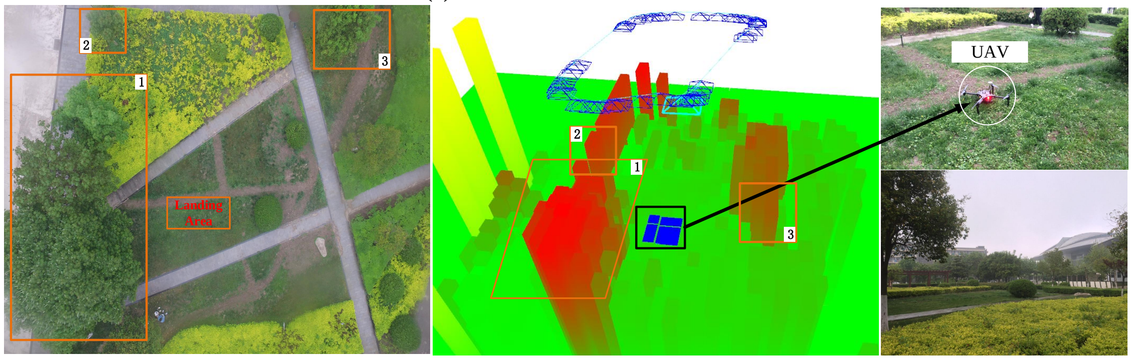
In this paper, several groups of data experiments and analysis were conducted. Five sets among the experimental results are shown in Figure 7. All experimental data were collected from the Northwestern Polytechnical University Chang'an campus. The first scene was a small forest with a suitable landing zone in the middle. The scene for the second set of data acquisition was a gentle slope. Surrounded by trees, a larger humanoid sculpture was located in the middle of the slope. The front part of the slope was relatively flat and suitable for landing. Several trees and shrubs were scattered in the third scene. The middle of the scene was flat and suitable for landing. The scene of the fourth set of data collection was a square with five highly-visible obstacles evenly distributed within it. Three of the obstacles were carved walls, while the others were long stone benches. The middle area of the square was flat and suitable for landing, without any obstacles.



(a) scenario 1 with obstacles of many trees

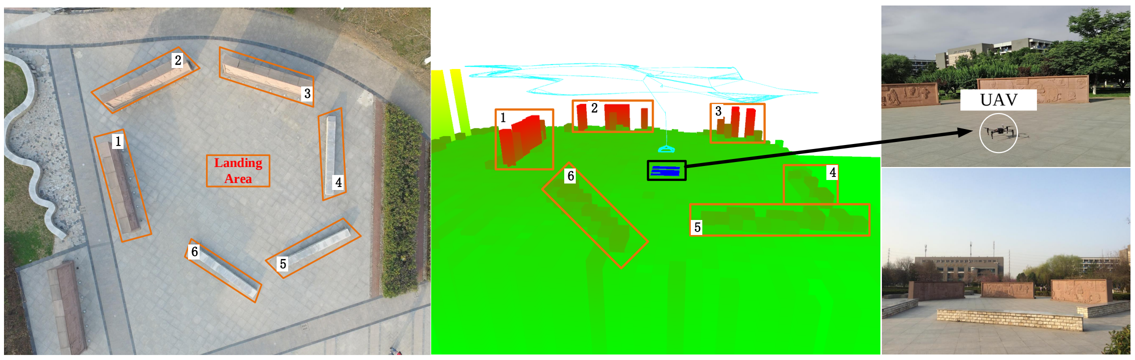


(b) scenario 2 with a hillside and obvious obstacles



(c) scenario 3 with obstacles of trees and shrubs

Figure 7. *Cont.*



(d) scenario 4 with even and noticeable obstacles

Figure 7. Two-dimensional grid height map creation and landing site selection result in five different scenarios. The left column is the real scene, the middle is a two-dimensional grid height map, and the dark blue area is a suitable landing spot. The blue trajectory above the map is the drone flight trajectory, and the right column is the real scene where the drone landed accurately at the landing site.

Figure 7 shows the specific experimental locations and experimental results of the proposed method. The robustness of the proposed system is demonstrated by experimental results in different conditions. The characteristics of the four experimental sites in Figure 7 are different and they simulate different actual application environments.

Scenarios 1, 2, and 3 were experiments simulating field environments. Scenario 1 simulated a forest landscape, with a large areas of trees. There was only a small piece of land suitable for landing in the middle, which our method chose. Scenario 2 simulated wild hillside and boulders. The system chose the flat ground between the hillside and the flat land and avoided obstacles well. Scenario 3 simulated a farm field. Although the shrub area was relatively flat, our method chose a flatter grassland. Scenario 4 simulated the urban environment with regular tall buildings and flat squares. There were even and obvious obstacles distributed in Scenario 4, and the experimental results can be intuitively understood. The landing site selected by our algorithm had the largest integrated distance and was located in the center of the five obstacles.

Take scenario 1 as an example. The drone has a flying height of 20 m. The entire landing process took 1 min and 52 s. Firstly, the drone scans the scene and builds a map. However, due to the high flying height, high scene complexity and the existence of many empty areas (area with large height differences but small areas), the sparse point cloud map constructed by monocular SLAM can not meet the requirements for detecting the landing area of drones. At this point, the grid map proposed in this paper shows its practicality. Based on the three-dimensional point cloud map, the grid map expands the space, fills in the area around the feature point, and realizes the perception of the overall environment. Only in the presence of a coherent flat area, the drone can land. All the revised parts are shown by blue font in the revision.

Experiments showed that, although the three-dimensional point cloud of visual SLAM was estimated to be sparse, the two-dimensional height grid map reconstructed supplied excellent scene information. The map was accurate enough to meet the needs of UAV landing. In various simulation environments and simulation field experiments, the landing sites selected by the proposed method were the safest places in the scenes, and the drone landed accurately in these areas.

For more insights, we invite the reviewer to take a look at the multimedia demonstration UAV_AutoLanding_Demo.mp4 of our system in the Supplementary Materials. The demo is included in the submitted zip file, or you can click here directly (https://page0607.github.io/UAV_Landing/).

4. Conclusions

This paper proposes a novel UAV autonomous landing approach based on monocular visual SLAM. In the proposed approach, we exploit a feature-based method to estimate the drone's pose and attain the fixed point in three-dimensional space. Regarding grid map construction, we first establish an image stream with a corresponding time stamp and barometric altitude information. Then, through the extraction of robust feature points and descriptors, the motion of the drone, the flight path, and the visible three-dimensional point cloud map are obtained by matching and tracking the features. A visible grid map can be built by putting the three-dimensional point cloud to the grid map and calculating the height of each grid after removing noise. After that, a method based on image segmentation is used to divide the height region of the grid map. On the basis of the divided grid map, the system can obtain the appropriate landing place after decision optimization. Finally, the drone takes the shortest path to reach the destination and starts the descent mode near the landing area. After finishing this, the UAV autonomous landing system is successfully executed. Extensive experimental results on multiple real scenes confirmed that the landing area selection and navigation based on visual technology is effective and efficient. In addition, it can be used to find a suitable path in the field of automated driving where unmanned vehicles and UAVs are used in combination. The drone circles in the air to detect the environment, and then a three-dimensional map is constructed to select the appropriate driving path for the unmanned vehicle. Furthermore, the current work shows great potential in various visual domains. In future work, we will consider extending our work to multiple fields and develop it to the general level.

Supplementary Materials: The following are available online at VideoS1:UAV_AutoLanding_Demo.mp4.

Author Contributions: T.Y. and P.L. conceived and designed the algorithm and wrote the paper; H.Z. performed the experiments; J.L. and Z.L. analyzed the data.

Acknowledgments: This work was supported by the National Natural Science Foundation of China under Grant 61672429 and the ShenZhen Science and Technology Foundation under Grant JCYJ20160229172932237.

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

References

1. Jung, Y.; Cho, S.; Shim, D.H. A trajectory-tracking controller design using L1 adaptive control for multi-rotor UAVs. In Proceedings of the 2015 International Conference on Unmanned Aircraft Systems (ICUAS), Denver, CO, USA, 9–12 June 2015; pp. 132–138.
2. Lee, H.; Jung, S.; Shim, D.H. Vision-based UAV landing on the moving vehicle. In Proceedings of the 2016 International Conference on Unmanned Aircraft Systems (ICUAS), Arlington, VA, USA, 7–10 June 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–7.
3. Falanga, D.; Zanchettin, A.; Simovic, A.; Delmerico, J.; Scaramuzza, D. Vision-based Autonomous Quadrotor Landing on a Moving Platform. In Proceedings of the 2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), Shanghai, China, 11–13 October 2017.
4. Kim, J.; Jung, Y.; Lee, D.; Shim, D.H. Outdoor autonomous landing on a moving platform for quadrotors using an omnidirectional camera. In Proceedings of the 2014 International Conference on Unmanned Aircraft Systems (ICUAS), Orlando, FL, USA, 27–30 May 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1243–1252.
5. Jung, Y.; Lee, D.; Bang, H. Close-range vision navigation and guidance for rotary UAV autonomous landing. In Proceedings of the 2015 IEEE International Conference on Automation Science and Engineering (CASE), Gothenburg, Sweden, 24–28 August 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 342–347.
6. Vlantis, P.; Marantos, P.; Bechlioulis, C.P.; Kyriakopoulos, K.J. Quadrotor landing on an inclined platform of a moving ground vehicle. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 2202–2207.
7. Kim, J.W.; Jung, Y.D.; Lee, D.S.; Shim, D.H. Landing Control on a Mobile Platform for Multi-copters using an Omnidirectional Image Sensor. *J. Intell. Robot. Syst.* **2016**, *84*, 1–13. [CrossRef]

8. Forster, C.; Faessler, M.; Fontana, F.; Werlberger, M. Continuous on-board monocular-vision-based elevation mapping applied to autonomous landing of micro aerial vehicles. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 111–118.
9. Yang, T.; Li, G.; Li, J.; Zhang, Y.; Zhang, X.; Zhang, Z.; Li, Z. A ground-based near infrared camera array system for uav auto-landing in GPS-denied environment. *Sensors* **2016**, *16*, 1393. [CrossRef] [PubMed]
10. Saripalli, S.; Montgomery, J.F.; Sukhatme, G. Vision-based autonomous landing of an unmanned aerial vehicle. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '02), Washington, DC, USA, 11–15 May 2002; pp. 2799–2804.
11. Richardson, T.S.; Jones, C.G.; Likhoded, A.; Sparks, E.; Jordan, A.; Cowling, I.; Willcox, S. Automated Vision-based Recovery of a Rotary Wing Unmanned Aerial Vehicle onto a Moving Platform. *J. Field Robot.* **2013**, *30*, 667–684. [CrossRef]
12. Muskardin, T.; Balmer, G.; Wlach, S.; Kondak, K.; Laiacker, M.; Ollero, A. Landing of a fixed-wing UAV on a mobile ground vehicle. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 1237–1242.
13. Lee, D.; Ryan, T.; Kim, H.J. Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA), Saint Paul, MN, USA, 14–18 May 2012; pp. 971–976.
14. Ghamry, K.A.; Dong, Y.; Kamel, M.A.; Zhang, Y. Real-time autonomous take-off, tracking and landing of UAV on a moving UGV platform. In Proceedings of the 2016 24th Mediterranean Conference on Control and Automation (MED), Athens, Greece, 21–24 June 2016.
15. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 10–52. [CrossRef] [PubMed]
16. Kameda, Y. Parallel Tracking and Mapping for Small AR Workspaces (PTAM) Augmented Reality. *J. Inst. Image Inf. Telev. Eng.* **2012**, *66*, 45–51. [CrossRef]
17. Forster, C.; Pizzoli, M.; Scaramuzza, D. SVO: Fast semi-direct monocular visual odometry. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 15–22.
18. Engel, J.; Schops, T.; Cremers, D. LSD-SLAM: Large-Scale Direct Monocular SLAM. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2014; pp. 834–849.
19. Mur-Artal, R.; Tardos, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2016**, *33*, 1255–1262. [CrossRef]
20. Weikersdorfer, D.; Adrian, D.B.; Cremers, D.; Conradt, J. Event-based 3D SLAM with a depth-augmented dynamic vision sensor. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 359–364.
21. Rebecq, H.; Horstschaefer, T.; Gallego, G.; Scaramuzza, D. EVO: A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time. *IEEE Robot. Autom. Lett.* **2017**, *2*, 593–600. [CrossRef]
22. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
23. Rosten, E.; Drummond, T. Fusing points and lines for high performance tracking. In Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV 2005), Beijing, China, 17–21 October 2005; Volume 2, pp. 1508–1515.
24. Calonder, M.; Lepetit, V.; Strecha, C.; Fua, P. BRIEF: Binary robust independent elementary features. In Proceedings of the 11th European Conference on Computer Vision, Crete, Greece, 5–11 September 2010; pp. 778–792.
25. Hartley, R.I. In defense of the eight-point algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.* **1997**, *19*, 580–593. [CrossRef]
26. Kummerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. G^2o: A general framework for graph optimization. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 9–13 May 2011; pp. 3607–3613.
27. Dorian, G.L.; Tardos, J.D. Bags of Binary Words for Fast Place Recognition in Image Sequences. *IEEE Trans. Robot.* **2012**, *28*, 1188–1197.

28. Robertson, S. Understanding inverse document frequency: On theoretical arguments for IDF. *J. Doc.* **2004**, *60*, 503–520. [CrossRef]
29. Nister, D.; Stewenius, H. Scalable recognition with a vocabulary tree. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–22 June 2006; Volume 2, pp. 2161–2168.
30. Comaniciu, D.; Meer, P. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 603–619. [CrossRef]