SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

November 2022, Vol. 65 212202:1–212202:15
https://doi.org/10.1007/s11432-021-3429-1

Robust autonomous landing of UAVs in non-cooperative environments based on comprehensive terrain understanding

Lyujie CHEN[1], Yao XIAO[2], Xiaming YUAN[2], Yiding ZHANG[3] & Jihong ZHU[2*]

[1]*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084*, China;*
[2]*Department of Precision Instrument, Tsinghua University, Beijing* 100084*, China;*
[3]*Department of Automation, Tsinghua University, Beijing* 100084*, China*

Received 25 September 2021/Revised 2 November 2021/Accepted 9 December 2021/Published online 25 October 2022

Abstract Autonomous landing in non-cooperative environments is a key step toward full autonomy of unmanned aerial vehicles (UAVs). Existing studies have addressed this problem by finding a flat area of the ground as the landing place. However, all these methods have poor generalization ability and ignore the semantic feature of the terrain. In this paper, we propose a well-designed binocular-LiDAR sensor system and a robust terrain understanding model to overcome these deficiencies. Our model infers both morphologic and semantic features of the ground by simultaneously performing depth completion and semantic segmentation. Moreover, during the inference phase, it self-evaluates the accuracy of the predicted depth map and dynamically selects the LiDAR accumulation time to ensure accurate depth prediction, which greatly improves the robustness of the UAV in completely unknown environments. Through extensive experiments on our collected low-altitude aerial image dataset and real UAVs, we verified that our model effectively learned two tasks simultaneously and achieved better performance than existing depth estimation-based landing methods. Furthermore, the UAV can robustly select a safe landing site in several complex environments with about 98% accuracy.

Keywords UAV, autonomous landing, terrain understanding, depth completion, semantic segmentation

Citation Chen L J, Xiao Y, Yuan X M, et al. Robust autonomous landing of UAVs in non-cooperative environments based on comprehensive terrain understanding. Sci China Inf Sci, 2022, 65(11): 212202, https://doi.org/10.1007/s11432-021-3429-1

1 Introduction

Autonomous landing is the most fundamental and crucial function of unmanned aerial vehicles (UAVs), which is closely related to various civilian and military missions, such as search and rescue, payload delivery, and emergency landing. Due to the complexity of near-ground situations, it is a very accident-prone task. The autonomous landing in cooperative environments that directly exchanges information through wireless communication [1,2], or manually provides a landing place, such as graphical signs [3–6], two-dimensional bar code signs [7–9], and natural runways [10,11], has been well-studied for decades.

Recently, the rapid development of robotics and perception technology has propelled many studies to explore autonomous landing in non-cooperative environments, of which the core problem is to select a safe landing site. Among most mitigating approaches, the mainstream approach is finding a flat ground as the landing place. Most studies perform depth estimation using onboard sensors, such as the monocular camera [12–14], binocular camera [15,16], or ultrasonic radar [17], to obtain the relative distance between the ground and the UAV. Further, the slope and roughness of the terrain can be calculated. Other studies directly detect the horizontal area of the ground by homography estimation [18] or a deep-learning method [14]. However, these studies have common shortcomings. The primary problem is that due to the lack of reliable range sensors and robust algorithms, these methods cannot self-evaluate and correct their results, which in turn results in poor generalization ability in unfamiliar environments and crashing of

* Corresponding author (email: jihong_zhu@hotmail.com)

Figure 1 (Color online) The overview of our work. The UAV predicts a dense depth map and a semantic segmentation map based on the image and sparse depth measurement acquired by the onboard camera and LiDAR, thereby selecting the safe landing site.

drones. More importantly, only recognizing the flatness of the terrain can lead UAVs to choose the wrong landing sites (e.g., calm lake) or land on suboptimal places (e.g., grassland with unnoticeable stones or the roof of a building).

In this paper, we propose a well-designed sensor system and robust terrain understanding model that simultaneously infers the flatness and safeness of the ground to overcome these problems, illustrated in Figure 1. Specifically, we built a UAV platform equipped with a lightweight and high-precision binocular-LiDAR system. The LiDAR has an advantage in that the areas scanned inside the field of view grow over time until 100% coverage. Thus, we proposed a model named TerrainNet to achieve a comprehensive terrain understanding from both morphologic and semantic aspects by simultaneously performing depth completion and weakly supervised semantic segmentation. During the inference phase, our model self-evaluated the accuracy of the predicted depth map and dynamically selected the LiDAR accumulation time. It adaptively balanced the accuracy and speed while ensuring accurate depth estimation. This feature greatly improved the model robustness and guaranteed that the UAV can safely land in a completely unknown environment.

To train this model, we exploit our UAV platform to collect a new low-altitude aerial image dataset, containing about 30000 automatically sparsely annotated depth maps and 3000 manually annotated semantic segmentation maps using scribbles. Based on this dataset, we verified that TerrainNet can effectively learn two tasks simultaneously, even with unbalanced data size, and obtain better results than separately learning each task. Using a carefully designed network structure, loss function, and the multitask learning paradigm, our model outperformed existing depth estimation-based landing methods. Our landing site selection method is validated to successfully find a safe area in different complex environments, with an accuracy of up to 98%. Moreover, we conducted complete autonomous landing experiments on real UAVs, verifying the effectiveness and robustness of our overall landing strategy. The detailed video is available at the website[1)].

The contribution of this article can be summarized as follows.

(1) We for the first time addressed the autonomous landing problem in non-cooperative environments

1) https://youtu.be/zeYe-tihIwQ.

by considering both the morphologic and semantic features of the ground.

(2) We proposed a robust inference strategy that ensures an accurate depth map and adaptively balances the accuracy and speed of our model.

(3) We collected the first low-altitude aerial image dataset with both weakly annotated depth maps and semantic segmentation maps.

(4) We applied our landing method to real UAVs and achieved safe landing in various complex environments.

2 Related work

2.1 Autonomous landing in non-cooperative environments

The autonomous landing problem in non-cooperative environments investigates how to select a safe landing site without guidance. Most studies conduct depth estimation or image segmentation to find a flat area for landing. For example, Ref. [17] perceived the depth using ultrasonic sensors for UAV safe landing at a low altitude of 150 cm. Refs. [15, 16] estimated the altitude of drones in indoor scenes using a stereo vision system. Refs. [12, 13] applied traditional visual methods to realize a monocular depth prediction. Ref. [14] proposed deep-learning methods to estimate the depth and orientation of the plane in an end-to-end way. Ref. [18] proposed the combination of a homography estimation and a strategy of adaptive thresholding of correlation scores to segment the plane area. Another series of methods have addressed this problem by estimating the safeness or surface type of the ground. For example, Refs. [19, 20] directly performed patch-level classification to evaluate the danger of the regions. Ref. [21] conducted regional segmentation and classification to recognize the grassland. Besides, some other factors have been considered for this problem. Ref. [16] considered the energy consumption of the drone. Ref. [22] applied geographic information system data to estimate the landing position. Ref. [23] tracked moving vehicles using aerial videos. In general, these existing methods considered only a single characteristic of the terrain. However, in this paper, we discussed a comprehensive terrain understanding from both morphologic and semantic aspects.

2.2 Depth completion

Depth completion [24–29] refers to completing the depth estimation given a sparse depth map and corresponding RGB image. It has been widely studied and used for robotics and unmanned vehicles. Benefiting from the partially known depth values, the performance of depth completion significantly outperforms methods using monocular depth prediction and binocular depth matching. Several studies have investigated depth completion in aerial scenes. For example, Ref. [30] used depth completion on sparse feature maps to enable place recognition at extreme viewpoint changes. Ref. [31] performed both depth completion and confidence estimation using an image-guided approach. However, these studies are implemented on virtual datasets with no real range sensors. In this paper, we carefully built our UAV platform and perception system to use high-precision LiDAR for depth completion in real environments.

2.3 Weakly supervised semantic segmentation

Weakly supervised semantic segmentation uses alternative annotations, such as image-level tags [32–34], labeled points [35], scribbles [36, 37], and bounding boxes [38, 39] to learn segmentation tasks. Among the several methods, image-level tags and bounding box annotations are more suitable for training images that have clear foreground objects. To adapt to our semantic segmentation task for ground surface type, we employed the scribble-style annotation, which is a highly efficient and relatively low-cost solution. Previously [36, 40], scribbles were used to generate fake full masks and to train models based on unreliable ground truth. However, recent studies [37, 41] are prone to directly use cross-entropy loss for partially labeled pixels.

3 System overview

In this section, we briefly introduce our UAV platform as well as the onboard perception system, based on which we construct a new aerial image dataset and propose our terrain understanding methods. The

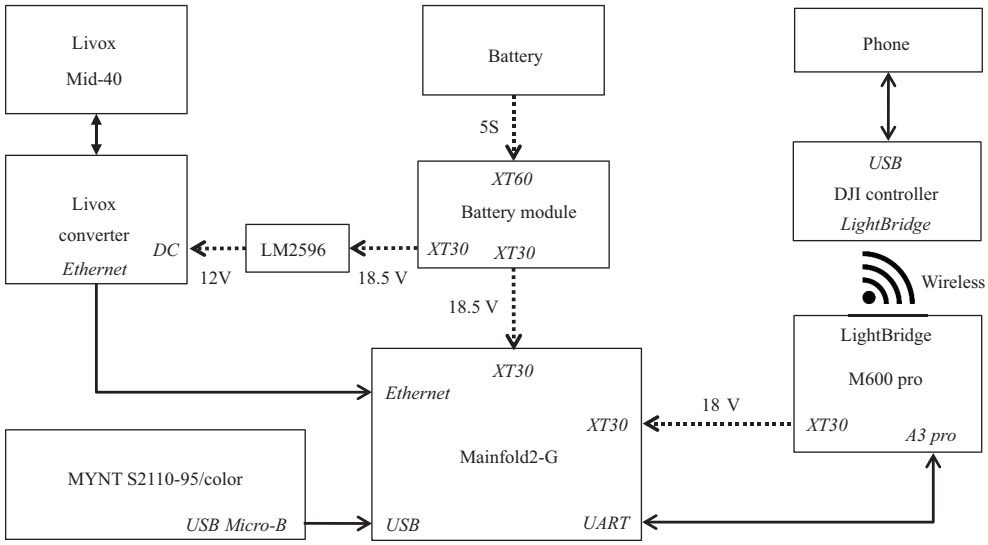

Figure 2 UAV hardware system overview. The solid arrowed lines indicate the data communication between each module and the interfaces are marked in italic. The dotted arrowed lines represent the connection of the power supply.

illustration of connection and communication between each module is shown in Figure 2.

3.1 UAV platform

The flight platform we used in this paper is DJI Matrice 600 Pro, a commercial hexacopter UAV equipped with a carbon-fiber airframe, the A3 Pro flight controller, Lightbridge 2 HD transmission system, six 4500 mAh intelligent batteries, and a built-in battery management system. The vehicle is 1133 mm in the diagonal wheelbase and weighs 9.5 kg including batteries. Its maximum payload capacity is about 6 kg. The flight time in hovering is about 16 min with full payload and 32 min without payload. The A3 Pro Flight Controller consists of a flight controller, three GPS-Compass Pro, two IMU Pro, and a power management unit (PMU), which provides triple modular redundancy, improving the system's anti-risk performance. Self-adaptive systems will automatically adjust flight parameters based on different payloads. Other sensors and devices are mounted on our designed two-layer carbon-fiber board at the bottom of the center airframe.

3.2 LiDAR

We adopt Livox Mid-40 as our onboard LiDAR because it is lightweight (760 g), low-cost ($599), and has two unique advantages fit for UAV landing scenes. First, it detects toward one side, thus very suitable to downward mount on the UAV. Another advantage is that it has a non-repetitive scanning pattern in which the areas scanned inside the field of view (FOV) grow the longer the integration time, increasing the likelihood of objects and other details within the FOV being detected. Its coverage performance is equivalent to the 32-line and 64-line products when the integration time is 0.1 and 0.5 s, respectively. As integration time increases, the FOV coverage will approach 100% ($\approx$ 1.5 s). This feature enables UAVs to dynamically choose different LiDAR integration times according to the needs of the algorithm.

3.3 Stereo camera

In this paper, although we use the fusion of monocular image and LiDAR points for depth completion, the binocular camera is still necessary. It can provide more self-supervision during the training phase, and more importantly, it helps to evaluate the accuracy of the predicted depth map during the inference phase, which is a key component of our robust landing strategy. More details are provided in Subsection 4.3. For this reason, we chose MYNT EYE Standard/Color (S2110-95/Color) binocular camera, which provides a stereo color image with a resolution of 1280×400@60 FPS. The baseline length is 8 cm and the FOV angle is 95°. To ensure the consistent quality of stereo images, the camera provides automatic image signal processing (ISP) exposure and automatic white balance functions. The global shutter function also effectively reduces the image distortion during rapid shooting scenes. At the same time, the camera

provides hardware-level time synchronization and binocular frame synchronization at millisecond-level accuracy.

3.4 Onboard computer

To accelerate the inference speed of the depth completion model with modern GPU, we choose DJI Manifold 2-G as the onboard computer. Its processor is the NVIDIA Jetson TX2, which is built around an NVIDIA GPU with 256 CUDA cores and it has 8 GB 128-bit LPDDR4 memory. The added UART ports are used to connect with the A3 Pro flight controller for accessing the flight status and manipulating the UAV. Since Manifold 2-G supports two external power supplies independently and can automatically choose the power source with a higher voltage, we use both a reserved output 18V power supply from the Matrice 600 Pro and a separate 5300 mA 5S LiPo battery to power the computer. In this way, Manifold 2-G will prefer the separate battery because it has a higher voltage. This not only saves more power to the vehicle but also ensures the availability of the onboard computer when the separate battery fails. In the whole system, all sensors except LiDAR support global time synchronization, so we use precision time protocol (PTP) to synchronize the timestamp of the onboard computer and LiDAR.

3.5 Others

The DJI power distribution unit is connected to the separate battery to power the Manifold 2-G and LiDAR since its electromagnetic compatibility can reduce interference to the GNSS or Wi-Fi signal from the power supply. However, because the battery voltage is higher than the working requirement of LiDAR, we use a DC-DC voltage converter module (LM2596) to provide 12 V power for LiDAR.

Through the LightBridge HD transmission system, we can monitor the status of UAV and the algorithm result of the onboard computer in real time. Also, we develop a mobile app based on DJI UXSDK to switch the flight mode of UAV, i.e., the auto landing mode and the manual control mode.

The software has been developed in C++ and Python as ROS (robot operating system) package, using the well known OpenCV libraries to process the images and PyTorch for model inference.

4 Robust landing site selection

In this section, we first introduce a method of calibration and alignment between stereo cameras and LiDAR to generate synchronized images and depth maps. Then we detail the network structure and training losses for two terrain understanding tasks. For the model inference, we propose a self-evaluation method to assess the depth prediction results. Based on it, the inference strategy is improved to balance the speed and accuracy of the model. Finally, with the accurate depth and semantic segmentation results, a robust landing site selection method is derived.

4.1 Calibration and alignment

The calibration and alignment are the basis for dataset construction and depth completion algorithm. The intrinsic and extrinsic parameters of the stereo camera are calculated using [42], while the calibration between LiDAR and the camera needs a special conversion. Exploiting the advantage of high FOV coverage of Livox Mid-40, we project a few seconds of LiDAR points of a static scene to a virtual image plane and allocate the grayscale colors based on their intensity to form an intensity-based LiDAR image. The corner points of the calibration chessboard from the LiDAR image and camera image respectively correspond to the 3D points in the world and the 2D projections. This constructs a perspective-n-point (PnP) problem [43], which can be effectively resolved to acquire the rotation matrix $R_{c,l}$ and the translation $t_{c,l}$ between the camera and LiDAR frame.

To generate the depth map corresponding to image I_0 captured at time t_0, we first perform time alignment to transform LiDAR data of several seconds to the LiDAR coordinate system l_0 at time t_0. Specifically, given a LiDAR point p^{l_1} measured at time t_1, we can get p^{l_0} by using the transformation as $p^{l_0} = T_{l_0,l_1} p^{l_1}$, in which

$$T_{l_0,l_1} = T_{l_0,w} T_{w,l_1} = T_{l,i} T_{i_0,w} T_{w,i_1} T_{i,l} = T_{l,i} T_{w,i_0}^{-1} T_{w,i_1} T_{l,i}^{-1}, \tag{1}$$

where $T_{l,i}$ is a fixed transformation matrix between LiDAR and the inertial measurement unit (IMU), and T_{w,i_0}, T_{w,i_1} are the position and attitude of the IMU in the world coordinate system at time t_0, t_1.

(a) (b) (c)

Figure 3 (Color online) Calibration between LiDAR and the camera. Corner points in the intensity-based LiDAR image (a) and the camera image (b); (c) generated depth map corresponding to the camera image.

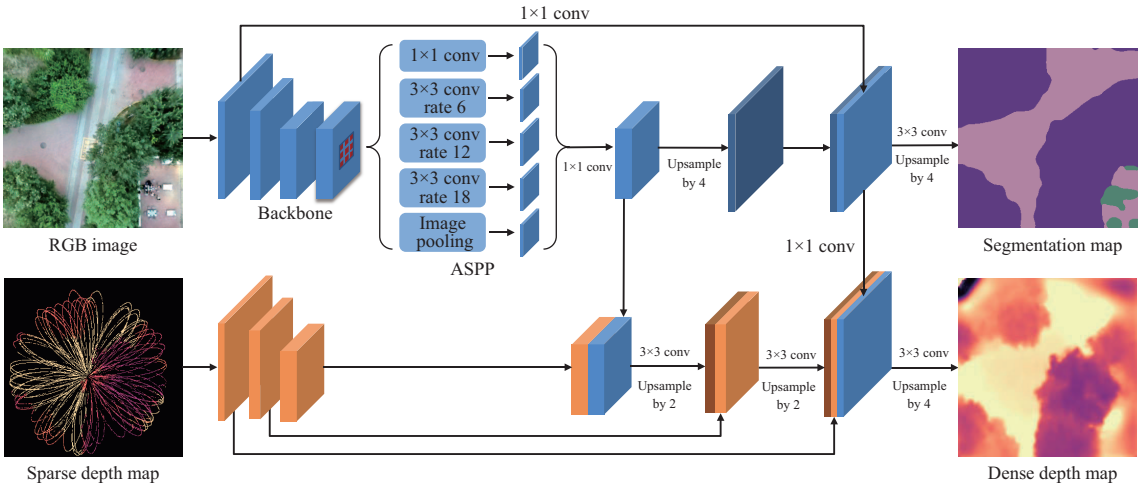

Figure 4 (Color online) The network structure of TerrainNet.

However, since the data acquisition frequency of various sensors is different, T_{w,i_0}, T_{w,i_1} are obtained by conducting a linear interpolation and a spherical linear interpolation to GPS and IMU data, respectively.

The second step is to spatially align the LiDAR point to the camera. Given a LiDAR point $p^l = (x^l, y^l, z^l)^{\mathrm{T}}$, its coordinate in the camera coordinate system is $p^c = (x^c, y^c, z^c)^{\mathrm{T}} = R_{c,l} \cdot p^l + t_{c,l}$, the corresponding projection coordinate in the image plane is $(x, y) = (f \cdot x^c/z^c + c_x, f \cdot y^c/z^c + c_y)$, where f is the focal length in pixel unit and (c_x, c_y) is the principal center point. Since x, y are generally not the integer, we assign it to the nearest pixel (u, v). If one pixel is allocated multiple times, we only select the point closest to the image in time. The generated depth map is shown in Figure 3.

4.2 Terrain understanding

We tackle the terrain understanding problem by learning two tasks, i.e., depth completion and weakly supervised semantic segmentation.

Network structure. Since both tasks produce pixel-level results, we construct our network on the basis of DeepLabv3+ [44]. As shown in Figure 4, we use two separate branches to estimate the depth map and semantic segmentation map respectively after a shared representation produced by a mutual backbone.

For the depth completion branch, we use a shallow encoder to process the input of the sparse depth map and then perform late fusion with the image feature produced by the ASPP module. As the feature map continues to be upsampled, the low-level features of both the image and the sparse depth map are gradually fused for estimating the final dense depth map.

For the semantic branch, we use the original decoder of DeepLabv3+. According to the safety levels, six terrain semantic categories are chosen, i.e., paved area, vegetation, obstacle, vehicle, building, and water. We combine trees and grass as 'vegetation' since it is too hard to manually separate them in the aerial images. Also, through the experiments on separately labeled data, we find that our model cannot

implicitly distinguish them well even with depth information. However, trees and grass have different safety levels, so we reckon them as a single category in network training while separating them in the post-processing with altitude differences, which achieves better results.

This network structure greatly reduces the model parameters and accelerates the inference speed. What is more, these two tasks are predicted at the same time, so it also eliminates the need to align the model results produced at different times.

Training losses. The training losses C of our network are composed of C_D for the depth completion task and C_S for the semantic segmentation task. Specifically, given input image as well as its corresponding sparse depth d, predicted depth p_d, and the ground truth depth map g_d, we define C_D as a combination of four terms,

$$C_D = C_d + C_r + C_p + C_s, \tag{2}$$

where C_d, C_r, C_p, C_s are depth loss, depth ratio loss, photometric loss, and smoothness loss, respectively. The weakly supervised depth loss C_d penalizes the differences between the network output and the ground truth depth map on the set of pixels with known sparse depth, which helps the training for fast convergence, defined as

$$C_d = \left\| \mathbb{1}_{\{g_d>0\}} \cdot (p_d - g_d) \right\|_2^2. \tag{3}$$

In addition, we propose a new depth ratio loss specialized for landing scenarios. During the landing process, we expect the predicted depth map to be increasingly fine-grained as the altitude goes down. Therefore, the depth error with the same absolute value should attach more supervision at a lower altitude, which can be realized by penalizing the proportion of the difference between the predicted and the ground truth value. Hence, C_r is defined as

$$C_r = \left\| \mathbb{1}_{\{g_d>0\}} \cdot \frac{p_d - g_d}{g_d} \right\|_1. \tag{4}$$

The photometric loss C_p is to indirectly supervise the predicted depth by comparing the image similarity between the current image and the reconstruction image inversely warped from the nearby frame. Supposing the current image is captured by left side of the stereo camera at time t_i denoted by $I_{l,i}$, we choose the adjacent two frames on the same side $I_{l,i-1}, I_{l,i+1}$ and the image with the same timestamp captured by the other side of the stereo camera $I_{r,i}$ as the nearby frames. The relative pose between stereo cameras is fixed while the transformation matrix between the adjacent frames can be calculated based on known flight position and attitude with the method elaborated in Subsection 4.1. With relative poses denoted by $T_{i-1,i}$, $T_{i+1,i}$, and $T_{r,l}$, the reconstruction images can be obtained as $I'_{l,i-1}$, $I'_{l,i+1}$, and $I'_{r,i}$ referred to [45]. So the photometric loss is

$$C_p = \frac{1}{3} \sum_s \alpha \frac{1 - \mathrm{SSIM}(I'_s, I_s)}{2} + (1-\alpha) \left\| I'_s - I_s \right\|_1, \tag{5}$$

where $s \in \{\{l, i-1\}, \{l, i+1\}, \{r, i\}\}$ and $\alpha = 0.85$. We use simplified structural similarity (SSIM) [46] with a 3×3 block filter.

The smoothness loss C_s forces a neighboring constraint to encourage depth to be locally smooth, which is the basis to achieve the stable terrain features with depth derivatives. Here we penalize the first and second-order derivatives of the depth predictions as

$$C_s = \left\| \nabla p_d \right\|_1 + \left\| \nabla^2 p_d \right\|_1. \tag{6}$$

For the semantic segmentation task, we follow the previous studies [37,41] to use partial cross entropy loss. Formally, given the predicted segmentation map p_s, the scribble-style ground truth g_s, and the labeled pixel set Ω of g_s, C_S can be defined as

$$C_S = \sum_{\Omega} H(p_s, g_s) = -\sum_{\Omega} p_s \log(g_s). \tag{7}$$

It is worth noting that the labeled data sizes of these two tasks in our dataset are extremely unbalanced, so we introduce a loss weight λ to stabilize the multi-task learning. Finally, the whole training loss C is defined as

$$C = 1/(\lambda+1) \cdot C_D + \lambda/(\lambda+1) \cdot C_S. \tag{8}$$

4.3 Model inference strategy

So far, although we have carefully designed the network structure and training losses, due to the insufficient generalization ability of deep learning-based methods, our model still cannot guarantee good performance in all environments. However, for the autonomous landing task, imprecise results, especially the inaccurate depth prediction, will result in a fatal UAV crash. Therefore, it is necessary to improve the robustness of the model in the inference stage, which is often ignored by existing methods. Our LiDAR can obtain accurate and high-coverage depth maps, but it takes too much time ($\approx 50\times$ slower than network inference) to integrate data. Hence, to balance the speed and accuracy of our model, we explore a self-evaluation method for assessing the accuracy of the depth estimation in the inference phase, and dynamically adjusting the accumulation time of the LiDAR. This ensures that the UAV can stably avoid obstacles in unknown conditions but also efficiently and quickly land in familiar environments.

Self-evaluation method. The self-evaluation method is proposed based on the intrinsic property of the binocular camera. Specifically, similar to the principle of photometric loss, given the stereo images I_l, I_r, the predicted depth p_d according to I_l, the transformed sparse depth d, and the transformation matrix $T_{r,l}$, we can generate the reconstructed right image $I_r^{p_d}$. Without considering the imaging difference between two cameras brought by hardware and occlusion, if the predicted depth value is completely correct, the image similarity $\text{sim}_{p_d} = \text{SSIM}(I_r^{p_d}, I_r)$ will approach to 1. Otherwise, sim_{p_d} will be small. Therefore, given $\text{sim}_{lr} = \text{SSIM}(I_l, I_r)$ and $\text{sim}_d = \text{SSIM}(I_r^d, I_r)$ as lower and upper bounds, we can evaluate p_d by comparing the relative size between sim_{p_d} and sim_{lr}, sim_d. Concluding through the statistic, when $\text{sim}_{p_d} > (\text{sim}_{lr} + \text{sim}_d)/2$ or $\text{sim}_{p_d} > \text{sim}_{lr} + 0.1$, the predicted depth is accurate enough for selecting the landing place.

Inference process. Using this evaluation method, the improved inference process is realized. In our system, the camera and the LiDAR produce the data at the frequency of 20 and 10 Hz, respectively. Every time we get an image, a coarse depth is first predicted. As the LiDAR data continue to be acquired, the image and gradually dense depth input are combined to refine the output depth map. After each step, the prediction result is evaluated until the depth map is accurate enough. Generally, after about one second of accumulation of LiDAR, the FOV coverage is more than 80%. If the result still does not meet the accuracy requirement, we directly perform the nearest neighbor interpolation on the sparse depth and set the depth outside the FOV area as unknown.

4.4 Landing site selection

After obtaining the accurate depth and semantic segmentation maps, we convert these two results into the flatness mask and safeness mask of the ground.

Specifically, the flatness mask is generated through the following three steps. (1) We first convert the original 'perspective' depth p_d to the 'plane' depth p_p where all points in a plane parallel to the camera have the same depth. (2) We then perform a perspective processing on p_p by correcting the roll and pitch to zero to obtain the depth map p_c that simulates the UAV is parallel to the ground. (3) The first and second-order derivatives of depth map p_c represent the slope and roughness of the ground. We define that a flat area should satisfy $\nabla p_c < a_s$ and $\nabla^2 p_c < a_r$, where a_s and a_r are the maximum acceptable slope angle and roughness angle of the landing plane. According to this formula, a binary mask m_d that indicates the flatness of the ground is calculated.

Similarly, the safeness mask is transformed through two steps. (1) We first perform the same perspective processing on p_s as the second step of the depth map to obtain the aligned segmentation map p_a. (2) Among the six terrain types for the semantic segmentation task, the paved area is the most suitable region for landing. So the safeness mask m_s is generated by setting the pixels of the paved area as 1.

The landing mask m_l is produced by merging two masks through AND operation. Finally, we choose the largest inscribed circle of the landing area as the candidate landing place. The real radius R of this circle can be calculated with the 'plane' depth c of the circle center and the radius r in the image by $R = c/f \times r$, where f is the focal length. Referred to the UAV size, if $R > 2m$, we hold that the candidate area is big enough and the center of the circle is selected as the safe landing site. It should be noted that if the depth map is only generated by the sparse depth map, it means that the current environment and the training scenes in our dataset are obviously different. Therefore, we will not use the segmentation result and the landing site is just calculated by flatness mask m_d.

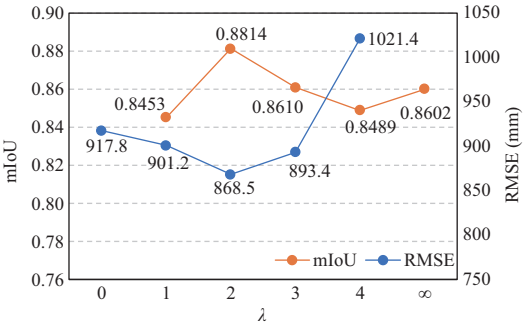

Figure 5 (Color online) Influence of different ratios of loss weight λ between two tasks. 0 and ∞ indicate only training with C_D and C_S, respectively.

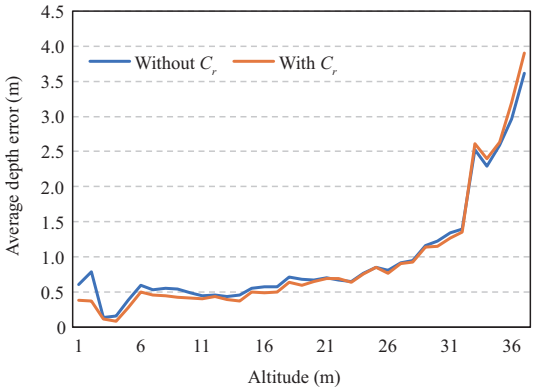

Figure 6 (Color online) Average depth error in different altitudes of models trained with and without C_r.

5 Experiment

5.1 Setup

Dataset. We collected 10 real flight records in different environments using onboard sensors. By performing spatial-and-temporal alignment described in Subsection 4.1 with high-frequency GPS and IMU data, 30000 sparsely annotated depth maps were automatically generated. To reduce the cost of manual labeling of semantic segmentation maps, we partially annotated 3000 images with scribble-style semantic ground truth. Among them, we select 8 records as the training set and the remaining 2 as the test set. The scenes covered by these clips are completely different. So it can effectively verify the algorithm.

Metrics. Except for the commonly used metrics such as root mean square error (RMSE), absolute relative (REL), and δ in the depth estimation problem [47] and mean intersection over union (mIoU) in the semantic segmentation problem [48], we also introduce a metric acc_L describing the accuracy of selecting the safe landing site. It is computed on all 5000 images of the test set which is a convincing averaged result. If the selected place is the paved area and is flat within 2 m radius, then $\text{acc}_L = 1$. Otherwise, $\text{acc}_L = 0$. We use the ground truth depth map to evaluate the flatness and manually judge the semantic type of the selected location. The final results are also manually double-checked.

Implementation details. To match the FOV range of the camera and the LiDAR, input images and sparse depth maps are center-cropped to a size of 352×352. For the experiments of evaluating the model performance, 50% of LiDAR points are transformed into a sparse depth map during both training and test phase. While in the real flight test, the percentage of LiDAR data is automatically determined by our improved inference strategy. We train the model for 80 epochs using Adam [49] optimizer with an initial learning rate of 1E−5 for backbone and 1E−4 for ASPP module and decoders. The backbone of our model in real flight tests is ResNet-18 while in the rest experiments it is ResNet-101 [50].

5.2 Result on TerrainNet

Influence of λ. For our multi-task learning problem with unbalanced data size, the choice of loss weight is very important. So we evaluate the RMSE and mIoU of the two tasks under different values of λ to find the optimal loss weight for training our model. Except for normal values, we also train our model only with C_D and C_S, denoted by $\lambda = 0$ and $\lambda = \infty$. It can be seen from Figure 5 that the lowest RMSE and highest mIoU are both achieved when $\lambda = 2$, which is also better than separately training two tasks, validating that our multi-task learning strategy is effective and provides more supervision to each task.

Compare to existing methods and ablation study. The mainstream landing methods generally detect the plane area by measuring the relative distance, so we then evaluate the depth prediction result of our model by comparing it with a novel depth estimation method for UAV landing [14] and a widely-used depth completion method [51]. The backbone models of them are both ResNet-101. To make a fair comparison and figure out the reason for accuracy improvements, we also conduct an ablation study. It can be seen from Table 1 that by using the multi-task learning strategy, the RMSE consistently decreases with and without LiDAR input. Compared with SafeUAV [14] that is implemented with the same network

Table 1 Comparison of different methods and the ablation study of TerrainNet[a)]

Method	DeepLabv3+	C_p, C_s	C_r	Multi-task	LiDAR	RMSE (mm) ↓
SafeUAV [14]	✓	–	–	–	–	3340.6
Ma et al. [51]	–	✓	–	–	✓	951.2
TerrainNet	✓	✓	✓	–	–	3151.3
	✓	✓	✓	✓	–	2953.1
	✓	✓	–	–	✓	926.3
	✓	✓	✓	–	✓	917.8
	✓	✓	✓	✓	✓	868.5

a) All methods use basic depth loss C_d. 'DeepLabv3+' represents using DeepLabv3+ [44] as a backbone model. 'Multi-task' indicates learning with the semantic segmentation task. 'LiDAR' represents using the sparse depth map input to perform depth completion. ↓ indicates the lower is better.

Table 2 Performance of the depth prediction results under different densities of LiDAR input[a)]

Density	RMSE (mm) ↓	REL (mm) ↓	$\delta < 0.25$ ↑	SSIM ↑
0	2953.1	185.4	75.6%	0.595
0.1	1573.5	80.9	93.1%	0.624
0.5	868.5	42.0	97.9%	0.656
1.0	798.0	37.9	98.1%	0.662

a) ↓ indicates the lower is better and ↑ indicates the higher is better.

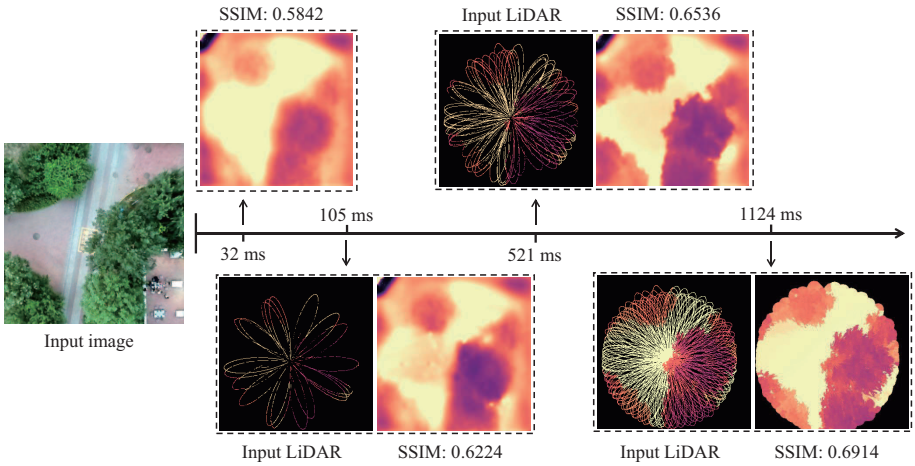

Figure 7 (Color online) As time progresses, the depth estimation becomes increasingly accurate.

structure and basic depth loss C_d, TerrainNet yields better performance owing to the use of more types of loss functions. While for the comparison with Ma et al. [51], the introduction of dedicated loss C_r and a more efficient network jointly improve the accuracy of our model. Further, we calculate the average depth error at different ground truth altitudes between lines 5 and 6 in Table 1 to verify the necessity of C_r. As shown in Figure 6, although the average depth error at high altitudes is slightly larger, the model trained with C_r performs better when altitude is small and the overall RMSE has decreased. This result is consistent with our expectations in Subsection 4.2, which ensures that the depth estimation results of the UAV will gradually be accurate during the landing process.

Inference strategy. Beyond the better learning performance of our method, the more important advantage of TerrainNet is its robust inference strategy mentioned in Subsection 4.3. The premise for this strategy to be feasible is that the accuracy of the predicted depth map should be increasingly better as the density of LiDAR increases. To this end, we compare the influence of the density of LiDAR input. We use RMSE, REL, and δ to directly assess the depth map results and also introduce the average SSIM as a metric to indirectly evaluate the model through the quality of the reconstructed images. The numerical results are reported in Table 2 and a visualized timeline of the prediction is illustrated in Figure 7. When using a raw image without LiDAR data for prediction, the depth map can distinguish obvious altitude differences, such as trees and ground. As the density of the LiDAR increases, every metric is becoming better and the predicted depth map is more fine-grained. After about one second of

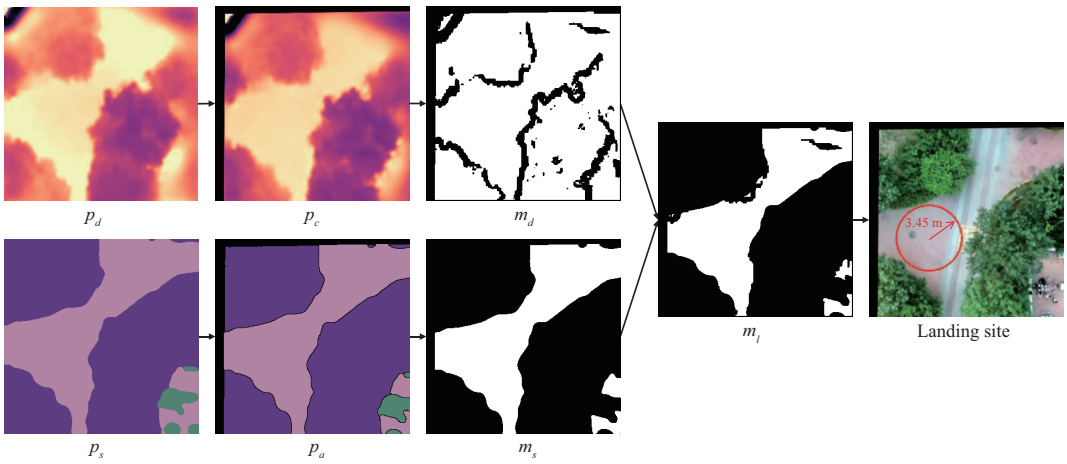

Figure 8 (Color online) Process of selecting the safe landing site.

Figure 9 (Color online) Selected landing sites in real scenes.

accumulation, we directly interpolate the sparse depth map to get the most refined prediction map. This reduces the perception range of the UAV but makes it possible to identify minor height differences.

5.3 Comparison of landing site selection strategies

After verifying that TerrainNet can acquire accurate prediction results, we then evaluate the effectiveness of our landing site selection strategy. Two angle thresholds are set to be $a_s = 10.0, a_r = 10.0$. The selection process is visualized in Figure 8, in which we perform a simple erosion and dilation to remove minor noises. The typical safe landing sites selected in various real complex environments are shown in Figure 9. To demonstrate the indispensability of morphologic and semantic features of the ground, we then compared the difference of the selected landing sites in various environments with different masks, i.e., m_d, m_s, m_l. Numerically, if we only consider the flatness of the ground as the existing methods do, the mean accuracy of choosing the safe landing sites acc_L is 86%. However, when further taking the semantic information into account, this value significantly boosts to 98%. Only very few hard-to-detect twigs and obstacles cause the wrong choices. As shown in Figure 10, by considering the semantic information, the UAV can avoid lakes or grassland, and always choose the most reliable hard paved ground. Similarly, the precise depth map helps to identify hard-to-segmented small obstacles and recognize the altitude changes within the same types of ground areas, such as ramps, road curbs, and inlets, as shown in Figure 11. The

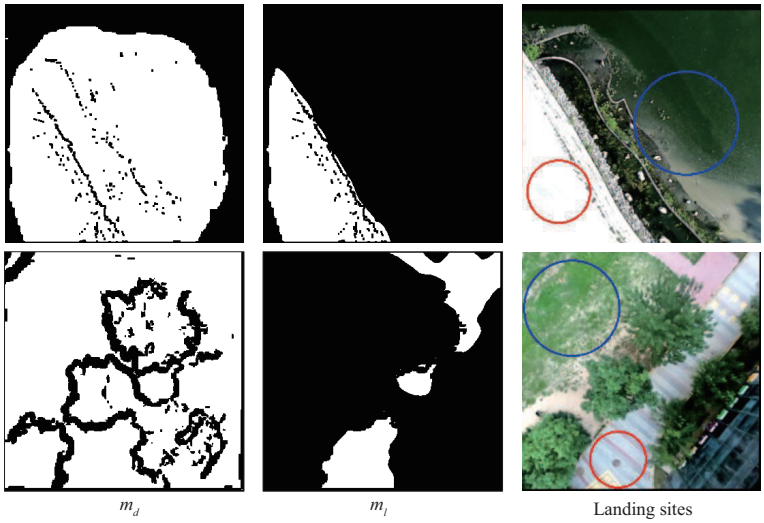

Figure 10 (Color online) Comparison of selected landing sites using the flatness mask (blue circle) and the landing mask (red circle).

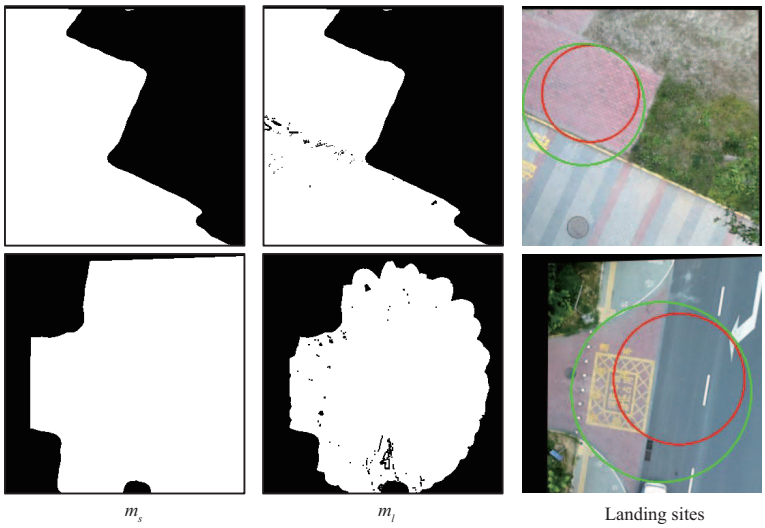

Figure 11 (Color online) Comparison of selected landing sites using the safeness mask (green circle) and the landing mask (red circle).

Table 3 Comparison of difference backbones[a)]

Backbone	RMSE (mm) ↓	REL (mm) ↓	$\delta < 0.25$ ↑	mIoU ↑	acc_L ↑
ResNet-18	877.7	41.3	97.8%	0.7964	95%
ResNet-101	868.5	42.0	97.9%	0.8814	98%

a) ↓ indicates the lower is better and ↑ indicates the higher is better.

results demonstrate that our landing site selection method is more reliable.

5.4 Real flight test

Next, we apply our model and landing site selection method to the real UAV. Due to the limitation of the capacity of the onboard computer, ResNet-18 is used as the backbone of the model in real flight tests. We compare the performance with different backbones in Table 3. It can be seen that the difference in the depth completion task is very small, which demonstrates that the design of separating the image branch and the sparse depth branch is more robust. Although the mIoU of the semantic segmentation task has dropped, it still maintains a very high accuracy of the landing site selection. As shown in Figure 12, we adopt a variety of measures to ensure a robust and stable flight in the real landing process. Specifically,

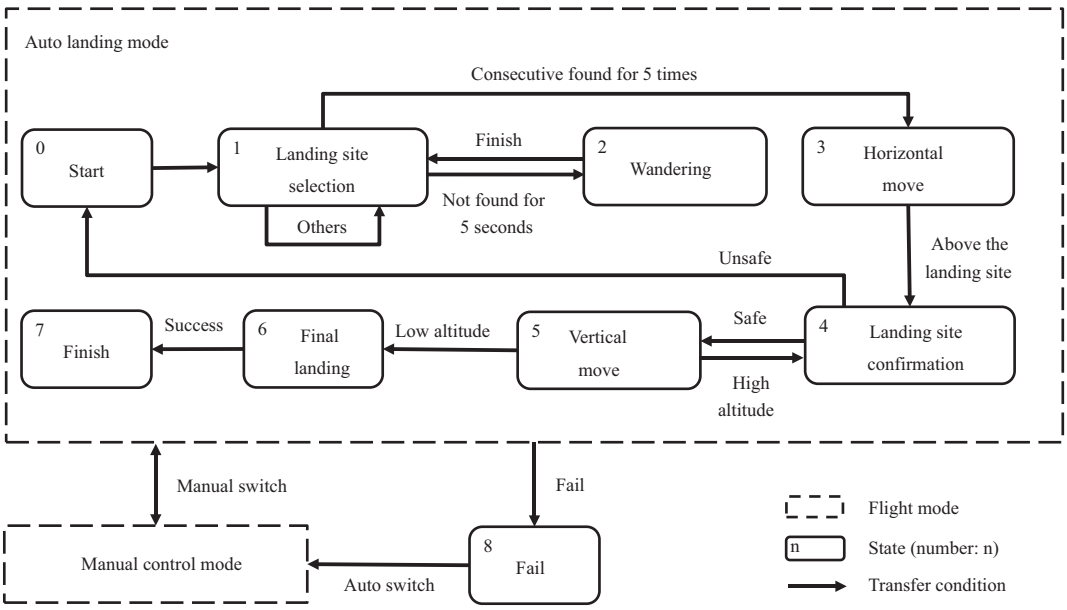

Figure 12 State transition of autonomous landing in the real flight test.

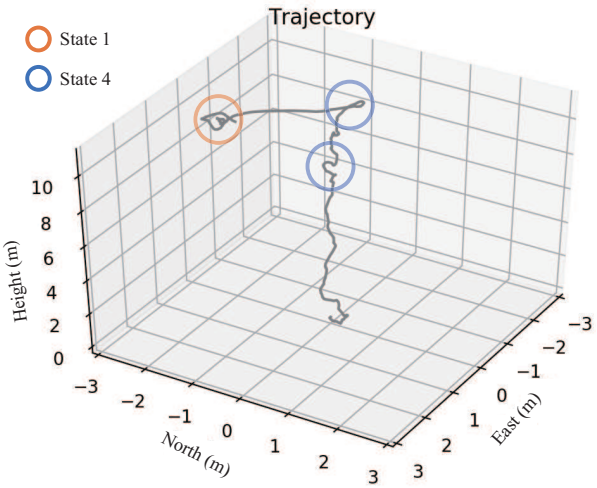

Figure 13 (Color online) 3D trajectory of the landing process in the local north-east-down (NED) coordinate.

in state 1, to eliminate the influence of the occasional errors, only the landing site that is confirmed by five consecutive frames will be chosen. On the contrary, if no landing site is confirmed for about 5 s, then it will shift to state 2 which performs a random wandering with a non-repetitive global path. It is also worth noting that for each frame we always choose the previous landing place as long as it is still safe and available to ensure continuous and stable control signals provided to the UAV. In state 4, the landing site is double checked by using the depth map directly generated from dense LiDAR data to identify extremely small obstacles, such as road curbs and inlets shown in Figure 11.

We conduct a series of autonomous landing experiments in a variety of environments. Under different circumstances, our UAV successfully finds the flat ground and land safely. We record the average LiDAR accumulation time to obtain an accurate depth map. In a familiar environment, it only takes 0.16 s, while in an unknown environment, the average time reaches 0.8 s. This demonstrates that our model can adaptively balance the accuracy and speed to ensure the reliability of the prediction results. A landing trajectory is drawn in Figure 13. It can be seen that the UAV performs a horizontal and vertical movement in order after a few seconds of hovering to select a safe landing site.

Finally, to evaluate the UAV performance in a completely unknown environment, we conduct an autonomous landing test in a night scene. Due to the dim light, the depth map and semantic segmentation

(a) (b)

Figure 14 (Color online) (a) Landing mask m_l and selected landing site; (b) landing process in the night condition.

map predicted by TerrainNet are very poor. However, by relying on onboard LiDAR, the UAV can still observe the small barriers in the FOV, as illustrated in Figure 14, thereby completing a safe landing.

6 Conclusion

In this paper, we studied the safe landing site selection of autonomous landing of UAVs in non-cooperative environments. Through the well-designed UAV platform and the robust terrain understanding model, the common problems of poor generalization ability and single terrain feature learning for existing methods are solved. Furthermore, we verified that our model can adaptively choose the LiDAR accumulation time to ensure accurate depth prediction. With the help of the predicted semantic segmentation result, our landing site selection method improved its accuracy in finding a safe landing place. Additionally, we applied our methods to real UAVs to achieve robust autonomous landing in both familiar and unknown environments. For future work, we plan to perform tracking and prediction of moving targets to further improve the overall safety level of the UAV landing.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61903216, 62073185).

References

1 Kong W W, Zhang D B, Wang X, et al. Autonomous landing of an UAV with a ground-based actuated infrared stereo vision system. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, 2013. 2963–2970
2 Kong W W, Zhou D L, Zhang Y, et al. A ground-based optical system for autonomous landing of a fixed wing UAV. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, 2014. 4797–4804
3 Sharp C S, Shakernia O, Sastry S S. A vision system for landing an unmanned aerial vehicle. In: Proceedings of IEEE International Conference on Robotics and Automation, Seoul, 2001. 2: 1720–1727
4 Saripalli S, Montgomery J F, Sukhatme G S. Visually guided landing of an unmanned aerial vehicle. IEEE Trans Robot Automat, 2003, 19: 371–380
5 Lange S, Sunderhauf N, Protzel P. A vision based onboard approach for landing and position control of an autonomous multirotor UAV in GPS-denied environments. In: Proceedings of International Conference on Advanced Robotics, Munich, 2009. 1–6
6 Benini A, Rutherford M J, Valavanis K P. Real-time, GPU-based pose estimation of a UAV for autonomous takeoff and landing. In: Proceedings of IEEE International Conference on Robotics and Automation, Stockholm, 2016. 3463–3470
7 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 IEEE International Conference on Robotics and Automation, St Paul, 2012. 971–976
8 Muskardin T, Balmer G, Wlach S, et al. Landing of a fixed-wing UAV on a mobile ground vehicle. In: Proceedings of IEEE International Conference on Robotics and Automation, Stockholm, 2016. 1237–1242
9 Wang H, Shi Z Y, Lu G, et al. Hierarchical fiducial marker design for pose estimation in large-scale scenarios. J Field Robotics, 2018, 35: 835–849
10 Saripalli S, Montgomery J F, Sukhatme G S. Vision-based autonomous landing of an unmanned aerial vehicle. In: Proceedings of IEEE International Conference on Robotics and Automation, Washington, 2002. 3: 2799–2804
11 Miller A, Shah M, Harper D. Landing a UAV on a runway using image registration. In: Proceedings of IEEE International Conference on Robotics and Automation, California, 2008. 182–187
12 Johnson A, Montgomery J, Matthies L. Vision guided landing of an autonomous helicopter in hazardous terrain. In: Proceedings of IEEE International Conference on Robotics and Automation, Barcelona, 2005. 3966–3971
13 Cherian A, Andersh J, Morellas V, et al. Autonomous altitude estimation of a UAV using a single onboard camera. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, 2009. 3900–3905

14 Marcu A, Costea D, Licaret V, et al. SafeUAV: learning to estimate depth and safe landing areas for UAVs from synthetic data. In: Proceedings of European Conference on Computer Vision Workshops, Munich, 2018
15 Eynard D, Vasseur P, Demonceaux C, et al. UAV altitude estimation by mixed stereoscopic vision. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, 2010. 646–651
16 Park J, Kim Y, Kim S. Landing site searching and selection algorithm development using vision system and its application to quadrotor. IEEE Trans Contr Syst Technol, 2014, 23: 488–503
17 Papa U, Del Core G. Design of sonar sensor model for safe landing of an UAV. In: Proceedings of IEEE Metrology for Aerospace, Benevento, 2015. 346–350
18 Bosch S, Lacroix S, Caballero F. Autonomous detection of safe landing areas for an UAV from monocular images. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, 2006. 5522–5527
19 Guo X F, Denman S, Fookes C, et al. Automatic UAV forced landing site detection using machine learning. In: Proceedings of International Conference on Digital Image Computing: Techniques and Applications (DICTA), Wollongong, 2014. 1–7
20 Guo X F, Denman S, Fookes C, et al. A robust UAV landing site detection system using mid-level discriminative patches. In: Proceedings of International Conference on Pattern Recognition, Cancun, 2016. 1659–1664
21 Hinzmann T, Stastny T, Cadena C, et al. Free LSD: prior-free visual landing site detection for autonomous planes. IEEE Robot Autom Lett, 2018, 3: 2545–2552
22 Patterson T, McClean S, Morrow P, et al. Utilizing geographic information system data for unmanned aerial vehicle position estimation. In: Proceedings of Canadian Conference on Computer and Robot Vision, Newfoundland, 2011. 8–15
23 Liu S H, Wang S Q, Shi W H, et al. Vehicle tracking by detection in UAV aerial video. Sci China Inf Sci, 2019, 62: 024101
24 Uhrig J, Schneider N, Schneider L, et al. Sparsity invariant CNNs. In: Proceedings of International Conference on 3D Vision, Qingdao, 2017. 11–20
25 Ma F C, Karaman S. Sparse-to-dense: depth prediction from sparse depth samples and a single image. In: Proceedings of IEEE International Conference on Robotics and Automation, Brisbane, 2018. 4796–4803
26 Chen Y, Yang B, Liang M, et al. Learning joint 2D-3D representations for depth completion. In: Proceedings of IEEE/CVF International Conference on Computer Vision, Seoul, 2019. 10023–10032
27 Qiu J, Cui Z, Zhang Y, et al. DeepLiDAR: deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image. In: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, 2019. 3313–3322
28 Poggi M, Pallotti D, Tosi F, et al. Guided stereo matching. In: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, 2019. 979–988
29 Tang J, Tian F P, Feng W, et al. Learning guided convolutional network for depth completion. IEEE Trans Image Process, 2020, 30: 1116–1129
30 Maffra F, Teixeira L, Chen Z T, et al. Real-time wide-baseline place recognition using depth completion. IEEE Robot Autom Lett, 2019, 4: 1525–1532
31 Teixeira L, Oswald M R, Pollefeys M, et al. Aerial single-view depth completion with image-guided uncertainty estimation. IEEE Robot Autom Lett, 2020, 5: 1055–1062
32 Xu J, Schwing A G, Urtasun R. Learning to segment under various forms of weak supervision. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Boston, 2015. 3781–3790
33 Kolesnikov A, Lampert C H. Seed, expand and constrain: three principles for weakly-supervised image segmentation. In: Proceedings of European Conference on Computer Vision, Amsterdam, 2016. 695–711
34 Feng J P, Wang X G, Liu W Y. Deep graph cut network for weakly-supervised semantic segmentation. Sci China Inf Sci, 2021, 64: 130105
35 Bearman A, Russakovsky O, Ferrari V, et al. What's the point: semantic segmentation with point supervision. In: Proceedings of European Conference on Computer Vision, Amsterdam, 2016. 549–565
36 Lin D, Dai J F, Jia J Y, et al. ScribbleSup: scribble-supervised convolutional networks for semantic segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, 2016. 3159–3167
37 Tang M, Djelouah A, Perazzi F, et al. Normalized cut loss for weakly-supervised CNN segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, 2018. 1818–1827
38 Dai J F, He K M, Sun J. BoxSup: exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In: Proceedings of IEEE/CVF International Conference on Computer Vision, Santiago, 2015. 1635–1643
39 Khoreva A, Benenson R, Hosang J, et al. Simple does it: weakly supervised instance and semantic segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, 2017. 876–885
40 Vernaza P, Chandraker M. Learning random-walk label propagation for weakly-supervised semantic segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, 2017. 7158–7166
41 Tang M, Perazzi F, Djelouah A, et al. On regularized losses for weakly-supervised CNN segmentation. In: Proceedings of European Conference on Computer Vision, Munich, 2018. 507–522
42 Zhang Z. A flexible new technique for camera calibration. IEEE Trans Pattern Anal Machine Intell, 2000, 22: 1330–1334
43 Fischler M A, Bolles R C. Random sample consensus. Commun ACM, 1981, 24: 381–395
44 Chen L C, Zhu Y K, Papandreou G, et al. Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of European Conference on Computer Vision, Munich, 2018. 801–818
45 Godard C, Aodha O M, Brostow G J. Unsupervised monocular depth estimation with left-right consistency. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, 2017. 270–279
46 Wang Z, Bovik A C, Sheikh H R, et al. Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process, 2004, 13: 600–612
47 Eigen D, Puhrsch C, Fergus R. Depth map prediction from a single image using a multi-scale deep network. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, 2014. 27: 2366–2374
48 Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Boston, 2015. 3431–3440
49 Kingma D P, Ba J. Adam: a method for stochastic optimization. In: Proceedings of International Conference on Learning Representations, San Diego, 2015
50 He K M, Zhang X Y, Ren S Q, et al. Deep residual learning for image recognition. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, 2016. 770–778
51 Ma F C, Cavalheiro G V, Karaman S. Self-supervised sparse-to-dense: self-supervised depth completion from lidar and monocular camera. In: Proceedings of International Conference on Robotics and Automation, Montreal, 2019. 3288–3295