Available online at www.sciencedirect.com

ScienceDirect

AASRI Procedia 4 (2013) 230 – 235

AASRI Procedia

www.elsevier.com/locate/procedia

2013 AASRI Conference on Intelligent Systems and Control

A Software Scheme for UAV's Safe Landing Area Discovery

Xiaoming Li*

Department of Mechatronics, Zhejiang Sci-Tech University, Xiasha, Hangzhou, China 310018

Abstract

This paper proposes a software processing scheme for small-sized UAV in its landing area discovery using its single onboard camera and machine learning algorithms. The two-stage processing procedure was proposed. In first stage a similarity based textured area identification method was adopted to find the possible landing areas. Afterwards, in second stage, these results were refined and evaluated by using some machine learning algorithms. The UAV can then take use of these results as its emergency landing target options. The software scheme we designed implemented the whole process but still allow the developers to embed their own algorithms to make better results. Our preliminary research has disclosed that this software and application are useful and can provide great convenience and efficiency.

Keywords: UAV; landing area discovery; machine learning, computer vision

1. Introduction

UAV(Unmanned Aerial Vehicle) is widely used in out-door exploration, remote monitor or search and rescue in disasters, due to its abilities such as neglecting the barriers on the ground and reaching the target directly and timely. Vision is the most common sensor that equipped on the UAV because of its small size, light weight, and economy, but providing abundant information. Currently on-service UAVs are mostly remote controlled by human operators; therefore it is the operators' responsibility to interpret the content of

* Corresponding author. Tel.: +86-571-86843349; fax: +86-571-86843349.
E-mail address: xiaominglee@ieee.org.

doi:10.1016/j.aasri.2013.10.035

visions. However, in some emergencies, such as power failure, or lost of control signal, the UAV should be able to find a safe place to land on autonomously and wait for further actions. To achieve this, the first step is to locate a safe landing area as soon as possible. That is, by utilizing the vision information from the onboard camera, together with other data such as GPS or Google map info, the UAV can autonomously find the optimized suitable landing spot, and the landing spot should be both (1) big enough in size for landing, and (2) either preferred landing terrain type such as paved road or flat square, or at least some non-dangerous space, e.g. forest, river surface or big gravels.

In this paper, a software process scheme for computer vision based autonomous landing area discovery for small sized, helicopter type UAV with single camera has been put forward, with detailed explanation of its working principle, process flow, software architecture, together with a prototype application based on it.

2. Related work

Some work has been previously done on the vision based autonomous landing site localization / recognition during the past years. To narrow the results, only those with single onboard camera are taken into account. Among them there were 3 different ways have been attempted to settle this problem.

The first method is to use the natural landmark to localize the landing point. This method is trying to use machine vision instead of the GPS, or to improve the accuracy of landing point. It requires templates for landing target areas beforehand and using some matching algorithms to localize the landing target. These studies included in literature [1][2][10].

The second method is to locate the landing point by calculating the geographic features of the land. It uses a mathematical model to describe the mapping between actual land plane and the plane in the image captured by camera. Given two photos on the same land area taken on two different positions, it can be computed that the surface of the area is a plane or not [3][8][9].

The third method is to classify the terrain by texture features first, then calculate the depth data by optical flow. By texture classification, the area under capture can be marked with safe/unsafe tag, and the depth information can further help to find the hazards during this process [4][6]. There are also some researchers using the second method to calculate the geometric features [5][7].

Among all the three methods, the third one is obviously most suitable for anonymous landing site localization application, because it considers features of both geography and terrain. The first method requires previous knowledge about landing site, and the second one requires large calculations, and not responsive to UAVs flying at the lower height. Therefore, our research was based on the third method.

3. Proposed software processing scheme

Generally the safe landing area for UAV should have at least two features: a flat surface and its size big enough for UAV's dimension and movement. In this research we define the target landing area as a square shape because our UAV is a helicopter with rotating wings and has round shape in itself, and the square shape can be a lot of easier in calculation. Therefore, the main functionality of the software is to process the input video stream data and get the best suitable square and its coordination.

3.1. Basic process procedure

In order to implement the software applications discussed in previous sections, we designed a basic process procedure for the software. It is shown in Fig. 1.

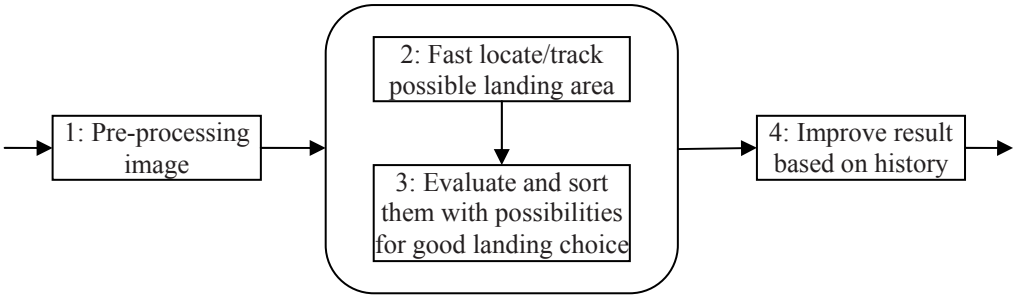

Fig. 1 Basic process procedures for vision based landing area recognition

This procedure can be explained as 4 steps, while step 2 and 3 compose the kernel.

Step 2 is used to fast locating all possible landing areas in initial image(s). It is necessary because to run dedicated, elaborate algorithms to get the best landing spot on the whole image directly is time-consuming and, sometimes, difficult. An image contains so much information, retrieving each individual content in it is still a challenge in Machine Vision. For example, an image composed by road, meadow, trees, and river is very typical in UAV captured images, but to get all such information the image need to be processed again and again by different algorithms using multi-templates, and it is time-consuming. If we can get some possible sub-images, and run these algorithms it would be faster.

Step 3 is a huge complicated process in itself. It is used to evaluate each possible landing areas generated in step 2 and sort them with possibilities for good landing choice, that is, the degree of confidence as the result is a good choice. The output of step 3 should be a sorted list, and the head of the list would be recommended landing site for UAV during this period of time. So, the key problem is how to assign points to these landing site candidates.

This procedure should be run repeatedly, because with the movement of UAV, there should be more landing areas being found. The time period of this big cycle depends on step 3, on how fast it can evaluate all the candidates, and how many images it needs to improve the accuracy.

3.2. Processing procedure of proposed software scheme

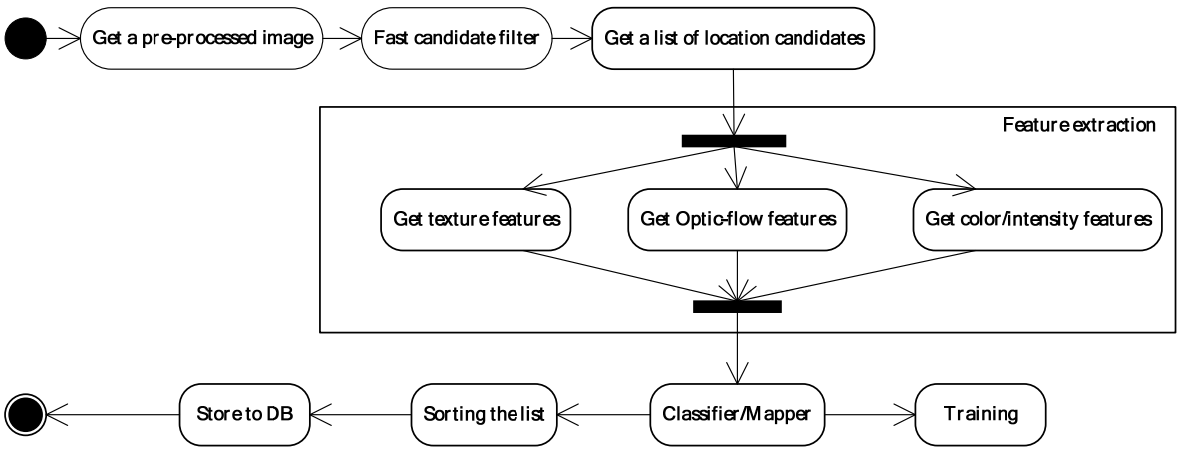

Fig. 2 Architecture for proposed framework

Here shows the designation of procedures for processing the images and getting desired results in Fig. 2. The box with colored background means it is a module that can be replaced by developers. This architecture is Machine-Learning based, because the kernel of it is a classifier. With the help of learned knowledge, the classifier can map the feature vector into a value that indicates DOC (Degree of Confidence) of the pre-outlined candidates.

The difference between this classifier with previous applications in section 2 is that it does not categorize the region of candidates, but just evaluates with possibilities of being suitable for UAV landing. This is a lower level mapper, somewhat similar with phototaxis of animals or plants; therefore it should be easier to be implemented. The mapping process is a typical ML problem; there are various methods can be studied and adopted in this case.

The whole process can be divided into 2 stages. Different methods are deployed in different stages.

3.3. Stage 1: fast landing area candidates locating using texture filters

Through observation and human expert, it is reasonable that safe landing areas are subject to be textured areas, such as paved road or grass land. On the other hand, it means if we can find out all the textured area in an image, the possible acceptable landing areas may be among them. So this is the first supporting point in this method: *textured areas in the images tend to be possible landing areas.*

In order to segment the textured area from the whole image, some algorithms should be adopted. It is well known that self-alikeness should be an important feature of textures. Although some objects have different visual textures under different zoom level, but the self-alikeness won't change in a big distance range. In another way, this self-alikeness feature may be used to judge and segment the textured area from an image, and this method should be robust to different image resolutions. And that is also how this method works. Then we get the second supporting point of this method: *self-alikeness can be used to segment the textured areas in the images.*

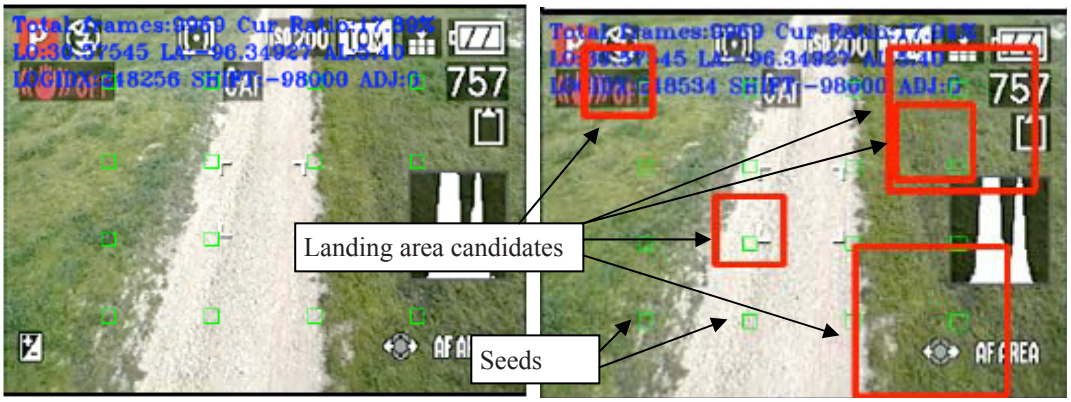

Fig. 3 Output of the stage 1

It is implemented as follows. (1) Randomly locate a point in the image. (2) Generate a square taken the point as the center point; the size of the square is pre-defined. (3) Calculate the texture features. (4) Grow the square, and then calculate the texture features again. (5) Compare the two set of features, and calculate the likeness. (6) If the value of likeness is bigger than a predefined value, go to step (4). (7) Stop and take the last square as the result.

The difficulty lies in how to select features to calculate the similarity between two sub-images. In this paper we use histogram comparison method provided in OpenCV library.

3.4. Stage 2: a machine learning based classifier

The task of stage 2 is to choose the best answer from the options provided in stage 1. Here we use a classifier to achieve this. Classification is a typical machine learning problem, so there are many available methods and algorithms. In current version of implementation, a Naive bayes classifier was implemented. Some extra features are taken into consideration, including the histogram features, optical flow features and geometric features. Among these features the most important one is the optic-flow features. If the vectors distribution is not even in the landing candidate, it is highly possible the candidate is not a flat surface. Fig.4 shows the example output of stage 2.

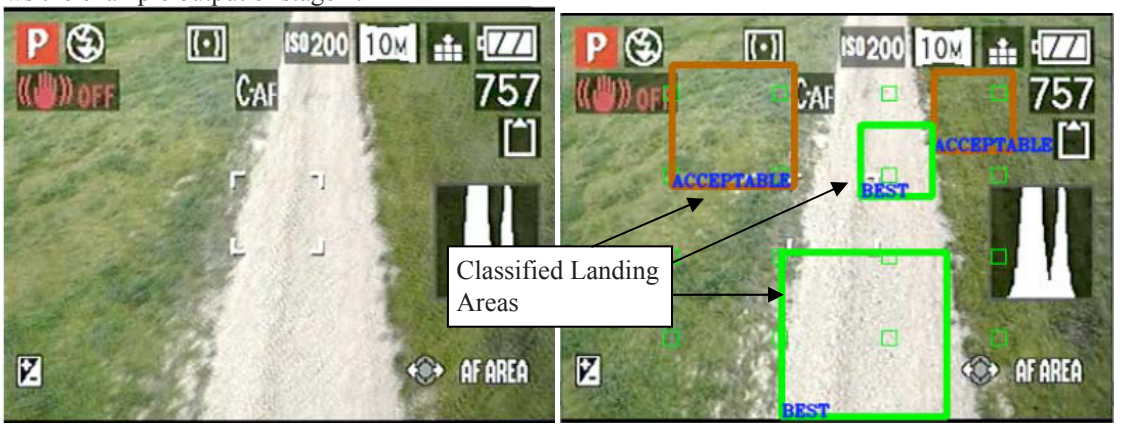

Fig. 4 Experimental result of the machine learning based classifier

4. Software design and implementation

The software of this application is developed as a single loop but handling parallel running of modules through state driven model (as shown in Fig 5(a)). This is highly efficient in processing real-time data. The whole application is mainly composed by 5 modules, and every module has a time-slice to execute part of its own code, depending on the current state. The cooperation of each module is shown in Fig 5(b), which is also the main procedure of the software.

The software was implemented using OpenCV and C language on an Mac Pro Laptop. It can process the video frame at 40 fps and give results after 4 continuous effective frames. However, the finding of possible landing areas is not definite because the classifier may reject all the candidates, and give no output.

5. Conclusion

This paper introduced a work about the autonomous landing area identification for a vision based UAV, including its idea, working principle, procedures and implementation. The main part of it is to find a proper software scheme, which can integrate everything together into a whole program, and applicable to different machine learning algorithms. Currently a Naive Bayes classifier has been tried to calculate the landing area as the output, however there may be other machine learning methods to achieve this, such as fuzzy logic or Artificial Neural Network, which are possible a better methods.

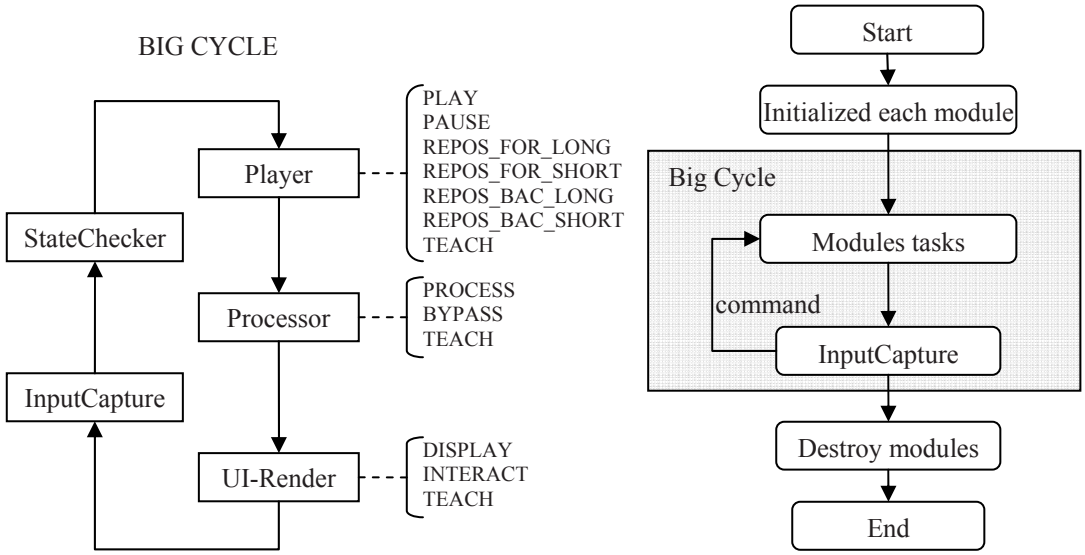

Fig. 5 (a) Big cycle of the main program (b) Main procedure of the program

References

[1] Sharp, C.S., Shakernia, O., Sastry, S.S.. "A Vision System for Landing an Unmanned Aerial Vehicle". Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on , vol.2, no., pp. 1720-1727 vol.2, 2001
[2] A. Cesetti, E. Frontoni, A. Mancini, P. Zingaretti and S. Longhi. "A Vision-Based Guidance System for UAV Navigation and Safe Landing using Natural Landmarks". Journal of Intelligent and Robotic Systems, vol. 57, Numbers 1-4 (2010), 233-257.
[3] Bosch, S., Lacroix, S., Caballero, F.. "Autonomous Detection of Safe Landing Areas for an UAV from Monocular Images". Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, pp.5522-5527, 9-15 Oct. 2006.
[4] Cesetti, A., Frontoni, E., Mancini, A., Zingaretti, P.. "Autonomous Safe Landing of a Vision Guided Helicopter". Mechatronics and Embedded Systems and Applications (MESA), 2010 IEEE/ASME International Conference on , vol., no., pp.125-130, 15-17 July 2010.
[5] Mejias, Luis, Fitzgerald, Daniel L., Eng, Pillar C., & Xi, Liu. "Forced Landing Technologies for Unmanned Aerial Vehicles : Towards Safer Operations.". In Thanh Mung, Lam (Ed.) Aerial Vehicles. In-Tech, Kirchengasse, Austria, pp. 415-442.
[6] Meingast, M., Geyer, C., Sastry, S.. "Vision Based Terrain Recovery for Landing Unmanned Aerial Vehicles". Decision and Control, 2004. CDC. 43rd IEEE Conference on , vol.2, no., pp. 1670-1675 Vol.2, 14-17 Dec. 2004.
[7] Mahmood, W., Shah, S.. "Vision Based Hazard Detection and Obstacle Avoidance for Planetary Landing". Nonlinear Dynamics and Synchronization, 2009. INDS '09. 2nd International Workshop on , vol., no., pp.175-181, 20-21 July 2009
[8] Shakernia, O., Yi Ma; Koo, T.J., Hespanha, J., Sastry, S.S.. "Vision Guided Landing of an Unmanned Air Vehicle". Decision and Control, 1999. Proceedings of the 38th IEEE Conference on , vol.4, no., pp.4143-4148 vol.4, 1999.
[9] Johnson, A., Montgomery, J., Matthies, L.. "Vision Guided Landing of an Autonomous Helicopter in Hazardous Terrain". Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on , vol., no., pp. 3966-3971, April 2005.
[10] Bach Van Pham, Simon Lacroix, Michel Devy. "Vision-based absolute navigation for descent and landing". Journal of Field Robotics, Volume 29, Issue 4, pages 627–647, July/August 2012.
[11] Dongwoon Jeon, Kiho Cho, Doo-Hyun Kim. "Vision-Based Autonomous Landing for Small-Scale Unmanned Rotorcraft". Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW), 2011 14th IEEE International Symposium on, vol., no., pp.274 - 280, 28-31 March 2011.