UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Visión Electrónica
Más que un estado sólido

https://doi.org/10.14483/issn.2248-4728

VISIÓN ELECTRÓNICA
A RESEARCH VISION

Autonomous trajectory tracking for an UAV based on computer vision

Seguimiento autónomo de trayectoria para un UAV basado en visión artificial

Jorge Daniel Gallo-Sanabria [1], ***Paula Andrea Mozuca-Tamayo*** [2], ***Rafael Iván Rincón-Fonseca*** [3]

INFORMACIÓN DEL ARTÍCULO

Historia del artículo:
Enviado: 06/10/2019
Recibido: 17/12/2019
Aceptado: 21/01/2020

Keywords:
Computer vision
Control law
Thresholding
Trajectory Tracking
UAV

Palabras clave:
Visión artificial
Ley de control
Umbral
Seguimiento de trayectoria
UAV

ABSTRACT

The trajectory tracking performed by unmanned aerial vehicles has several advantages that can be extended to many industrial applications from package delivery to agriculture. However, this involves several challenges that depend on the way tracking is performed, particularly when using computer vision. In this article, we show the design, the simulation and the implementation of a simple algorithm for trajectory tracking using computer vision. This algorithm is executed on a drone that successfully navigates to a desired location.

RESUMEN

El seguimiento de trayectorias por parte de vehículos aéreos no tripulados trae consigo varias ventajas que pueden llevarse a aplicaciones que van desde la mensajería hasta la agricultura. Sin embargo, esto involucra diferentes desafíos dependiendo de la forma como se realice; particularmente en el caso del seguimiento de trayectorias haciendo uso de la visión artificial. Este artículo describe el diseño, la simulación y la implementación de un algoritmo simple para el seguimiento de una trayectoria a través de la visión artificial, que permite el seguimiento de un dron y su aterrizaje en un punto deseado.

[1] BSc. in Mechatronics engineering, Universidad Nacional de Colombia, Colombia. E-mail: jdgallos@unal.edu.co.
[2] BSc. in Mechatronics engineering, Universidad Nacional de Colombia, Colombia. E-mail: pamozucat@unal.edu.co.
[3] BSc. in Mechatronics engineering, Universidad Nacional de Colombia, Colombia. E-mail: ririnconf@unal.edu.co.

Cite this article as: J. D. Gallo-Sanabria, P. A. Mozuca-Tamayo and R. I. Rincón-Fonseca, "Autonomous trajectory tracking for an UAV based on computer vision", Visión electrónica, vol. 14, no. 1, pp. 51-56, january 2020.
https://doi.org/10.14483/22484728.15968

1. Introduction

Process automation has greatly impacted industry over the years. Recently, new technologies (e.g. drones) have been impacted. Several companies worldwide are now using drones to accomplish various tasks. For success, drones must follow given trajectories autonomously. When drones were initially developed, they were guided primarily by GPS coordinates with constant human supervision because of drone's lack of obstacle avoidance and tracking. Today, drones use cameras that constantly acquire images during flight, allowing the use of computer vision to automatically avoid obstacles and follow trajectories without constant supervision. MathWorks has designed a toolbox that allows the Parrot Mambo drone to be programmed with autonomous behaviours. With the Parrot Mambo toolbox, it is possible to read the values from sensors on the drone, obtain images from its camera and individually control the motors. Considering the fact that the drone is very small and that it needs to operate in constricted environments, simplicity, navigation and jerk-avoidance are the top goals.

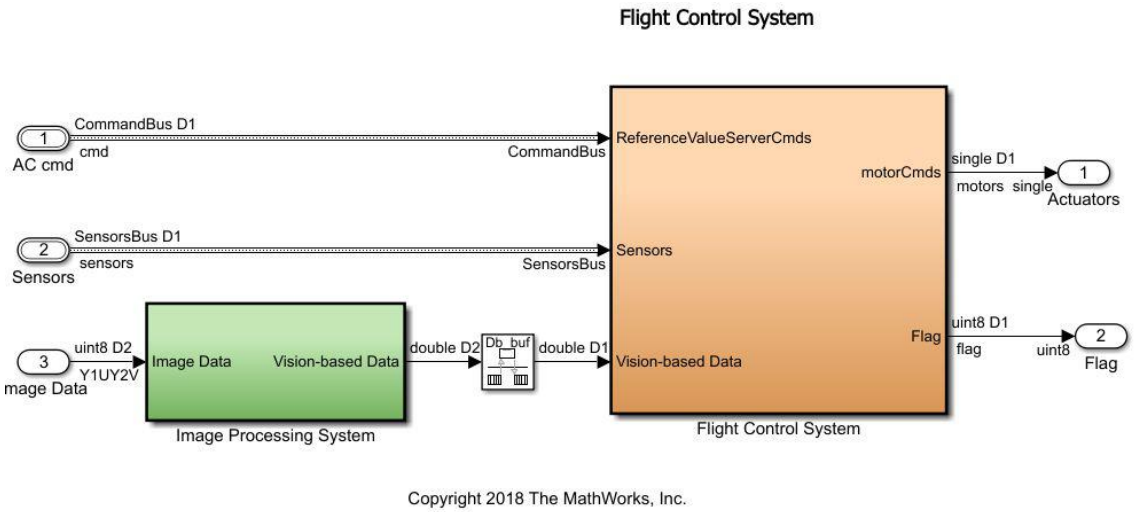
In Section 2, hardware and software resources are introduced, and the methodology used to acquire and process imagery is discussed alongside the computer vision and control algorithm. Experimental results are shown in Section 3, and Section 4 contains further discussion and the conclusion.

2. Materials and Methods

The Parrot Mambo has a gyroscope–accelerometer that provides acceleration and inclination data from three main axes. The drone has an ultrasound sensor for applications involving object proximity, and it wields a low-resolution (120 × 160) camera that supports real-time image collection for computer-vision processing. MathWorks' *Simulink Support Package for Parrot Minidrones* was used with MATLAB to allow communications with Simulink, providing a computer-vision programming capability [1]. The toolbox generates C code to program the drone's microcontroller.

Figure 1 visualizes the system that controls the actuators and allows programming. The system displays sensor readings, camera feeds and a clock to the user. The output of the control system controls the drone's actuators.

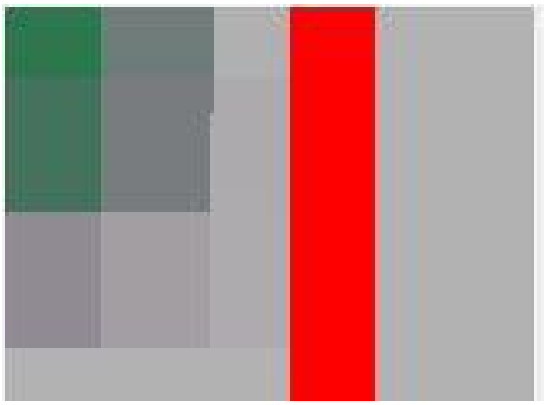
Figure 1: Flight-control system [2].



2.1. Image Acquisition and Treatment

Camera imagery produces YCbCr colour-space data, which we chose to convert into red– green–blue (RGB) to allow easier manipulation and comprehension of the images. After this initial conversion, it is possible to appreciate the images obtained by the drone in the simulation and during the flight as it can be seen in the Figures 2 and 3 respectively.

Figure 2: Screenshot of the drone simulation.



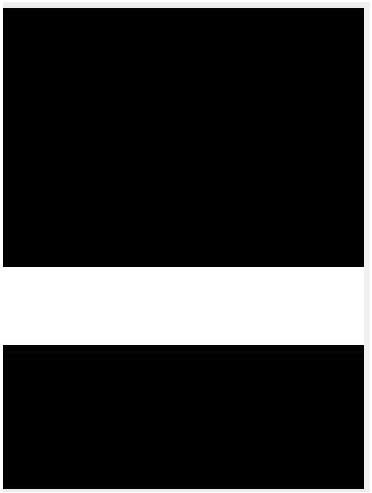
Source: own

Figure 3: Pictures taken with the Parrot Mambo camera.

Source: own

After the data is adequately represented, it is necessary to ensure that the drone processes the appropriate information corresponding to the trajectory. A thresholding mask is used to separate the trajectory from the rest of the surface, requiring a thresholding value of separation. This also allows posterior processing on the image. This value is currently manually obtained, but the goal is to automate this process. We take the value of the RGB matrix (with hue and saturation values if the lighting conditions are unstable) at a position of certainty during the initial moments of flight. The colour corresponding to the track is used as the seed, assuming the drone holds a steady position during the first few seconds. When the seed is determined, an interval of acceptable values is considered, owing to variations, as part of the track. Thus, a range of upper and lower values from the seed is given. Finally, thresholding is used to perform banalization to ease the track-analysis burden. Thresholding simulation can be seen in Figure 4.

Figure 4: Image capture of the simulated drone after passing through the mask and seed algorithm.



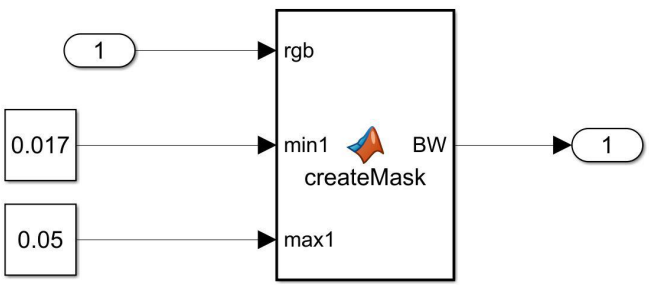
Source: own

Figure 5 displays the block diagram used for creating the mask. In this diagram, the image taken by the drone's camera is used as input to establish the filtering range during the route. The output is a black-and-white Boolean with the applied mask.

Figure 6 displays twos shots from the camera after filtering. One image is taken the star of the track (left) and a forward curve of the track (right).

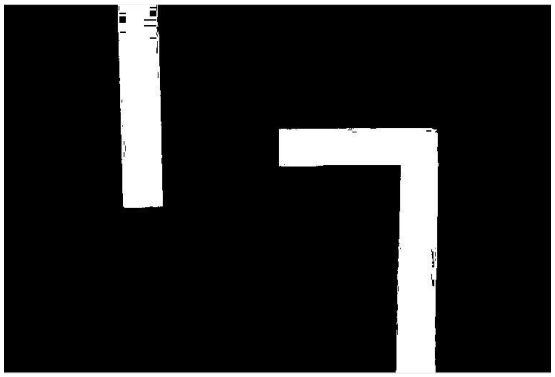
During image processing, it is important to obtain values of area, perimeter and orientation. Thus, A block for the region (blob) analysis is employed to obtain the necessary parameters from the Boolean images.

Figure 5: Block diagram of creating the mask and filter.



Source: own

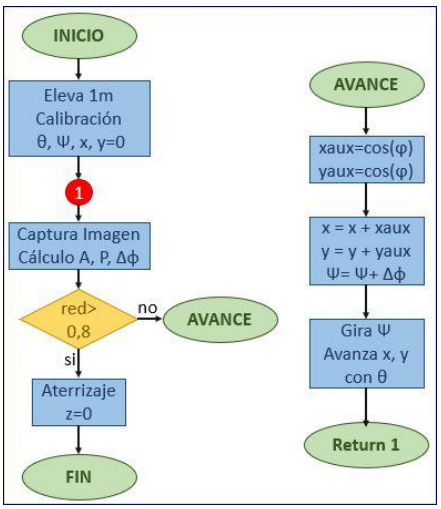
Figure 6: Image capture of the Parrot Mambo camera after undergoing the mask and seed algorithm.



Source: own

During the first part of image processing, a concatenation is performed using RBG matrices to obtain a single work matrix. Then, the matrix is rotated 90°, as seen in Figure 4. This is made, because Simulink calculates the orientation from the vertical axis of the image, and, when the track is oriented vertically, the values of the angle can be around 179° and 180° instead of negative. After thresholding and filtering the image, region analysis is performed.

Figure 7: Flight-control algorithm.



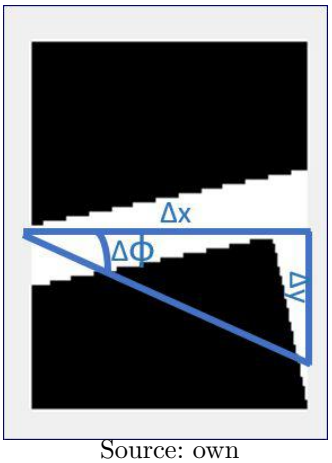
Source: own

To calculate the orientation, an ellipse is drawn, which must contain all the white pixels in the image to calculate the angle between the major axis of the ellipse and the x-axis. The area is the number of white pixels in the image, and the perimeter is the number of white pixels at the edges of each object.

2.2. *Tracking Algorithm*

Algorithm design is based on the acquisition and processing of image data. Via the algorithm, the drone follows the route of the track to arrive at a landing point (circle). When arriving at the circle, the drone will descend smoothly, avoiding jerking effects. The algorithm can be seen in Figure 7.

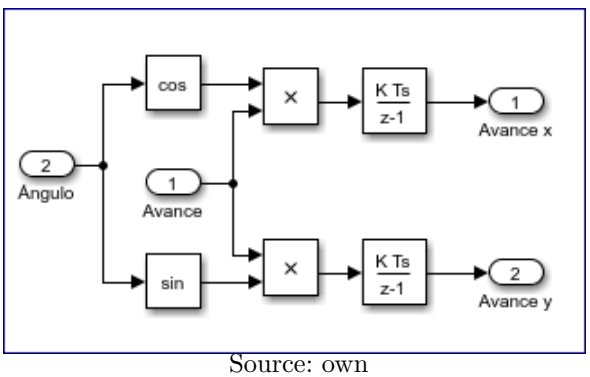
Figure 8: Obtaining orientation and position variations.



Source: own

The actual route is, therefore, based on the orientation acquired via computer vision, considering 2D image data (Figure 8). Movement, therefore, is restricted to two axes: *pitch* and yaw. The system uses a fixed-coordinate system oriented at the take-off location. The x-axis is horizontal and the y-axis is vertical. Data obtained from orientation ($\Delta\phi$) correspond to the angle of rotation required for the drone to remain on track. However, owing to the fact that the system is based on a fixed reference, progressive changes in angle are taken. *Pitch* movements are used advance x and y coordinates (Figure 9), which are calculated based on angular changes via sin and cos functions.

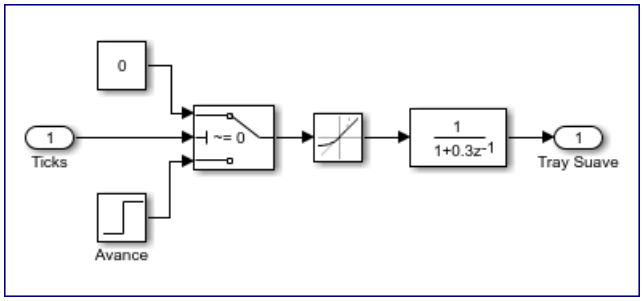
Figure 9: Moving in Simulation.



Source: own

During flight, smooth movement is desired without rapid changes in acceleration. Thus, a *rate limiter* is used with a discrete filter to smoothen the slope between the changes of angle and position (Figure 10).

Figure 10: Smoothing of the paths.



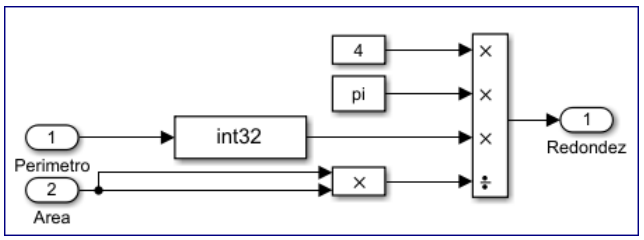
Source: own

To verify the presence of the circle landing point in the, the property of roundness is used (Figure 11). This is the relation between the area and the perimeter previously obtained, as shown in Eq. (1) [3].

$$rpund = \frac{4\pi A}{p^2} \tag{1}$$

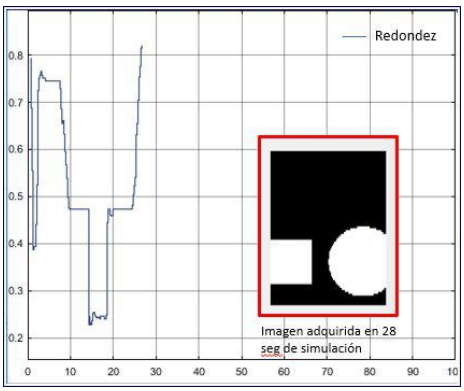
where A the area, and p is the perimeter.

Figure 11: Roundness calculation in simulation.



Source: own

Figure 12: Roundness result throughout the simulation.



Source: own

When the roundness takes a high value ($> 0{,}8$), a circle is indicated and the drone will land softly and smoothly.

3. Results

When the algorithm functioned appropriately, the drone achieved the desired goals of following the trajectory and landing smoothly at the desired point. A track with three linear segments was used alongside the circle indicating the landing point. The illumination conditions were controlled to ensure only small variations in calibration values of the thresholding mask. It was necessary to make adjustments to the roundness condition, because shadows created roundness values that fooled the camera.

Figure 13: The Parrot Mambo and the track.

Source: own

4. Conclusions

Using a computer-vision-based algorithm that performs image segmentation in real time, a track was successfully followed autonomously by a Parrot Mambo drone, and automated flight was achieved, including landing. A rate limiter was used to smoothen slope changes during descent. For future works, an automated mask is needed to avoid the manual calibration process. Additionally, adequate flight needs to be achieved without fully controlled illumination. We noted that, during the simulation and the implementation, the drone required a servovisual control system capable of maintaining centred and steady trajectory based on the captured images, because, when the drone rotated, position and image mismatches occurred, increasing because of the actions of the integrator.

5. Acknowledgements

Thanks to MathWorks for allowing us the use of their toolbox and copyrighted materials and for taking charge of the National Minidrone Competition of 2018. Thanks to Maitreyee Mordekar, a member of the MathWorks India team, for her advice for improving the simulation and achieving adequate implementation of our algorithm. Thanks to Ricardo Ramirez, teacher of the Universidad Nacional de Colombia, for encouraging us to participate in the contest, for promoting the formation of our investigation group and for obtaining the necessary tools to promote the researching process. Thanks to Flavio Prieto and German Ramos, teachers of the Universidad Nacional de Colombia, whose knowledge and advice helped us during the creation of the algorithm.

References

[1] MathWorks, "PARROT Minidrones Support form Simulink", 2019, [Online] available at: https://www.mathworks.com/hardware-support/parrot-minidrones.html

[2] MathWorks, "Compute Statistics for Labeled Regions", 2019, [Online] available at: https://www.mathworks.com/help/vision/ref/blobanalysis.html

[3] M. A. Arias-Marta and J. A. Sierra-Ruiz, "Procesamiento de Imágenes para la Clasificación de Café Verde", thesis, Pontificia Universidad Javeriana, Colombia, 2016.