Landing Site Detection for UAVs Based on CNNs Classification and Optical Flow from Monocular Camera Images

Kikumoto, Kikumoto
Harimoto, Yoh
Isogaya, Kazuki
Yoshida, Takeshi
Urakubo, Takateru

(Citation)
Journal of Robotics and Mechatronics,33(2):292-300

(Issue Date)
2021-04-20

(Resource Type)
journal article

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/0100477313

https://doi.org/10.20965/jrm.2021.p0292

Paper:

Landing Site Detection for UAVs Based on CNNs Classification and Optical Flow from Monocular Camera Images

Chihiro Kikumoto*, Yoh Harimoto*, Kazuki Isogaya*, Takeshi Yoshida, and Takateru Urakubo***

*Kobe University
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan
E-mail: {kikumoto, harimoto, isogaya}@al.cs.kobe-u.ac.jp, t.urakubo@silver.kobe-u.ac.jp
**Ritsumeikan University
1-1-1 Noji-higashi, Kusatsu, Shiga 525-8577, Japan
E-mail: yoshi-da@fc.ritsumei.ac.jp
[Received October 14, 2020; accepted February 2, 2021]

The increased use of UAVs (Unmanned Aerial Vehicles) has heightened demands for an automated landing system intended for a variety of tasks and emergency landings. A key challenge of this system is finding a safe landing site in an unknown environment using on-board sensors. This paper proposes a method to generate a heat map for safety evaluation using images from a single on-board camera. The proposed method consists of the classification of ground surface by CNNs (Convolutional Neural Networks) and the estimation of surface flatness from optical flow. We present the results of applying this method to a video obtained from an on-board camera and discuss ways of improving the method.

Keywords: unmanned aerial vehicle, autonomous landing, land cover classification, topographic mapping

1. Introduction

In the event of a large-scale disaster, such as a high-magnitude earthquake, it is important to collect information on damages and deliver relief supplies quickly. UAVs are a promising means for accomplishing these tasks because they can be used even when roads become disconnected due to landslides [1]. However, it is necessary to develop an automated landing system [2, 3] because the delivery tasks would be performed in an unknown environment in the event of a large-scale disaster. Automatic detection of a safe landing site with on-board sensors is the key technology for achieving successful automatic landing in an unknown environment. Additionally, automatic landing site detection is also useful for emergency landing in case of an aircraft malfunction [4, 5]. Establishing a method for such automatic detection is important to enable the construction of next-generation transportation systems with UAVs.

Many researchers have been tackling the problem of automatic landing site detection using two approaches: land cover classification and land surface topography. In the first approach, the types of ground surface, such as grass, trees, and roads are classified using on-board sensors. In [6], camera images were used to classify the surface into three classes by the Support Vector Machine (SVM). In the second approach, the shape of ground surface was estimated by on-board sensors, such as LiDAR and cameras [7–10]. The elevation map or 3D map of the surface is built to detect a flat area without slopes as a safe landing site. In [11], both approaches were used simultaneously for more reliable detection. Since they targeted autonomous landing of a fixed-wing UAV, grass was the only surface class to be classified for impact mitigation purposes. However, for multi-rotor and VTOL UAVs, it would be better to classify the surface into multiple classes because they land vertically without largely impacting any flat and solid surface, such as roads and bare lands as well as grass. To the best of our knowledge, no previous study have approached this problem in terms of multiclass classification and topography.

This paper proposes a safe landing site detection method that combines the two aforementioned approaches, namely surface classification and topographic recognition, using images from a single on-board camera on a UAV. In the proposed method, a camera image is divided into small image tiles, and two safety scores for each small tile are obtained respectively from CNNs that classify the ground surface into nine classes and OF (Optical Flow) that is used to estimate the degree of unevenness of the ground surface. A heat map for safety evaluation of image tiles is calculated by integrating these two scores. This method requires no prior knowledge about the environment like the method used in [11]. We apply the method to actual on-board camera images to show that successful detection of safe landing sites can be achieved using this method, and discuss ways of enhancing the method.

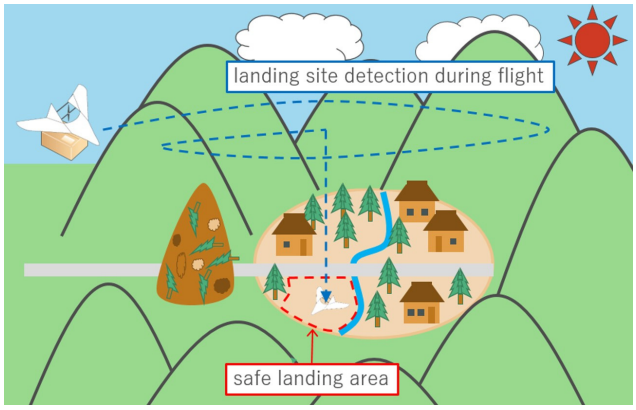


Fig. 1. Approach and landing to a disaster-stricken area.

2. Safe Landing Site Detection

2.1. Scenario for Autonomous Landing

When a disaster strikes, all land routes to the affected area may become inaccessible due to landslides, heavy traffic, or other reasons. Even in such cases, UAVs can approach and land in the target area to transport relief supplies quickly as shown in **Fig. 1**. The following four flight procedures for landing in a disaster-stricken area without any prior knowledge about its environment are considered.

1) Flying to the target area using GPS location information.
2) Determining a potential landing site with an on-board camera while flying over the target area.
3) Approaching above the potential landing site.
4) Landing on the site while reconfirming the safety of the landing point using camera and other equipment.

This paper focuses on procedure 2) and proposes a method of detecting a safe landing site during a level flight over the target area. The altitude in this procedure is assumed to be about 100 m so that no flight maneuver is required to avoid collision with trees, utility poles, buildings, and so on.

Further, the potential landing sites should satisfy the following three requirements for safe landing of UAVs.

(1) It should not be a water-filled place, such as a river or pond.
(2) It should not be a place that is difficult for people to reach, such as the roof of a building.
(3) It should not be a non-flat place, such as a steep slope or uneven surface.

Requirement (1) is necessary to avoid catastrophic damage to UAVs, and requirement (2) is needed for people to pick up the relief supplies that are delivered by UAVs. Requirement (3) is also necessary because the risk of landing failure is high in case of a non-flat surface.

To detect a site that satisfies the three requirements, we need to know both the type and shape of the ground surface. Moreover, an additional requirement should be considered for procedure 4) to avoid collision with moving objects, such as people and vehicles. We will discuss this requirement in Section 7.

2.2. Overview of the Proposed Method Using Camera Images

We propose a method of automatic landing site detection for procedure 2) by integrating classification of the ground surface and estimation of the ground flatness. **Fig. 2** shows the processing flow of the proposed method: the input image (a) taken from an on-board camera is divided into image tiles whose size is 128×128 pixels before the two processes classification by CNNs (b) and flatness estimation by OF (c). The size of image tile is chosen so that each tile corresponds to a 5×5 m area of the ground surface when a UAV flies at an altitude of 100 m. It should be noted that the size is large enough for small multi-rotor and VTOL UAVs to land there.

In the classification (b), each image tile is classified into nine classes by CNNs, and a safety score is given to it based on the classification results. In the flatness estimation (c), the image tiles that can be considered flat based on the OF between consecutive two images are detected, and another safety score is given based on the flatness estimation results.

From (b) and (c), two kinds of safety scores in the form of heat maps (d) and (e) are obtained and integrated to draw a heat map (f) for safety evaluation. In the heat map (f), the safety score for each image tile is calculated by averaging the two scores in (d) and (e). If the score of an image tile is higher than the threshold, the corresponding area is detected as a safe landing site in the process (g), and three image tiles with the highest scores are displayed as three promising landing sites in the process (h).

3. Land Cover Classification by CNNs

If a landing site is detected only by evaluating the ground surface flatness, it is difficult to avoid a place unsuitable for landing, such as a water surface and an inaccessible roof of a building. To distinguish between suitable and unsuitable landing zones, it is necessary to know the types of surface from on-board camera images.

In this paper, the types of ground surface were classified by CNNs. We used the Xception model that was proposed recently in [12]. It shows better classification performance compared to other network structures of CNNs. To obtain nine class outputs, two fully-connected layers were added to the Xception model that is denoted as base model layer in **Fig. 3**. The input layer was also modified so that 128×128 pixel input images could be utilized. We initialized the Xception model using the weights pretrained for ImageNet that are available in Keras, and we fine-tuned its weights during training.

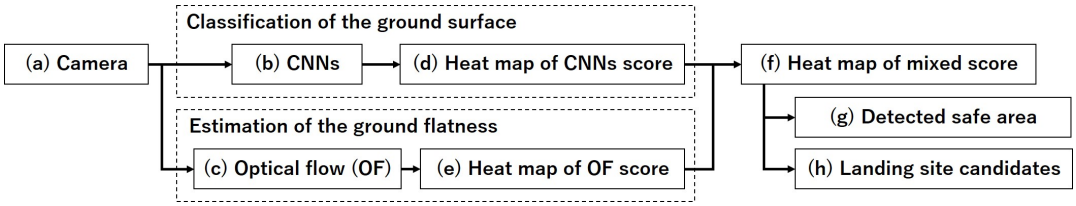


Fig. 2. Proposed framework for landing site detection.

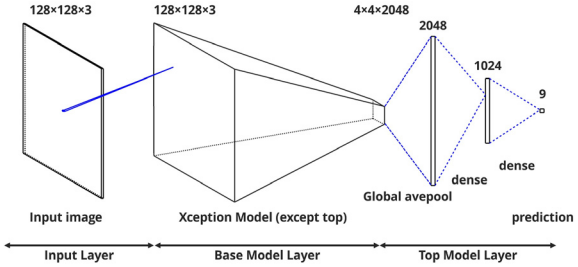


Fig. 3. Networks architecture for land surface classification.

Table 1. Learning parameters.

Image size	128×128
Input channel	3
Solver	Adam
Base learning rate	1e-3
Batch size	32
Regularizer	L2(1e-3)
Activation	LeakyReLU
Dropout	0.5

Fig. 4. Examples of training data set: grass, bare land, trees, water, wilderness, road, buildings, cars, and ships, in order from the left column.

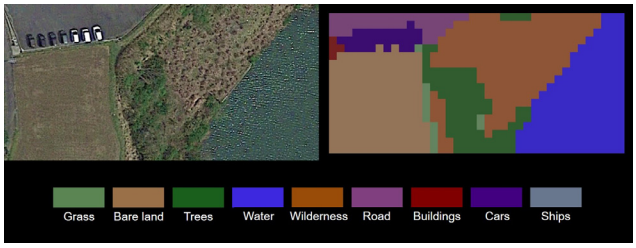


Fig. 5. Classification results by trained CNNs (color figure online).

Surface image data for training and evaluating CNNs were generated based on satellite images from Google Maps. We created 128×128 pixel images of the ground surface as shown in **Fig. 4**. We classified them into nine classes, namely grass, bare land, trees, water, wilderness, road, buildings, cars, and ships. A total of 36088 training and evaluating data were created through rotation and flipping processes. During the training, we also used dynamic data extensions, such as shifts, brightness and saturation conversions, and Random Erasing [13]. **Table 1** shows settings of the training parameters. Training was performed on 95% of the data. The prediction accuracy for the remaining 5% of the data was 98.5%. **Fig. 5** depicts an example of the prediction result obtained by the CNNs for a satellite image from Google Maps where the image is divided into 128×128 pixel image tiles, and each image tile is classified to one of the nine classes. The classification result seems accurate for most of the image tiles.

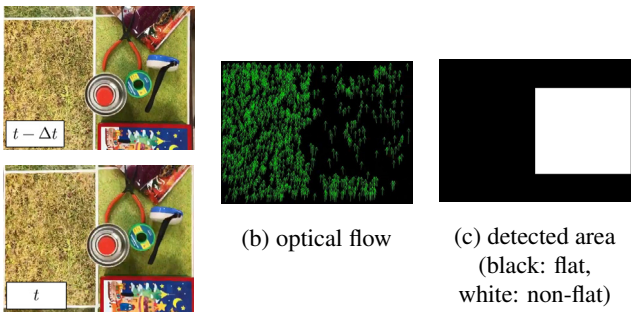
4. Detection of Flat Surface by Optical Flow

To satisfy requirement (3), we need to find areas with no slope or unevenness. Computationally demanding 3D reconstruction is unnecessary for detecting such a flat ground area. In [14], a less computationally expensive method for detecting flat areas was proposed using the standard deviation of the norms of optical flow vectors. We utilize and modify the method so that the detection of flat areas is robust to flight speed changes by using the coefficient of variation instead of the standard deviation.

Since the UAV in procedure 2) flies horizontally with its on-board camera pointing downward, the optical flow vector for an object that is located at high altitude is larger than the one for an object at low altitude. Therefore, when the ground surface is flat, the standard deviation is small, and it is large when the surface is uneven. The results of preliminary experiments showed that the standard deviation of the optical flow vectors depends on the flight speed of UAV even when the flight altitude is kept at a constant. By using the coefficient of variation, robust detection of flat areas can be achieved.

The optical flow is obtained from consecutive image frames. SIFT features are calculated in the current and previous frames and matched to obtain feature point pairs $\boldsymbol{P}_{\text{current}}(i)$ and $\boldsymbol{P}_{\text{previous}}(i)$. The magnitude of the optical flow $F(i)$ is computed as follows:

$$F(i) = \|\boldsymbol{F}(i)\| = \|\boldsymbol{P}_{\text{current}}(i) - \boldsymbol{P}_{\text{previous}}(i)\|. \quad . \quad (1)$$



(a) video images (b) optical flow (c) detected area (black: flat, white: non-flat)

Fig. 6. Results of flat area detection.

Table 2. Safety score.

Class	Score
Grass	1.0
Bare land	1.0
Road	0.8
Wilderness	−0.5
Cars	−0.5
Ships	−0.8
Trees	−0.8
Buildings	−1.0
Water	−1.0

We calculate the coefficient of variation C_V for a rectangular image tile. We denote the location of an image tile in the current image frame as (x,y), and assume that there are n optical flow vectors in the image tile (x,y). The coefficient of variation for the image tile (x,y) is obtained by the following equation:

$$C_V(x,y) = \frac{\sigma_F(x,y)}{\bar{F}(x,y)}, \quad \ldots\ldots\ldots\ldots \quad (2)$$

where $\bar{F}(x,y)$ and $\sigma_F(x,y)$ are respectively the average and standard deviation of $F(i)$ $(i = 1,\ldots,n)$.

This section demonstrates the detection of flat tiles with OF by introducing a threshold T for C_V. The area of tile (x,y) is judged to be flat if $C_V(x,y) < T$. However, C_V itself will be used in the next section to generate a heat map for safety evaluation.

Figure 6 shows the results of experiments with an artificially constructed environment. In the environment, the left half area of the images (a) is flat, and a textured sheet is put on the area to obtain optical flow vectors. In the right half area, there are some objects on a flat surface: a box whose top surface is flat in the lower area, some objects whose heights are small in the upper area, and some tall objects in the center area. From the two images, optical flow vectors in **Fig. 6(b)** are obtained. In **Fig. 6(c)**, the white and black areas are considered non-flat and flat, respectively. By using the coefficient of variation, flat areas are extracted correctly.

Although this method generally assumes a horizontal motion of the camera, the altitude and posture may vary in actual flight. The influence of altitude fluctuation at an altitude of about 100 m would be small because the altitude control of UAVs can limit the fluctuation to about 5 m. For suppressing the influence of posture variation, a gimbal mechanism can be attached to the camera, or the image can be corrected by using other sensors, such as IMU.

5. Integrated Heat Map

This section integrates the surface classification in Section 3 and the flatness estimation in Section 4 to generate a heat map that represents safety evaluation for 128×128 pixel image tiles. First, two heat maps that are respectively based on the classification results and the flatness estimation results were made by assigning safety scores to each image tile. The scores for each tile were calculated by merging properly the results of its surrounding tiles in order to detect safe landing sites more reliably. Subsequently, an integrated heat map was obtained by averaging the two scores.

5.1. Heat Map of CNNs Score

To generate a heat map from the results of the surface classification, we define the safety score for each class as shown in **Table 2**. The score assumes values from -1.0 to 1.0. A higher value indicates that the corresponding class is more suitable for landing. The grass and bare land are considered to be the safest for landing. The score of road is set at 0.8 because there tend to be power lines that are difficult to detect from an altitude of 100 m. Although wilderness and cars are unsuitable for landing, we could reach them to pick up relief supplies. Therefore, we choose their scores as -0.5. The scores of trees and ships are set at -0.8 since it is more difficult to reach them. The water and buildings is set to the lowest value of -1.0 for two reasons. First, they are really unsuitable for landing. Second, when an image tile belongs to unlearned types of the ground surface, the classifier in Section 3 tends to classify it as water or buildings.

The score of a target 128×128 pixel image tile is calculated by combining the results of classification for the surrounding image regions as shown below. At first, we create eight 128×128 pixel image regions by sliding a target image tile by 64 pixels as shown in **Fig. 7** and apply the classifier to them. We denote the target image tile as i, four areas surrounded by a dashed line as b_i, and four areas surrounded by a dotted line as r_i. The probability vectors $\boldsymbol{p}$ that are composed of the probabilities for the nine classes are obtained for the image tile and the eight image regions. These nine probability vectors are added together with weights to obtain a modified probability vector $\boldsymbol{p}_m$ for the target image tile:

$$\boldsymbol{p}_m(i) = 4\boldsymbol{p}(i) + 2\sum_{b_i=1}^{4} \boldsymbol{p}(b_i) + \sum_{r_i=1}^{4} \boldsymbol{p}(r_i), \quad \ldots \quad (3)$$

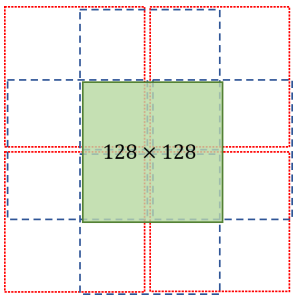


Fig. 7. Surrounding image regions for score calculation by CNNs.

where the weights are chosen as 4 for the target image tile, 2 for four areas surrounded by a dashed line, and 1 for four areas surrounded by a dotted line, according to the degree of overlap with the target image tile. All the components of the modified probability vector are multiplied by the corresponding scores in **Table 2**, and they are added together to obtain a temporal score $\bar{S}_C$ of the target image tile. Repeating this process for every image tile, we obtain temporal scores for all 128×128 pixel image tiles.

We then consider eight image tiles that surround a target image tile and denote them as s_i. An image tile would be safer for landing if its surrounding tiles also have high scores. The final score S_C for the target image tile is obtained by multiplying the temporal scores of the surrounding tiles by 0.1, summing them up, and adding that to the temporal score of the target tile:

$$S_C(i) = \bar{S}_C(i) + 0.1 \sum_{s_i=1}^{8} \bar{S}_C(s_i). \quad \ldots\ldots\ldots \quad (4)$$

We use the obtained score $S_C(i)$ for each image tile i to draw a heat map of CNNs score (**Fig. 8(d)**). The final score takes a value between -28.8 and 28.8 through the processes mentioned above.

5.2. Heat Map of OF Score

To obtain a safety score based on the flatness estimation for a 128×128 pixel image tile, we first consider four 256×256 pixel regions and denote them as l_i. The four regions include the target image tile i in the lower right, upper right, lower left, or upper left area as shown in **Fig. 9**. The coefficients of variation C_V for optical flow vectors are calculated for these four regions, and we obtain a temporal score for the target image tile by adding them together:

$$\bar{S}_O(i) = \sum_{l_i=1}^{4} C_V(l_i). \quad \ldots\ldots\ldots\ldots \quad (5)$$

The reason for considering the larger image regions is because increasing the image size is desired so that the image contains enough number of optical flow vectors to calculate the coefficient of variation C_V.

To integrate the score from optical flow with the score from CNNs classification properly, the range of OF score needs to be the same as that of the CNNs score. The results of preliminary experiments showed that the value of $C_V(l_i)$ is approximately between 0 and 0.05, that is, the range of $\bar{S}_O(i)$ is within 0 to 0.2. The final score S_O from OF is defined as:

$$S_O(i) = -288\left(\bar{S}_O(i) - 0.1\right), \quad \ldots\ldots\ldots \quad (6)$$

so that the range of the final score is from -28.8 to 28.8. We use the obtained score $S_O(i)$ for each image tile i to draw a heat map of OF score (**Fig. 8(e)**). Moreover, when the number of optical flow vectors in a 256×256 pixel region l_i is less than 20, $C_V(l_i)$ is set at 0.025. If $C_V(l_i)$ becomes larger than 0.05, it is set at 0.05.

5.3. Heat Map of Mixed Score

A safety score S for each 128×128 pixel image tile i is defined in this paper as the average of the two scores S_C and S_O:

$$S(i) = \frac{1}{2}(S_C(i) + S_O(i)). \quad \ldots\ldots\ldots\ldots \quad (7)$$

The safety score S is utilized to draw an integrated heat map as shown in **Fig. 8(f)**. In this paper, an image tile i is considered a safe landing site if $S(i) \geq 14.4$ (**Fig. 8(g)**). Furthermore, the top three highest-scoring image tiles are displayed as the promising landing sites, as shown in **Fig. 8(h)**.

The results of safe landing area detection depend on the threshold of the judgment as well as the definition of the score S. The threshold value was chosen in this paper as 14.4 such that the proposed algorithm works as follows: when an image tile and its surrounding regions are grass or bare land, the tile is judged as safe even for the OF score of 0. On the other hand, when an image tile is classified as wilderness, cars, etc. and has a negative score of CNNs, it is judged to be unsafe for any score obtained from the OF. As described in the next section, this choice of threshold results in successful detection of safe landing area for the images obtained in experiments.

However, to confirm the safety according to the UAV's size and landing gears, the relationship between the OF score and the actual height differences on uneven surfaces should be examined carefully. Further investigation with more images collected for various environments may lead to more appropriate choice of the score S and its threshold.

6. Experimental Results

In this section, in order to examine whether safe landing sites can be detected using the proposed method, we apply it to actual on-board camera images that were taken by a commercially available multi-rotor UAV, Mavic Pro Platinum (DJI). The on-board camera is mounted on the UAV with a gimbal mechanism. It can keep pointing downward during flight. To obtain consecutive images, a video was taken by the on-board camera when the UAV flew over a park at an altitude of about 100 m.

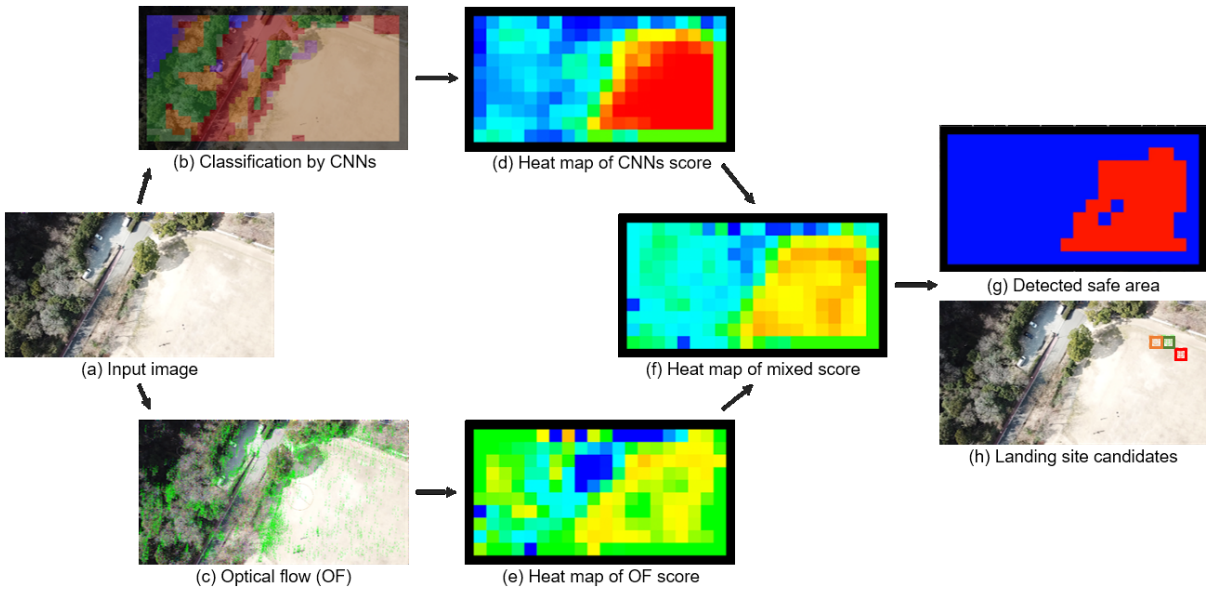


Fig. 8. Results of landing site detection (color figure online).

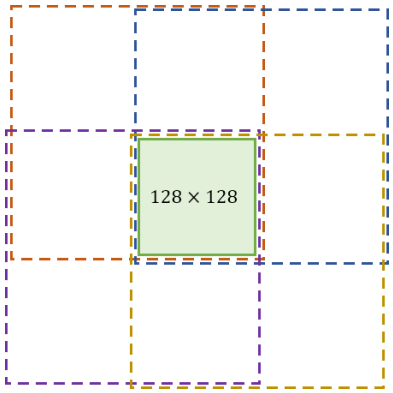
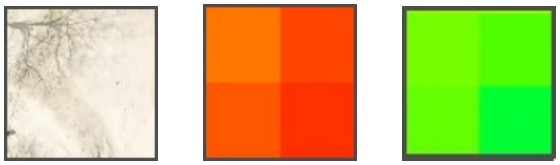
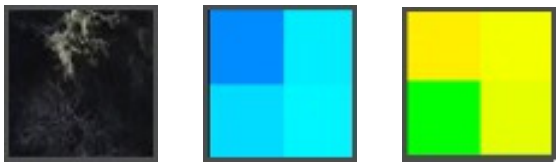


Fig. 9. Surrounding image regions for score calculation by OF.



(a) cropped image (b) heat map by CNNs score (c) heat map by optical flow

Fig. 10. An example in which integration with optical flow is effective.



(a) cropped image (b) heat map by CNNs score (c) heat map by optical flow

Fig. 11. An example in which integration with CNNs is effective.

Figure 8 shows the results of landing site detection by the proposed method for one image frame during flight. There is a large region of bare land in the input image (a), that is suitable for landing. The region was classified correctly by the CNNs classifier as shown in (b). The corresponding image tiles exhibited high scores in the heat map (d). As shown in (c) and (e), most image tiles in the region were also detected as flat surfaces by using OF, although the scores for some image tiles were low even in the region due to few optical flow vectors. From the integrated heat map (f), effective detection of safe landing area was achieved as shown in (g), and three promising landing sites were displayed in (h). Moreover, for every frame of the video, safe landing sites were detected successfully in a similar way.

The rest of this section shows two examples where the combination of land cover classification and flatness estimation works effectively for reliable detection of safe landing sites. **Fig. 10** shows a small sand hill whose height is about 50 cm in the lower left part of the cropped image (a). The image area was correctly classified as bare land by the CNNs classifier, and the score from the CNNs was high as in (b), although the area was unsuitable for landing. On the other hand, the OF score for the area was sufficiently low due to its unevenness as shown in (c). Consequently, the area was not considered safe for landing from the integrated heat map.

In **Fig. 11**, only the top of trees is illuminated by the sunlight in (a). The score from the optical flow was unexpectedly high as in (c) because almost all the feature points that were used for optical flow calculation were located at about the same height. On the other hand, CNNs classifier correctly classified the brighter image tiles as

trees and other image tiles incorrectly as water because they are too dark due to shadows. Even so, the CNNs score was sufficiently low in the heat map (b) because we assigned low safety scores to those two classes as shown in **Table 2**. Thus, the area was not detected as a safe landing site based on the integrated heat map.

7. Discussion

As shown in the previous section, the proposed method enables us to find an area suitable for landing. However, to achieve autonomous landing, there are many issues that should be addressed, such as implementation of the method to actual UAVs, enhancement of detection accuracy, and landing site tracking during descent flight. This section discusses these issues from the following four points of view.

(1) Computational speed

The algorithm for landing site detection in **Fig. 2** was calculated offline from the video that was taken during the experiments mentioned in Section 6. It took about 26 s to finish the calculation from two consecutive images by using a desktop PC (CPU: Intel Core i7-6700 (8 CPUs), GPU: NVIDIA GeForce GTX970, Memory: 16 GB 2133 MHz). The flat area detection by OF took about 23 s, while the classification by CNNs took about 3 s. This is probably because we performed brute-force matching with SIFT descriptors to obtain OF vectors. We could reduce the computational time to a large extent by using other OF algorithms [15–18]. For example, we have already checked through preliminary calculation that sufficiently accurate and dense OF vectors can be obtained within about 1.5 s using the Farneback algorithm [15].

It should be noted that we do not have to obtain the detection results within the frame rate of the video. The required computational speed depends on the total strategy for landing, such as flight plan and final decision of a landing point [2, 14]. If the whole computation for detecting a landing point is finished within about 4.5 s by using a more efficient OF algorithm, the UAV can keep flying until the detection results are obtained, preparing for final approach to the target area. In the future, we will check the feasibility of implementation of the method by using a compact, lightweight, high-speed embedded computer that can be mounted on small UAVs, such as Jetson Xavier NX (NVIDIA).

Moreover, an additional way of reducing the amount of computation is reducing the image area processed for landing site detection [11]. If we process land cover classification and flatness estimation sequentially, the area that is judged to be definitely unsuitable for landing by one of the two does not need to be processed by the other. The image area to be processed can be also reduced by using the results of the last few frames, if they overlap.

(2) Mapping and accumulation of detection results

In the proposed method, the landing sites are detected independently in each frame. It is possible to improve the efficiency of processing and the accuracy of detection by accumulating the detection results, if the images in multiple frames overlap. To accumulate the results and use them for landing control, we need to map them to the corresponding position on the ground. As described later, it would be better to construct a 3D map [19] or an elevation map using SLAM and other methods [20].

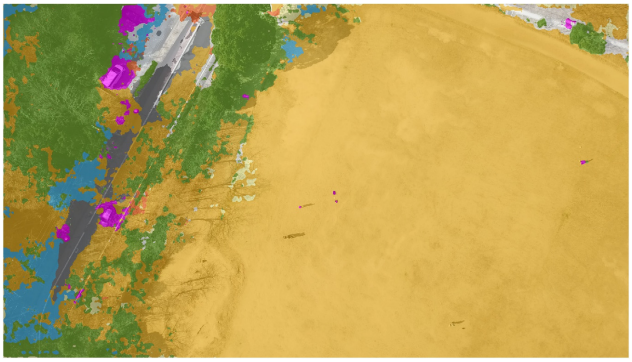


Fig. 12. Result of semantic segmentation (color figure online).

(3) Fine detection of safe landing area

The landing site detection by the proposed method is performed only for each 128×128 pixel image tile that corresponds to 5×5 m ground area. The detection for smaller areas would enable us to know more accurate shape of the region suitable for landing, and would allow the UAVs to land in narrow areas.

We are currently developing a classifier for semantic segmentation [21, 22] to classify the ground surface at the pixel level. A preliminary result for the same input image as in Section 6 is shown in **Fig. 12**. For fine terrain recognition, a 3D map would be useful, if constructed.

(4) Landing site tracking during descent flight

In procedure 4) described in Section 2.1, a multi-copter or VTOL type of UAV approaches one of the promising landing sites while descending. It is desirable to construct a 3D map using SLAM [23] because it is useful for detecting flat areas even during descent flight and enables a more detailed understanding of the ground shape.

An important requirement for safe landing in procedure 4) is detecting and avoiding moving objects, such as people and vehicles in addition to the requirements (1)–(3) mentioned in Section 2.1. We are currently working on human detection from aerial images using Single Shot Multi-box Detection (SSD). As shown in **Fig. 13**, preliminary results show that a person can be recognized from an image taken at an altitude of about 50 m. The human detection would also be useful for finding people in need of help in a disaster-stricken area [24].

8. Conclusion

In this paper, we proposed a method for detecting safe landing sites with a monocular camera by combining

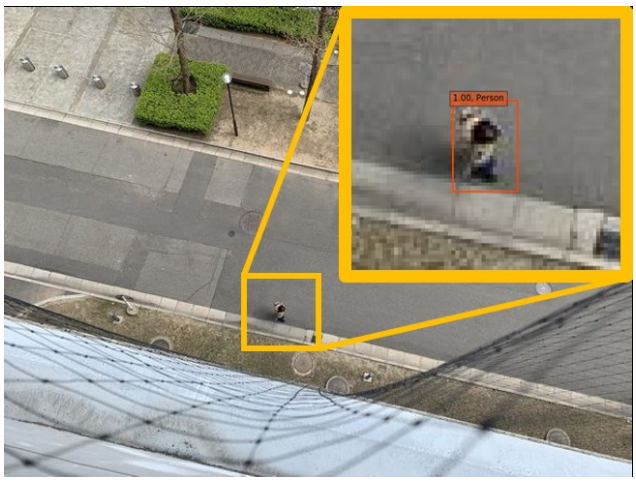


Fig. 13. Human detection from an overhead shot image with SSD.

ground surface classification and flatness estimation. Dividing a camera image into 128×128 pixel image tiles, each image tile was classified into nine classes by CNNs, and its degree of flatness was estimated based on optical flow vectors. These results were integrated as a heat map of safety score. The area with a score higher than a threshold is considered suitable for landing. The experimental results demonstrated that the proposed method can achieve a successful detection from actual aerial images. Finally, we discussed some issues that should be solved to enable its practical application.

Acknowledgements

This work was partly supported by the Futaba Foundation.

References:

[1] K. Nonami, "Drone Technology, Cutting-Edge Drone Business, and Future Prospects," J. Robot. Mechatron., Vol.28, No.3, pp. 262-272, 2016.

[2] S. Thurrowgood et al., "A Biologically Inspired, Vision-based Guidance System for Automatic Landing of a Fixed-wing Aircraft," J. Field Robot., Vol.31, No.4, pp. 699-727, doi: 1002/rob.21527, 2014.

[3] M. Laiacker et al., "Vision aided automatic landing system for fixed wing UAV," Proc. of 2013 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 2971-2976, doi: 10.1109/IROS.2013.6696777, 2013.

[4] M. Warren et al., "Enabling aircraft emergency landings using active visual site detection," Proc. of the 9th Conf. on Field and Service Robotics, pp. 1-14, doi: 10.1007/978-3-319-07488-7_12, 2013.

[5] X. Guo et al., "Automatic UAV Forced Landing Site Detection using Machine Learning," Proc. of 2014 Int. Conf. on Digital Image Computing: Techniques and Applications, pp. 140-145, doi: 10.1109/DICTA.2014.7008097, 2014.

[6] X. Guo et al., "A robust UAV landing site detection system using mid-level discriminative patches," Proc. of the 23rd Int. Conf. on Pattern Recognition, pp. 1659-1664, doi: 10.1109/ICPR.2016.7899875, 2016.

[7] M. Humenberger et al., "Vision-based Landing Site Evaluation and Trajectory Generation Toward Rooftop Landing," Robotics: Science and Systems X, doi: 10.15607/RSS.2014.X.044, 2014.

[8] S. Bosch, S. Lacroix, and F. Caballero, "Autonomous Detection of Safe Landing Areas for an UAV from Monocular Images," Proc. of 2006 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 5522-5527, doi: 10.1109/IROS.2006.282188, 2006.

[9] C. Forster et al., "Continuous On-Board Monocular-Vision-based Elevation Mapping Applied to Autonomous Landing of Micro Aerial Vehicles," Proc. of 2015 IEEE Int. Conf. on Robotics and Automation, pp. 111-118, doi: 10.1109/ICRA.2015.7138988, 2015.

[10] L. Yan et al., "A Safe Landing Site Selection Method of UAVs Based on LiDAR Point Clouds," Proc. of 2020 39th Chinese Control Conf., pp. 6497-6502, doi: 10.23919/CCC50068.2020.9189499, 2020.

[11] T. Hinzmann et al., "Free LSD: Prior-Free Visual Landing Site Detection for Autonomous Planes," IEEE Robotics and Automation Letters, Vol.3, Issue 3, pp. 2545-2552, 2018.

[12] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," Proc. of 2017 IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1800-1807, 2017.

[13] Z. Zhong et al., "Random Erasing Data Augmentation," Proc. of the AAAI Conf. on Artificial Intelligence, Vol.34, No.07, pp. 13001-13008, doi: 10.1609/aaai.v34i07.7000, 2020.

[14] A. Cesetti et al., "A Vision-Based Guidance System for UAV Navigation and Safe Landing using Natural Landmarks," J. Intell. Robot. Syst., Vol.57, pp. 233-257, doi: 10.1007/s10846-009-9373-3, 2010.

[15] G. Farnebäck, "Two-Frame Motion Estimation Based on Polynomial Expansion," Proc. of Scandinavian Conf. on Image Analysis, pp. 363-370, doi: 10.1007/3-540-45103-X_50, 2003.

[16] E. Ilg et al., "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks," Proc. of 2017 IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1647-1655, doi: 10.1109/CVPR.2017.179, 2017.

[17] J. D. Adarve and R. Mahony, "A Filter Formulation for Computing Real Time Optical Flow," IEEE Robotics and Automation Letters, Vol.1, No.2, pp. 1192-1199, doi: 10.1109/LRA.2016.2532928, 2016.

[18] N. J. Sanket et al., "GapFlyt: Active Vision Based Minimalist Structure-Less Gap Detection for Quadrotor Flight," IEEE Robotics and Automation Letters, Vol.3, No.4, pp. 2799-2806, doi: 10.1109/LRA.2018.2843445, 2018.

[19] A. Hornung et al., "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," Auton Robot, Vol.34, pp. 189-206, doi: 10.1007/s10514-012-9321-0, 2013.

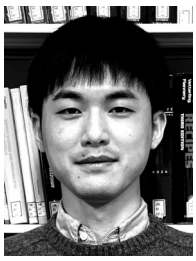
[20] T. Hinzmann et al., "Robust Map Generation for Fixed-Wing UAVs with Low-Cost Highly-Oblique Monocular Cameras," Proc. of 2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 3261-3268, doi: 10.1109/IROS.2016.7759503, 2016.

[21] M. Ghiasi and R. Amirfattahi, "Fast semantic segmentation of aerial images based on color and texture," Proc. of 2013 8th Iranian Conf. on Machine Vision and Image Processing, pp. 324-327, doi: 10.1109/IranianMVIP.2013.6780004, 2013.

[22] R. Miyamoto et al., "Visual Navigation Based on Semantic Segmentation Using Only a Monocular Camera as an External Sensor," J. Robot. Mechatron., Vol.32, No.6, pp. 1137-1153, 2020.

[23] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct MonocularSLAM," Proc. of the 13th Eur. Conf. Comput. Vis., Vol.8690, pp. 1-16 doi: 10.1007/978-3-319-10605-2_54, 2014.

[24] M. Rabah et al., "Autonomous Vision-based Target Detection and Safe Landing for UAV," Int. J. Control Autom. Syst., Vol.16, pp. 3013-3025, doi: 10.1007/s12555-018-0017-x, 2018.



Name:
Chihiro Kikumoto

Affiliation:
Department of Information Science, Graduate School of System Informatics, Kobe University

Address:
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan

Brief Biographical History:

2018- Department of Information Science, Graduate School of System Informatics, Kobe University

Membership in Academic Societies:

- The Institute of Systems, Control and Information Engineers (ISCIE)

Name:
Yoh Harimoto

Affiliation:
Department of Information Science, Graduate School of System Informatics, Kobe University

Address:
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan

Brief Biographical History:
2018- Department of Information Science, Graduate School of System Informatics, Kobe University

Name:
Kazuki Isogaya

Affiliation:
Department of Information Science, Graduate School of System Informatics, Kobe University

Address:
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan

Brief Biographical History:
2016- Department of Information Science, Graduate School of System Informatics, Kobe University

Name:
Takeshi Yoshida

Affiliation:
Associate Professor, Research Organization of Science and Technology, Ritsumeikan University

Address:
1-1-1 Noji-higashi, Kusatsu, Shiga 525-8577, Japan

Brief Biographical History:
2015- Assistant Professor, Aoyama Gakuin University
2017- Assistant Professor, Ritsumeikan University
2019- Associate Professor, Ritsumeikan University

Main Works:
- Robotics, computer vision

Membership in Academic Societies:
- The Robotics Society of Japan (RSJ)
- The Society of Instrument and Control Engineers (SICE)
- Information Processing Society of Japan (IPSJ)

Name:
Takateru Urakubo

Affiliation:
Associate Professor, Department of Information Science, Graduate School of System Informatics, Kobe University

Address:
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan

Brief Biographical History:
2001- Research Associate, Faculty of Engineering, Kobe University
2007-2009 Visiting Research Scientist, Carnegie Mellon University
2010-2016 Assistant Professor, Graduate School of System Informatics, Kobe University
2016- Associate Professor, Graduate School of System Informatics, Kobe University

Membership in Academic Societies:
- The Institute of Electrical and Electronics Engineers (IEEE)
- The Society of Instrument and Control Engineers (SICE)
- The Robotics Society of Japan (RSJ)
- The Institute of Systems, Control and Information Engineers (ISCIE)
- The Japan Society of Mechanical Engineers (JSME)
- The Japan Society for Aeronautical and Space Sciences (JSASS)