Vision-Based automatic flight control for small UAVs

Chung-Cheng Chiu, Ching-Tung Lo, Chung-Hsieh Tsai, and Sheng-Yi Chiu

Department of Electrical and Electronic Engineering,
Chung-Cheng Institute of Technology, National Defense University,
No. 190, Sanyuan 1st St., Dashi Jen, Taoyuan, 335 Taiwan.
davidchiu@ndu.edu.tw

Abstract

A vision-based flight control system is developed for application to small unmanned aerial vehicles (UAVs) in this paper. The proposed system integrates a remote controller, a remotely controlled airplane, a camera, a wireless transmitter/receiver, a ground control computer, and the proposed skyline-detection algorithm to achieve automatic control of flight stability. Static and dynamic tests are conducted to validate the system. In the static tests, the average accuracy rate for skyline detection is 98.62%, based on five test videos. In the dynamic tests, straight and circular flights are used to verify lateral and longitudinal stability for the proposed flight control system. The experimental results demonstrate the performance and robustness of the algorithm and the feasibility and potential of a low-cost, vision-only flight control system.

1. Introduction

Recently considerable interest has arisen in small-scale, lightweight, low-cost unpiloted aircraft designed to fly at low altitudes with limited payloads and power constraints. In addition, a robust, reliable, and inexpensive autopilot system is also necessary, and attitude determination and control are the key to achieving autonomy. Using a CCD/CMOS camera for obtaining valuable attitude information is regarded as an alternative approach in attitude-determination and -control research.

In numerous studies, the horizon line is extracted under the assumption of a straight skyline. Bao et al. [1] use an orientation-projection technique applied to binary images to detect the horizon. Moreover, as the peaks in the Hough space can be represented as straight lines, the Hough Transform is employed in [2][3] to choose a maximum peak as the horizon.To make sky and ground segmentation more accurate, Fefilatyev et al. [4] have devised some classifiers to produce better binary images for horizon detection. To extract useful edge features in complex images, Dush et al. [5] propose an edge-based approach to utilizing edge information from the red, green, and blue channels morphologically. In their study of the interference of haze over a skyline, Yuan et al. [6] apply a haze-removal technique to extract a straight skyline from foggy aerial images. Lie et al. [7] propose an edge-linking method for detecting the skyline according to the adjacent characteristics of edge points that extend from one side of the image to the other. Because the skyline separates the image into sky and ground distributions, Ettinger et al. [8] utilize pixel-distribution information from color statistics to find a best-fit horizon that maximizes the between-class variance of the two pixel regions.

To ensure the performance of the technique, it is necessary to take both color and texture features into consideration. Thus Todorovic et al. [9] propose a block-based method for decomposing an image into a number of sub-images.

This manuscript introduces a vision-based flight control system for small UAVs, using a novel skyline-detection algorithm. The system involves systematic integration of the airplane, skyline-detection algorithm, and controller for real-time applications. The most important issues in designing a vision-based system are real-time constraints, robustness to noise, and reliable detection.

This manuscript introduces a vision-based flight control system for small UAVs, using a novel skyline-detection algorithm. The system involves systematic integration of the airplane, skyline-detection algorithm, and controller for real-time applications. The most important issues in designing a vision-based system are real-time constraints, robustness to noise, and reliable detection.

2. The proposed system

The vision-based flight control system comprises a UAV subsystem and an image processing and transmitter subsystem. The UAV subsystem integrates a video acquisition module, a remote control receiver module, a servo control module, and a power supply module on a radio-controlled aircraft. The aircraft is a SOARJET EP-19 (fuselage length: 87 cm), a high-wing (wingspan: 140 cm), four-channel (controlling rudder, ailerons, elevator, and throttle), battery-powered, single-motor aircraft. The aircraft weighs about 600 g and carries the four modules, which have a combined weight of roughly 187 g. The video acquisition module consists of a 1/3-inch CMOS camera $20 \times 20 \times 20$ mm in size and 15 g in weight, together with a 1.2-GHz wireless transmitter and antenna.

The image-processing and transmitter subsystem comprises a video receiver module, a ground control computer module, and a remote control transmitter module. The video receiver module is responsible for receiving the analog video signals from the video transmitter of the UAV subsystem. When the video signals are received by the video receiver, they are sent to a frame-grabber card in the ground control computer (GCC). The images are digitalized by the grabber card and then processed by the GCC, which is equipped with a Pentium IV 2.6-GHz processor and 1-GB RAM.

The tasks performed by the GCC include digitalizing

the video signals, detecting the skyline, estimating the attitude of the aircraft, and sending the control signals to the remote control transmitter. The control signals from the GCC are sent to the remote control transmitter through an analog/digital converter and a RS232 connecter.

3. Skyline-detection algorithm

Captured images are first transferred from the color plane (RGB) to the grayscale plane, and the region of interest (ROI) is defined at the boundaries of the captured images. The proposed skyline-detection algorithm is composed of initial and tracking stages.

A. The detection algorithm for the initial stage

The detection algorithm for the initial stage uses image blocks of $s \times s$ pixels, which overlap half the width of the neighboring blocks for increased accuracy along the ROI while searching the candidate blocks (where the terminal points of the skyline may be located).

Let the gray-level distribution of block b_l range from 1 to m. The probability distribution of gray-level i is denoted by:

$$p_i = \frac{C_i}{\sum_{j=1}^{m} C_j}, \text{ where } i = 1,2,\cdots,m, \text{ and } \sum_{i=1}^{m} p_i = 1. \quad (1)$$

The parameters C_i and $\sum_{j=1}^{m} C_j$ denote the number of pixels with gray-level i and the total number of pixels in a given block, respectively. The mean and variance of b_l are given by

$$\mu_{bl} = \sum_{i=1}^{m} i p_i \quad \text{and} \quad (2)$$

$$\sigma_{bl}^2 = \sum_{i=1}^{m} (i - \mu_{bl})^2 p_i . \quad (3)$$

To distinguish whether a block belongs to the sky or the ground, the algorithm compares the mean value to a given threshold via the following equation:

$$Label_{bl} = \begin{cases} 1 \ if \ \mu_{bl} > T_1 \\ 0 \ otherwise \end{cases}, \quad (4)$$

where $Label_{bl}$ represents either a sky or ground block, identified by 1 or 0, respectively.

To select the threshold T_1, a pixel-based thresholding technique is adopted. Suppose that the pixel values of a given ROI with grey-level {1, 2, … , L} can be divided into two classes, C_{sky} = {1, 2, ... , k} and C_{ground} = {k+1, k+2, … , L}, where k is the threshold value. Given that the probabilities of the pixel values of the two classes are C_{sky}: {p_1, p_2, … , p_k} and C_{ground}: { p_{k+1}, p_{k+2},…, p_L}, the probabilities of the classes are

$$\omega_{sky}(k) = \sum_{i=1}^{k} p_i \quad \text{and} \quad (5)$$

$$\omega_{ground}(k) = \sum_{i=k+1}^{L} p_i . \quad (6)$$

The mean values of the two classes and the total mean value μ_T in the ROI are computed as follows:

$$\mu_{sky}(k) = \sum_{i=1}^{k} i \times \frac{p_i}{\omega_{sky}(k)} \quad (7)$$

$$\mu_{ground}(k) = \sum_{i=k+1}^{L} i \times \frac{p_i}{\omega_{ground}(k)} \quad (8)$$

$$\mu_T = \omega_{sky}(k) \times \mu_{sky} + \omega_{ground}(k) \times \mu_{ground} = \sum_{i=1}^{L} i \times p_i \quad (9)$$

On the basis of the aforementioned statistical formulations, the between-class variance and the total variance in the ROI are given by

$$\sigma_B^2(k) = \omega_{sky} \times (\mu_{sky} - \mu_T)^2 + \omega_{ground} \times (\mu_{ground} - \mu_T)^2 \quad (10)$$

$$\text{and} \quad \sigma_T^2 = \sum_{i=1}^{L} (i - \mu_T)^2 p_i . \quad (11)$$

To measure the separability between the sky and the ground, a separable factor is applied to the algorithm. The factor is defined by

$$SF(k) = \frac{\sigma_B^2(k)}{\sigma_T^2}, \quad (12)$$

where σ_T^2 serves as a normalization factor, and $SF(k)$ is a normalization value. Accordingly, an optimal threshold for distinguishing a sky block from a ground block can be obtained when the separable factor is maximized. Hence, the threshold T_1 is obtained from Eq. (13).

$$T_1 = \arg\max_{1 < k < L} \{SF(k)\} \quad (13)$$

To confirm the candidate blocks, the algorithm compares the block variance with a variance-based threshold using the following criterion:

$$Candidate_{bl} = \begin{cases} true \ , if \ \sigma_{bl}^2 > T_2 \\ false \ , otherwise \end{cases} \quad (14)$$

where $Candidate_{bl}$ represents the chosen candidate block. As the candidate block contains both sky and ground regions, the variance of the intensity in the candidate block is large. Let the total number of blocks in the ROI be N. The threshold T_2 is then defined by

$$T_2 = \frac{1}{N} \sum_{l=1}^{N} \sigma_{bl}^2 . \quad (15)$$

Finally, the intersection of sky and ground is extracted from the candidate block to complete the skyline-detection process. On the basis of the aforementioned algorithm, the details of the proposed algorithm for the initial stage are as follows.

Step 1: Transform the captured color image into a gray-level image $g(x, y)$, where $0 \le x < m_W$ and $0 \le y < n_H$.

Step 2: Partition the ROI into blocks of size $s \times s$ pixels, which overlap the neighboring blocks to a width of s/2 pixels. The total number of blocks in the ROI is N.

Step 3: Compute the block mean μ_{bl} and block variance σ_{bl}^2 of each block via Eqs. (2) and (3), where $\{l \in Z : 1 \le l \le N\}$.

Step 4: Compare the block mean μ_{bl} with the threshold T_1 determined by Eq. (13). If $\mu_{bl} > T_1$, mark the block as a sky block and set $label_{bl} = 1$; otherwise mark it as a ground block and set $label_{bl} = 0$.

Step 5: Search for two candidate blocks in both the left- and right-hand directions, starting from one of the sky blocks. Compare the ground-block variance σ_{bl}^2 with the threshold T_2 determined by Eq. (15). If $\sigma_{bl}^2 > T_2$, mark the ground block as a candidate block; otherwise relabel the ground block as a sky block. Repeat the processing until two candidate blocks are found.

Step 6: Segment the pixels of the two candidate blocks into sky and ground regions using the mean values of the

candidate blocks. In the candidate blocks, the boundaries between the sky and ground regions define the edge points of the skyline. The edge points intersect the image frame at the terminal points of the skyline.

B. The detection algorithm for the tracking stage

In the tracking stage, a new ROI is confined to the estimated region of the terminal points of the skyline to avoid image interference and increase the processing speed. As the time interval between two consecutive frames is minimal, the estimation can be accomplished using equations of motion without acceleration:

$$\begin{cases} x_{f+1} = x_f + \dot{x}_f \cdot \Delta t \\ y_{f+1} = y_f + \dot{y}_f \cdot \Delta t \end{cases}, \tag{16}$$

where x_f and y_f are the coordinates of the terminal points in the *f*-th frame, and Δt is the time interval. The linear Kalman filter employed to track the terminal points from consecutive frames is modeled by

$$\mathrm{X}_{f+1} = \Phi \cdot \mathrm{X}_f, \tag{17}$$

where the state vector is defined by

$$\mathrm{X}_{f+1} = \left[x_{f+1}, y_{f+1}, \dot{x}_{f+1}, \dot{y}_{f+1} \right]^T \text{ and } \mathrm{X}_f = \left[x_f, y_f, \dot{x}_f, \dot{y}_f \right]^T, \tag{18}$$

and the state transition matrix derived from Eq.(16) is given by

$$\Phi = \begin{vmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}. \tag{19}$$

After applying the Kalman filter, the estimated coordinates are regarded as the central point of an $r \times r$ search area. The terminal point search is conducted in this area.

To extract the edge features of the search area, the algorithm compares the magnitude of the gradient with a given threshold via the following equation:

$$Edge_{point} = \begin{cases} true, & if\ mag(\nabla g(x,y)) > T_3 \\ false, & otherwise \end{cases}, \tag{20}$$

where the gradient vector of an image $g(x,y)$ is defined as

$$\nabla g = \begin{bmatrix} G_x \\ G_Y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial g}{\partial x} \\ \dfrac{\partial g}{\partial y} \end{bmatrix}, \tag{21}$$

and the magnitude of the gradient vector is given by

$$mag(\nabla g) = \left[G_x^2 + G_y^2 \right]^{1/2}. \tag{22}$$

For the sake of computational efficiency, the gradient operational equation applied to approximate Eq. (22) has the form

$$\begin{cases} G_x(x,y) = mask_x * l(x,y) \\ G_y(x,y) = mask_y * l(x,y) \end{cases}, \tag{23}$$

where $*$ denotes convolution, $l(x,y)$ is a neighborhood of (x,y), and the two orthogonal masks are defined by

$$mask_x = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}, \quad mask_y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}. \tag{24}$$

The threshold value T_3 is defined as

$$T_3 = \begin{cases} 50, & if\ \beta \times \mu_r > 50 \\ \beta \times \mu_r, & if\ \beta \times \mu_r \le 50 \end{cases}, \tag{25}$$

where the mean value of the local gradient image is given by

$$\mu_r = \frac{1}{r^2} \sum_{x=0}^{r-1} \sum_{y=0}^{r-1} g(x,y), \tag{26}$$

and the weight coefficient β is set equal to 0.35.

The detected edge points may include unnecessary points, such as clouds, terrain, or noise. To determine the candidates for the terminal points of the skyline from among the edge points, the algorithm compares the difference of sky and ground with a given threshold T_4 via the following criterion:

$$Edge_{\text{terminal}} = \begin{cases} true, & if\ \left| \mu_{up} - \mu_{down} \right| \ge T_4 \\ false, & otherwise \end{cases}, \tag{27}$$

where μ_{up} and μ_{down} are the mean values of the upward and downward vertical pixels of an edge point, respectively. A terminal point of the skyline is found when $\left| \mu_{up} - \mu_{down} \right|$ is maximized.

Once the two terminal points of the skyline have been detected, the horizon line connecting the terminal points can be used to estimate the roll angle (ϕ) and pitch distance (η) as the flight attitude parameters. The roll angle can be derived from the inverse tangent of the slope of the horizon using Eq. (28).

$$\begin{cases} \phi = \arctan\left(\left(\dfrac{j_{right} - j_{left}}{i_{right} - i_{left}} \right) \right) \times \dfrac{180}{\pi} \\ -90^\circ < \phi < 90^\circ \end{cases}. \tag{28}$$

The pitch angle cannot be measured accurately from the horizon of an image, as the pitch angle is related to the altitude of the UAV and the distance to the horizon. Therefore, the distance $\overline{MD}$ from the central point of the image to the horizon, known as pitch distance η, is substituted to estimate the pitch angle.

$$\overline{MD} = \left| \vec{LM} \right| \cdot \sin(\xi), \quad \text{where } \xi = \arccos\left(\frac{\vec{LM} \cdot \vec{LR}}{\left| \vec{LM} \right| \cdot \left| \vec{LR} \right|} \right). \tag{29}$$

4. Experimental results

Before actual test flights are conducted, five test videos are recorded under varying weather and illumination conditions to evaluate the accuracy of the proposed algorithm. Test videos 1 to 3 are captured with an onboard camera, and test videos 4 and 5 are downloaded from the websites [http://vimeo.com/2014108 and http://vimeo.com/9314939] to enrich the test sample. The skyline-detection accuracy for a test video is defined as

$$\text{Accuracy} = \frac{N_C}{N_T} \times 100\%, \tag{31}$$

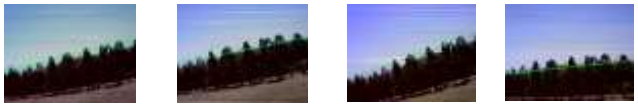
where N_C is the number of correctly detected frames, and N_T is the total number of frames in the video. By definition, the skyline is correctly detected when the deviation between the terminal points detected by the proposed algorithm and the human eye is less than five pixels. Based on the test results of the five test videos, the average accuracy rate is 98.62%.

Figure 1 shows that both straight and uneven skylines

can be successfully detected. Moreover, even though the number of edge points of a skyline may change under different lighting conditions, the terminal points can still be located (as Fig. 2 indicates) as long as the skyline is distinguishable by visual inspection.

frame n frame n+4 frame n+9 frame n+14

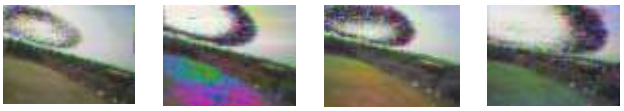
Fig. 1 Skyline detection with varying skyline geometries



frame n frame n+4 frame n+9 frame n+14

Fig. 2 Skyline detection with varying illumination

Noise-corrupted images may cause serious errors in vision-based methods. Fortunately, when compared with other global detection methods, the proposed algorithm restricts the errors to a limited area surrounding the previously detected terminal points. Thus, flight attitude may not change dramatically due to error detection. Using the tracking algorithm, the errors may also be recovered in a few frames, as shown in Fig.3.

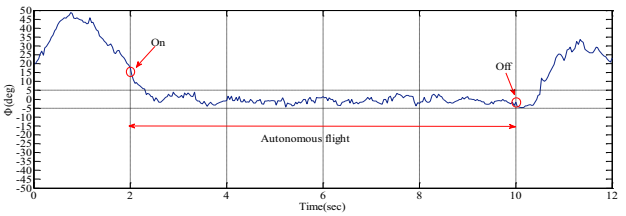


frame n frame n+4 frame n+9 frame n+14

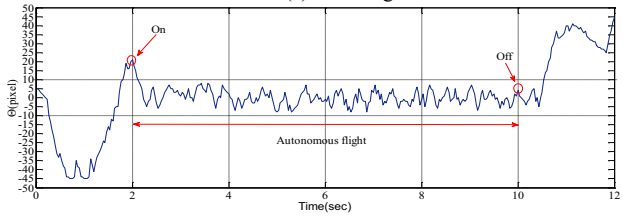
Fig. 3 Skyline detection with varying noise

For performance analysis, the proposed algorithm is coded in Microsoft Visual C++ 2005 and operated with an image size of 320×240 pixels on a Pentium IV 2.6-GHz processor with 1 GB of memory. In the tracking stage, the total processing time from image acquisition to voltage output is around 31.2 milliseconds.

To demonstrate lateral and longitudinal stability, two test scenarios are provided: straight flight and circular flight. The variations in roll angle and pitch distance are shown in Fig.4.



(a) Roll angle



(b) Pitch distance

Fig. 4 Variations of the roll angle and pitch distance in a straight flight

5. Conclusions

This study introduces a vision-based flight control system for small UAVs using a skyline-detection algorithm. As the experimental results indicate, this low-cost flight control system offers the advantages of real-time constraint, robustness to noise, and reliable detection. Furthermore, the proposed skyline-detection algorithm can detect both straight and uneven skylines with an average accuracy rate of 98.62%, based on the five test videos. The flight test results demonstrate that the system is able to control and stabilize a UAV with vision-only flight control. In future research, we will add rudder and throttle controls and transmit the GPS signal to the GCC to fully control the automatic flight of a UAV.

Acknowledgments: This work was also partially supported by National Science Council of Taiwan under the Grant No. NSC 99-2221-E-606 -020.

References

[1] G. Q. Bao, S. S. Xiong, and Z. Y. Zhou, ”Vision-based horizon extraction for micro air vehicle flight control,” IEEE Transactions on Instrumentation and Measurement, vol. 54, pp. 1067-1072, June 2005.

[2] T. D. Cornall, G. K. Egan, and A. Price, “Aircraft attitude estimation from horizon video,” Electronics Letters, vol. 42, no. 13, pp.744-745, June 2006.

[3] G. A. S. Pereira, P. Iscold, and L. A. B. Torres, “Airplane attitude estimation using computer vision: simple method and actual experiments,” Electronics Letters, vol. 44, no. 22, Oct. 2008.

[4] S. Fefilatyev, V. Smarodzinava, L.O. Hall, and D. B. Goldgof, “Horizon detection using machine learning techniques,” The 5th International Conference on Machine Learning and Applications, pp. 17-21, Dec. 2006.

[5] D. Dusha, W. Boles, and R. Walker, “Fixed-wing attitude estimation using computer vision based horizon detection,” in Proc. of the 12th Australian International Aerospace Congress, pp. 1–19, Melbourne, Australia, April 2007.

[6] H. Z. Yuan, X. Q. Zhang, and Z. L. Feng, “Horizon Detection in Foggy Aerial Images,” in IEEE International Conference on Image analysis and Signal Processing, pp. 191-194, Zhejiang, China, April 2010.

[7] W. N. Lie, T. C. I. Lin, T. C. Lin, and K. S. Hung, ”A robust dynamic programming algorithm to extract skyline in images for navigation,” Pattern Recognition Letters, vol. 26, no. 2 , pp. 221-230, Jan. 2005.

[8] S. M. Ettinger, M. C. Nechyba, P. G. Ifju, and M. Waszak, "Vision-guided flight stability and control for micro air vehicles," in Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2134-2140, Switzerland, Oct. 2002.

[9] S. Todorovic and M. C. Nechyba, “A Vision System for Intelligent Mission Profiles of Micro Air Vehicles”, IEEE Transaction on Vehicular Technology, vol. 53, no. 6, pp. 1713-1725, Nov.2004.