Check for updates

Research Article

INTERNATIONAL JOURNAL OF ADVANCED ROBOTIC SYSTEMS

International Journal of Advanced Robotic Systems
September-October 2017: 1–15
© The Author(s) 2017
DOI: 10.1177/1729881417731083
journals.sagepub.com/home/arx
SAGE

Helipad detection for accurate UAV pose estimation by means of a visual sensor

Cosimo Patruno, Massimiliano Nitti, Ettore Stella and Tiziana D'Orazio

Abstract
In this article, we tackle the problem of developing a visual framework to allow the autonomous landing of an unmanned aerial vehicle onto a platform using a single camera. Specifically, we propose a vision-based helipad detection algorithm in order to estimate the attitude of a drone on which the camera is fastened with respect to target. Since the algorithm should be simple and quick, we implemented a method based on curvatures in order to detect the heliport marks, that is, the corners of character H. By knowing the size of H mark and the actual location of its corners, we are able to compute the homography matrix containing the relative pose information. The effectiveness of our methodology has been proven through controlled indoor and outdoor experiments. The outcomes have shown that the method provides high accuracies in estimating the distance and the orientation of camera with respect to visual target. Specifically, small errors lower than 1% and 4% have been achieved in the computing of measurements, respectively.

Keywords
Visual landing, helipad detection, UAV pose estimation, computer vision

Date received: 28 March 2017; accepted: 7 August 2017

Topic: Vision Systems
Topic Editor: Antonio Fernandez-Caballero
Associate Editor: Grazia Cicirelli

Introduction

Nowadays, the mobile robot localization has an important role in the development of robot systems. A continuous monitoring of the vehicle pose is mandatory when dealing with autonomous real-time applications. The pose estimation is achieved by means of onboard sensors[1] such as odometers, inertial measurement units, infrared and ultrasonic sensors, Radio Frequency IDentification (RFID) tags,[2] visual sensors, and so on. Among all of these, vision systems have opened new scenarios concerning the accurate robot localization,[3–5] taking advantage of their accessible cost.

The localization problem also affects the field of aerial robotics, since autonomous pose estimation can enable unmanned aerial vehicles (UAVs) to be employed in several innovative scenarios of great interest. The most common applications of UAVs are rescue operations and medical assistance, inspection and monitoring of disaster areas, surveying and mapping of areas inaccessible to humans, commercial aerial surveillance, forest fire detection, military services, and several other applications.

One of the most challenging tasks for UAV is landing. In this regard, the aerial vehicles must be able to

National Research Council of Italy-Institute of Intelligent Systems for Automation, Bari, Puglia, Italy

Corresponding author:
Cosimo Patruno, National Research Council of Italy-Institute of Intelligent Systems for Automation, Amendola 122/D, Bari, Puglia 70126, Italy.
Email: patruno@ba.issia.cnr.it

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (http://www.creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).

autonomously land onto a dynamic or moving platform. During these operations, a high safety level is required in order to avoid damages to themselves or other external agents. Most of the autonomous landing systems employ computer vision (CV) algorithms as well as other auxiliary sensors as Global Positioning Systems, gyroscopes, and accelerometers (Inertial Measurement Units [IMUs]), and so on. These algorithms are able to recognize known patterns which can be natural or artificial (i.e. human-made). In the latter category, different methods were proposed over the years in order to detect artificial drone pads such as helipad-shaped targets, landing platforms made of concentric circles,[6,7] squared markers,[8] and so on.

In this article, we propose a vision-based method able to provide the pose of a landing pad with respect to the camera. The pose estimation of UAV on which the camera is fastened is implicitly computed. The algorithm has to support continuously the flying vehicle during its landing phase. We consider the standard helipad (made of the marker H inside circles) as visual target because it is commonly located at airports, on the top of skyscrapers, on large ships and oil platforms, hospitals, and so on. By computing the curvatures of H contour, we are able to identify its 12 corners, which are then used to estimate the homography matrix and evaluate the pose information of heliport with respect to the camera.

The article is organized as follows: The second section discusses the related works. Then, the explanation of visual algorithm which allows to find the helipad marks is presented in "Computer vision algorithm" section, whereas the description of experiments and their outcomes are reported in "Results" section. Final remarks and future works are in "Conclusions and further works" section.

Related works

In literature, there are many methods based on CV for helipad detection and pose estimation of UAVs.[9] In Prakash and Saravanan,[10] an algorithm which employs the scale invariant robust features was implemented to detect the features of character H. A preloaded template image of the helipad is matched with the aerial images by using these invariant features. The method also deals with blurred images that are typically due to vibrations of UAV or external weather conditions. The authors claimed that the average helipad detection time was found equal to about 300 ms.

Another algorithm able to recognize the international standard helipad was proposed by Lee et al.[11] By computing the edge distribution function of the circle and the inner H, it is possible to extract a signature useful to locate the visual marks in the aerial image.

A shape recognition approach based on geometric features such as the area, perimeter, and invariant Hu moments was proposed in Saripalli et al.[12–14] to detect the visual target. Such features are commonly employed in task of optical character recognition because of their invariance to translation, rotation, and scaling. A similar methodology was presented in Sanchez-Lopez et al.[15] where the principal component analysis (PCA) applied to the first seven Hu moments is performed for each blob contained in the image. Afterward, an artificial neural network takes as input the information obtained from the PCA processing and provides the classification of blobs. The signature of H is used for extracting its corners and then computing the helipad pose. Further analyses and experimental validations were presented by Sanchez-Lopez et al.[16]

Silva et al.[17] proposed a global search procedure able to detect the best vantage points in the environment where is more likely to find the heliport. Then, for each vantage point, the UAV moves toward it and a vision-based local search of the helipad is performed. The local method aims at ranking all the objects of the scene according to their likelihood to be the heliport. Such likelihood is learnt during the takeoff phase and it is based on the saliency regional descriptor related to helipad marks.

Other two methods which rely on machine learning were presented by de Oliveira et al.[18] in order to automatically detect two types of landing pads. These pads consist of the traditional characters "H" and "X" inside circles. Six classifiers based on Haar-like features and local binary patterns (LBPs) were trained in order to recognize the artificial helipad marks. The outcomes proved that the Haar-based method was more accurate than the LBP-based one, although the latter is faster than the former.

In the study by Wu and Tsai,[19] an omni-vision–based localization algorithm was presented to support the drone landing on standard helipad. The method assesses the proper geometric features of points, circles, and lines of heliport marks in order to estimate pose of helicopter with respect to the landing site. Specifically, the method owns a first stage aimed at approaching the heliport and a second one that constrains the drone to fly between the outmost boundaries of the H mark and docking on its center. The computed relative distance errors were below 5% and the orientation errors were smaller than 2.5 considering a distance range of 1.20 m.

In the study by Premachandra et al.,[20] another method for searching the heliport was described. The data from the camera and an ultrasonic sensor were processed by an onboard Raspberry Pi unit. The altitude of the drone was computed by means of the ultrasonic sensor, whereas the "H" mark was recognized exploiting the Hough transform framework.

A monocular vision–based real-time target recognition was proposed by Lin et al.[21] as well. The helipad contours are used to perform the point feature mapping and clustering. The algorithm is able to recognize the international landing target in a cluttered environment providing the 4 degrees of freedom pose of drone with respect to pad.

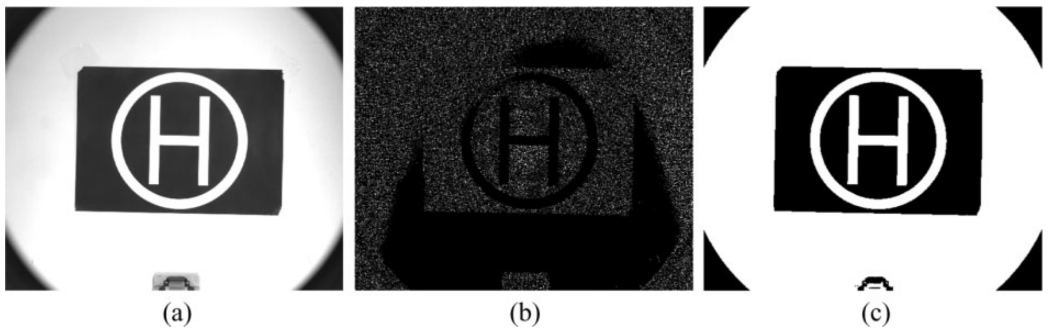


Figure 1. (a) Output image obtained after applying the Wiener filter, (b) noise detected in the image during the application of median filtering, and (c) resulting binary image after the thresholding step.

Proposed approach

The algorithm presented in this work falls in the category of papers which detect autonomously the helipad target and estimate its pose with respect to the vision camera. We propose a novel approach to detect the helipad in order to achieve high accuracies of pose measurements meeting simultaneously the strict processing time constraints of a real-time application. In fact, the analysis of the related literature reveals that many methods require high computation resources that are not always available onboard the drone. On the contrary, faster and simpler algorithms do not continuously ensure enough robustness in the searching step of heliport marks.

Hence, we devised a quicker and robust image processing pipeline able to return the accurate pose measurements, ensuring continuous aids to the drone during its landing over the platform. The analysis of the local curvatures enables to detect precisely the locations of H corners, then geometrical considerations on the distance among couple of corner centroids allow the detection of the main direction of the helipad. The algorithm is quite simple and avoids the computational costs of the standard approaches based on feature extraction, template matching, line fitting, and so on.

CV algorithm

As before mentioned, the aim of the algorithm is to provide the pose of the target platform with respect to the camera observing the pattern. Thus, the pose estimation of UAV on which the camera is attached is directly derived.

The target used in our experiment was the standard international helipad having the character H inside circles as illustrated in Figure 1.

The main steps concerning the helipad detection and pose estimation can be listed as follows:

1. image acquisition and preprocessing,
2. helipad mark extraction,
3. corner detection of helipad,
4. corner matching, and
5. pose estimation.

The details of each step are explained in the following paragraphs.

Image acquisition and preprocessing

It is a typical and essential step to preprocess the images in order to get an improvement of their quality. This preliminary stage aims at reducing the undesirable noise effects and highlighting the particulars of image. Therefore, a cascade filtering is applied to images before extracting the information useful in the other steps.

The acquired image represented in the red–green–blue color space is converted in grayscale in order to focus only on the luminance information. As a matter of fact, in many applications of CV the hue and saturation information are not quite necessary, whereas the luminance is more suitable in distinguishing visual features or extracting edges. Since the captured aerial images might be affected by blurring due to abrupt vibrations of UAV or to external weather conditions, a Wiener filtering[22] is performed as well. In Figure 1(a), the result obtained by applying this filter is shown. Afterward a median filter[23] is used to achieve the noise reduction on the image. This filtering is widely used because of its important capability in preserving edges and removing the unavoidable noise (refer to Figure 1(b)). Finally, the resulting image obtained from the previous steps is segmented by thresholding (see Figure 1(c)). By taking advantage of the high contrast in the helipad region between the dark background and the light foreground, we are able to extract the binary image by employing an adaptive thresholding[24] on the intensity values. In this way, strong illumination gradient in the image is taken into account and a more robust segmentation is achieved.

Helipad mark extraction

Once the binary image is computed, we detect all the connected components (i.e. blobs) inside the image under

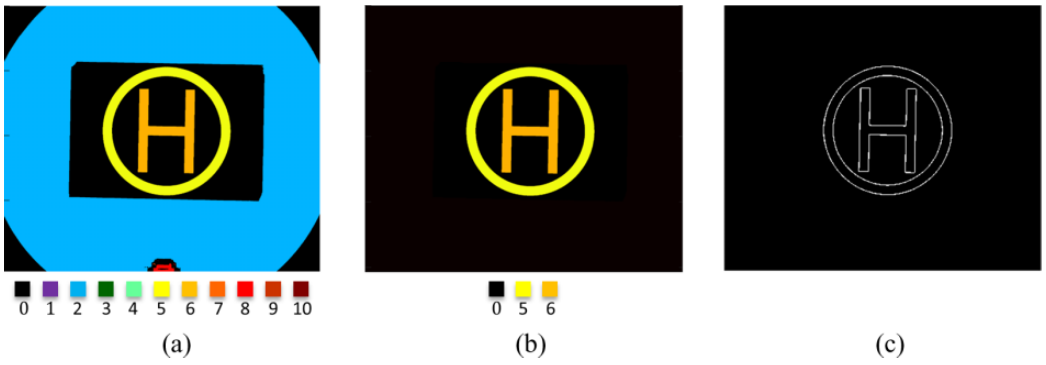


Figure 2. (a) All the candidates before the cascade processing labeled using different colors. Ten different blobs are identified for the image under investigation. Some of these blobs are not quite visible because their areas are very small (few pixels). (b) Final helipad marks represented by the two remaining blobs after the detection procedure and (c) contours of helipad.

investigation (refer to Figure 2(a)). By exploiting some intrinsic properties of each blob such as the location of its center of mass, the Euler number, the eccentricity, the perimeter, and the area, we are able to identify the blobs which represent the helipad marks, namely the character H and the circumscribing circles.

Among all the extracted blobs, the ones having a small area value in terms of pixel number are discarded from the successive processing. A second level of analysis is performed by evaluating the corresponding Euler number of the remaining candidates. Specifically, the Euler number is an integer value defined as number of connected components minus the whole numbers. Consequently, this value should be equal to zero for circle blobs and one for H blobs. Therefore, blobs having a different Euler number are discarded. Moreover, blobs having Euler number equal to zero but eccentricity much larger than zero are rejected as well. In fact, eccentricity equal to zero indicates that the blob is a circle, whereas eccentricity equal to one stands for a line segment. All the middle eccentricity values indicate possible ellipses.

The next level of classification takes advantage of the knowledge that the blob H has to be inside the circle blob. As a consequence, their corresponding centroids should be near, namely their Euclidean distance has to be small (ideally zero).

The last classification level checks the ratios between the areas and perimeters of blobs. In this regard, the actual ratios computed by considering the circle and the H blobs should keep their expected values, namely they should be invariant with respect to rotations and scale changes. The equation (1) defines mathematically the above conditions.

$$\left|\frac{P_{H_a}[px]}{P_{\mathrm{CIRCLE}_a}[px]} - \frac{P_{H_e}[m]}{P_{\mathrm{CIRCLE}_e}[m]}\right| \le \delta$$
$$\left|\frac{A_{H_a}[px^2]}{A_{\mathrm{CIRCLE}_a}[px^2]} - \frac{A_{H_e}[m^2]}{A_{\mathrm{CIRCLE}_e}[m^2]}\right| \le \delta \qquad (1)$$

The capital letters P and A stand for the perimeters and the areas of blobs, whereas the subscripts a and e point out the actual and the expected values, respectively. The value δ is set to 0.05.

The main steps of helipad mark detection are summarized in algorithm 1.

Algorithm 1. Helipad mark extraction.

```
Input: Blobs, thsArea, thsEcc, thsEuclideanDist, RatioP, RatioA, $\delta$.
Output: BlobHelipad.
Classification step I:
1.  Foreach item i of Blobs do
2.      If Blobs[i].Area < thsArea then discard blob i from Blobs;
3.  End
Classification step II:
4.  Foreach item i of Blobs do
5.      If Blobs[i].EulerNumber = 1 then update BlobH;
6.      If Blobs[i].EulerNumber = 0 and Blobs[i].Eccentricity
          $\le$ thsEcc then update BlobCircle;
7.  End
Classification step III:
8.  Foreach item i of BlobH and item j of BlobCircle do
9.      If EuclideanDistance(BlobH[i].Centroid,
          BlobCircle[j].Centroid) $\le$ thsEuclideanDist then
    update
          BlobH, BlobCircle;
10. End
Classification step IV:
11. Foreach item i of BlobH and item j of BlobCircle do
12. If abs(BlobH[i].Perimeter/BlobCircle[j].Perimeter – RatioP)
        $\le \delta$ and abs(BlobH[i].Area/BlobCircle[j].Area – RatioA)
        $\le \delta$ then update BlobHelipad;
13. End
14. Return BlobHelipad;
```

At the end of detection procedure, we can identify which blobs among all the candidates are the helipad marks. By looking at Figure 2(a), 10 different blobs can be detected. The black color indicates the image background. After the first classification step, the blobs 1, 3, 4, 7, 9, and 10 are

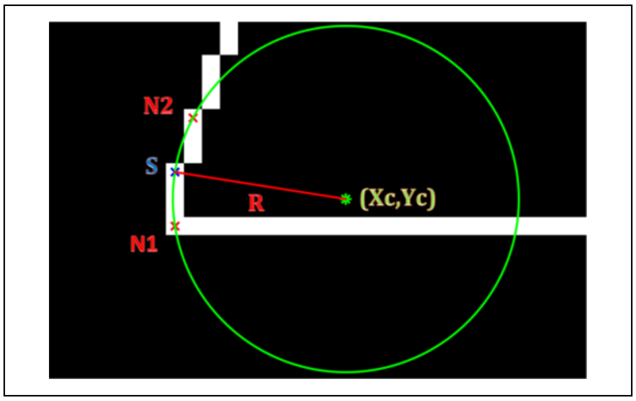


Figure 3. The seed pixel is represented by the blue cross, whereas the neighboring pixels are the red crosses. The radius of the circle defines the value R related to seed point.

discarded from the computation. The second step filters out the blobs 2 and 8 as well. In fact, they own negative Euler numbers.

At this stage only two blobs remain, that is, the blobs 5 and 6. The third and fourth steps check if the Euclidean distance of their centroids and the ratios of related areas and perimeters are still met. The ultimate extracted blobs are shown in Figure 2(b).

Once the helipad marks have been identified, the Canny edge detector is performed in order to extract the contours (please refer to Figure 2(c)). Since the corner detection step aims at finding the 12 corners of H edge, we operate only on the inner contour. The two-dimension (2-D) image pixel coordinates belonging to edge are arranged in a sorted matrix which will be the input of the following computations.

Corner detection of helipad

The main purpose of this step is to find the H corners. Although there are many corner detector algorithms in the state of art,[25–27] we implemented a faster and more accurate method to achieve this goal. The corners could be detected at the intersection of the 12 straight lines of character H as well. In this regard, feature extraction operators as the Hough transform,[28] convolutional-based techniques having specific kernels,[22] and line following algorithms might be used to detect such straight lines. However, all these methods return many false positive detections as they heavily depend on the parameter settings and can be computationally expensive. The proposed approach tries to overcome the above limitations.

The method is based on the radius-of-curvature computation for each pixel belonging to the helipad contour. In details, let S be the pixel seed under investigation and N_1 and N_2 be the neighboring pixels to S (located at an integer distance of n pixels along the edge), we compute the circle passing through these three points as shown in Figure 3. It is worth underlining that the distance value n between the seed point and the neighboring points is settled according to contour length. The smaller the edge length, the smaller the distance value and vice versa. In this way, different edge lengths of H (due to distance variations between the camera and the heliport) does not affect the corner detection process.

At the end of this step, a radius of curvature R will be associated with every 2-D point of H edge. By evaluating the array of radii, it is possible to efficiently detect the corners of interest. A big radius value denotes that the point is far from a corner. On the contrary, a small value indicates that the point might be a possible candidate to be a corner.

In summary, the smallest values of curvature array stand for all the possible corner candidates.

However, the false positive ones could be detected by the algorithm. As a consequence, we identify the possible outliers by taking advantage of knowledge of H size and exploiting the Euclidean distances between these points and the centroid of H contour.

By looking at Figure 4(a), three expected distance values can be derived from the geometry of H-shaped edge. In details, the Euclidean distances of the four outer corners from the center O_H are equal to D_o, whereas the four inner corners have an Euclidean distance of D_i. Finally, the remaining four middle corners own distance values equal to D_m.

At this stage, three checks are performed. Specifically, the Euclidean distance of a corner candidate C from the centroid O_G of the edge under investigation is compared with the three expected ones by employing equation (2), where $[px]$ stands for pixel units.

$$\begin{aligned} \left|\overline{CO_G}[px] - D_i[px]\right| &\leq \varepsilon \\ \left|\overline{CO_G}[px] - D_m[px]\right| &\leq \varepsilon \\ \left|\overline{CO_G}[px] - D_o[px]\right| &\leq \varepsilon \end{aligned} \tag{2}$$

The expected adaptive distances are derived by estimating the value x in Figure 4(a). In this regard, by knowing the contour length E_s (expressed in pixel units) and the geometry of H, the value x (in pixels) is computed accordingly ($E_s[px] = 46x$). At this stage, simple mathematical relations enable to calculate the expected values D_i, D_m, and D_o, which are equal to $\sqrt{4.25}x$, $\sqrt{24.25}x$, and $\sqrt{29.25}x$, respectively.

In this way, in the event that a candidate pixel has a distance value much different with respect to these three adaptive parameters, then it is labeled as outlier, and thus it is discarded from the computation. Conversely, if one of the three checks is satisfied, the candidate is labeled as inlier. The value ε represents a small value which depends once more on the total edge length. However, it is worth highlighting that such distance comparisons are valid if the perspective deformations are not too accentuated.

At this point, all the inlier candidates can be detected by performing the Euclidean distance checks. In case more inlier candidates are found in proximity of an actual corner

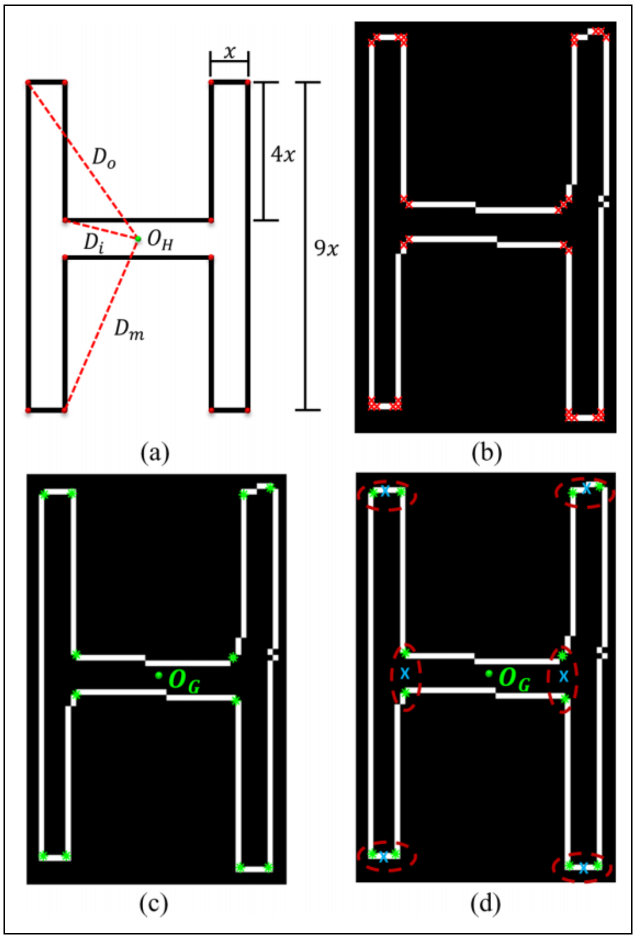


Figure 4. (a) Sketch of the actual size of H together with the three expected Euclidean distances referred to one inner corner, one middle corner, and one outer corner. (b) Candidate corners identified by using the radius-of-curvature method and the Euclidean distance checks. (c) Twelve centroids of the actual corners and (d) six centroids (blue crosses) of the clustered corners (dark red ellipses).

(refer to Figure 4(b)), we provide their centroids. At the end of the process, 12 corner locations are returned as observable in Figure 4(c). Once the 12 corners have been detected, we group them in pairs by using the hierarchical clustering thus identifying six centroids (see Figure 4(d)). This step is mandatory to accomplish the following steps.

Corner matching

The six centroids obtained in the previous stage are useful to estimate the principal directions of H mark. This information is exploited to correctly match the corners in the image with their corresponding ones in the three-dimension (3-D) reference system. In this way, the heliport pose can be estimated. All the eligible straight lines passing through a pair of 2-D points taken among the six centroids are computed. As a consequence, 15 lines will be detected since each pair of points must be considered once (see Figure 5(a)). Subsequently for each obtained line, we compute the four distances of other remaining centroids with respect to itself. As an example, by looking at Figure 5(b), the red segment on the left side of H stands for the straight line under investigation, whereas the distances of other centroids are represented with green dotted lines. At the end of the step, 60 distance measurements are collected. Therefore, the smallest distance D_{MIN} among all the computed ones enables to identify the line going through the main direction of the H (the red line in Figure 5(b). Hence, the vertical direction of H is defined as the slope of such line. Conventionally, we define the horizontal direction as the perpendicular to the vertical one. In Figure 5(c), the green arrow represents the vertical direction of H and the red one stands for the horizontal direction.

The knowledge of principal directions enables to split-up the H mark in four quadrants as shown in Figure 5(d). Therefore, rotations of H in the image do not affect the corner matching step since we are always able to identify the quadrants. The center O_G of new reference system is placed at the centroid of H edge. If the perspective is not too accentuated, three corners for each quadrant can be distinguished. By computing the Euclidean distances between such corners and the centroid of H, it is possible to associate them to the related 3-D corners. As an example, the three 2-D corners of first quadrant Q1 in Figure 5(d) are matched with the corresponding 3-D ones in Figure 5(e) by evaluating the Euclidean distances. Since the corner 1 has the largest Euclidean distance from the center O_G, it means that it represents the outer corner of Q1. On the contrary the corner 3, which has the least Euclidean distance value, stands for the inner corner of Q1. Consequently, the corner 2 represents the middle corner since it has an average distance value.

In case that a quadrant does not have inside three corners, we discard it from the computation because we are not able to properly accomplish the matching. However, we need at least four correspondences to attain the pose estimation.

Pose estimation

The last step of the method has to provide the 3-D pose of the heliport with respect to camera. At this stage, the 2-D corners have to be corrected from distortion effects due to camera lens curvature. Therefore, a preliminary calibration step aimed at extracting the intrinsic camera parameters is mandatory in order to ensure accurate measurements.[29–31]

The 2-D and 3-D correspondences are related to each other as follows

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|t]P = \begin{bmatrix} f_x & \alpha & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3}$$

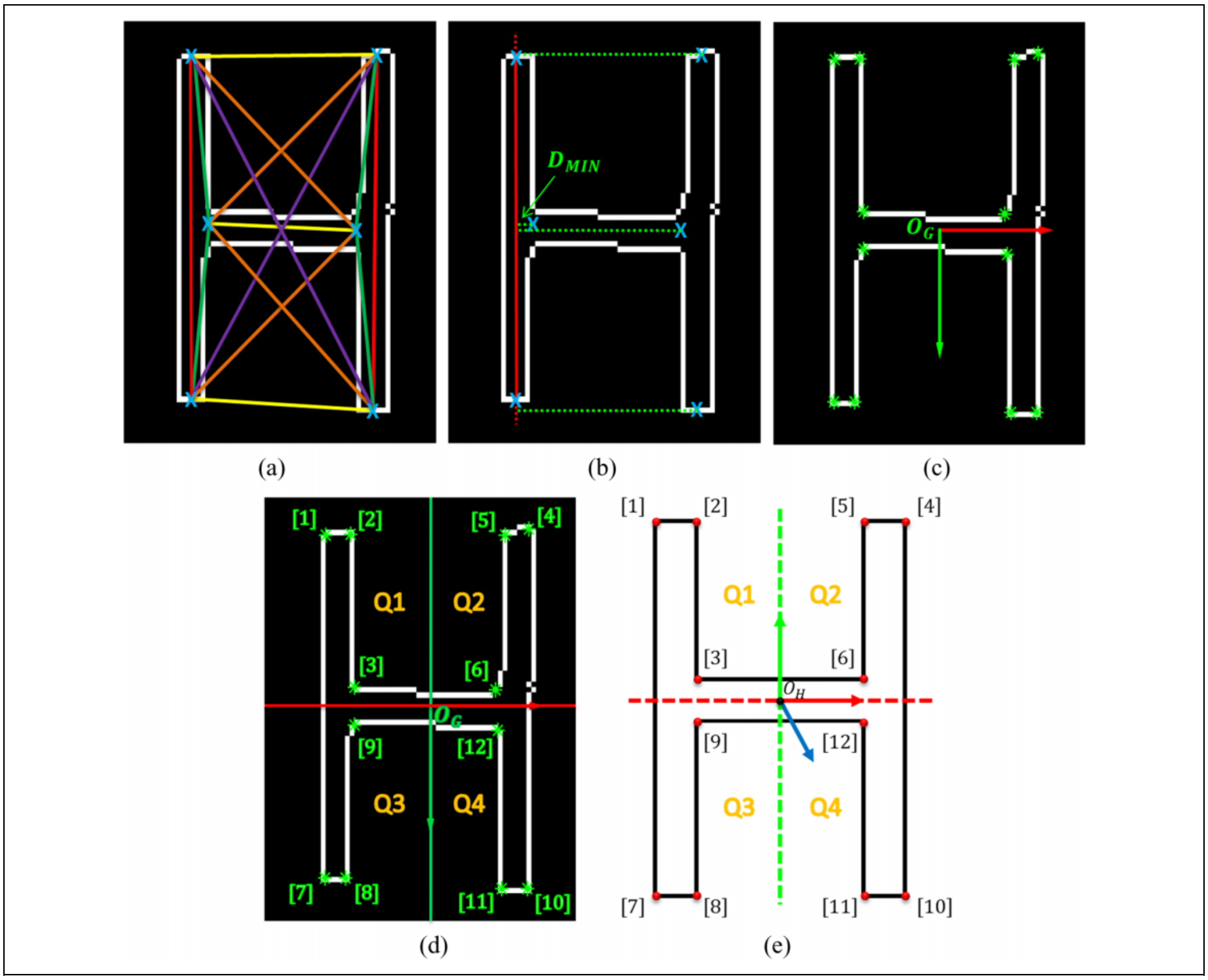


Figure 5. (a) Computing of all possible straight lines passing through the six centroids, (b) detection of vertical direction of H mark, and (c) obtained principal directions. (d) Partitions of H in four quadrants and (e) corresponding partition of actual target with the corner associations.

By looking at equation (3), (u, v) are the 2-D corner coordinates expressed in pixels, K is the intrinsic camera matrix, $[R|t]$ represents the extrinsic matrix, and P contains the 3-D homogeneous metric coordinates of corresponding corners. Since our aim is to estimate the extrinsic matrix, we need at least four correspondences to compute it. By exploiting the direct linear transformation method,[32] we can estimate the homography matrix between the 2-D and the corresponding 3-D points. Thus, the rotation matrix R and the translation vector t can be calculated accordingly. Figure 6 reports an example of the computed pose by our algorithm.

Results

The effectiveness and accuracy of proposed vision algorithm have been validated through controlled indoor and outdoor tests performed at our institute. "Indoor experiments" section describes the indoor setup together with the discussion about the related experiments. "Outdoor experiments" section introduces the outdoor tests and its outcomes, whereas "Time performance" section discusses the performance of our method in terms of processing time.

Indoor experiments

The experimental setup for indoor tests is shown in Figure 7. The used camera is the UI-2280SE-M-GL produced by Imaging Development Systems (IDS)[33] having a resolution of 2448×2048 pixels. It has been fixed onto a rotational stage in order to provide angle variations along the Z_{CAM} axis. The panel containing the heliport mark has been attached to another rotational stage used to provide rotations along the X_H and Y_H axes (see Figure 6(b)). Finally, a translational stage has been employed to vary the distance between the camera and the landing target. In this way, we can simulate some of UAV movements during the flight.

The first validation has been performed on the analysis of single images, evaluating the relative errors concerning the

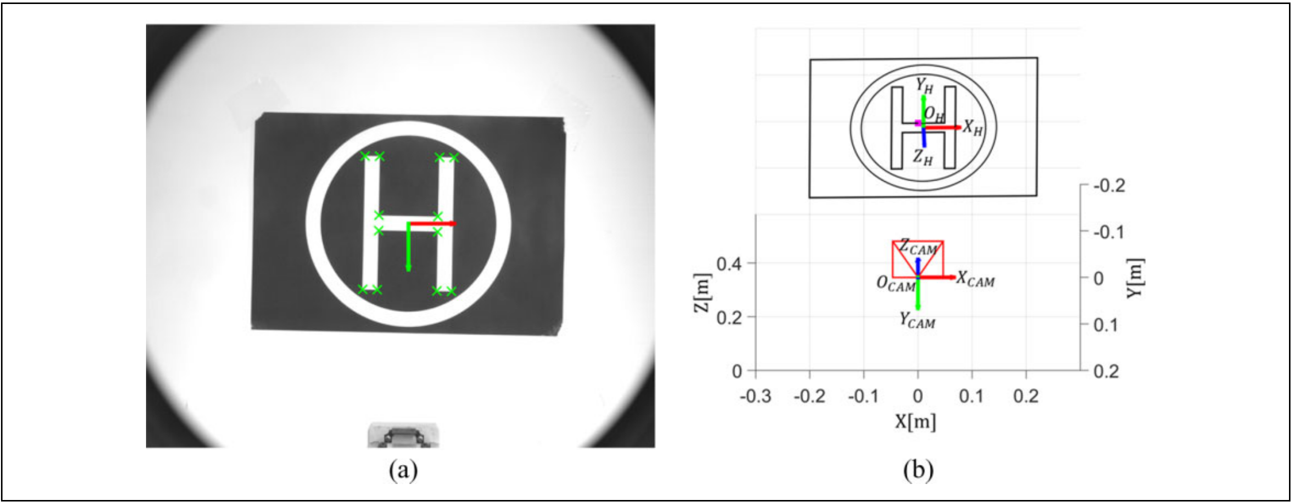


Figure 6. (a) Example of output image after the algorithm and (b) 3-D view of heliport with respect to camera reference system.

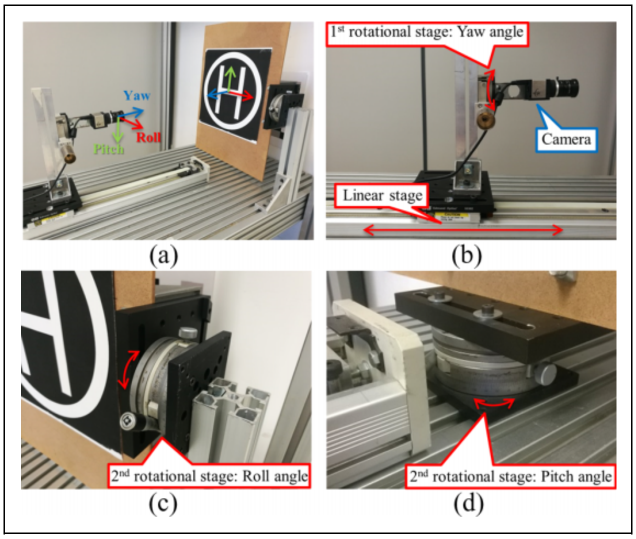

Yaw
Roll
Pitch
(a)
1st rotational stage: Yaw angle
Camera
Linear stage
(b)
2nd rotational stage: Roll angle
(c)
2nd rotational stage: Pitch angle
(d)

Figure 7. (a) Overview of acquisition setup for the indoor tests, (b) enlargement of translational and rotational stages along with the camera, and (c, d) enlargement of the second rotational stage used to provide both pitch and roll angle variations.

distance estimation of heliport. Specifically, the camera has been moved away from the visual target by means of the linear support by steps of 0.10 m. The other auxiliary stages have been locked in order to avoid other unwanted movements. In details, the panel has been kept parallel to image plane, that is, the roll, pitch, and yaw angles have been held to 180°, 0°, and 0°, respectively (refer to Figure 7(a)).

The distance measurements D have been derived as the Euclidean distance among the two reference system origins by using equation (4)

$$D = \overline{O_H O_{\mathrm{CAM}}} = \sqrt{(X_H - X_{\mathrm{CAM}})^2 + (Y_H - Y_{\mathrm{CAM}})^2 + (Z_H - Z_{\mathrm{CAM}})^2} \tag{4}$$

The method has been evaluated in short and long distance measurements. Specifically, Table 1(a) reports the relative errors by taking into account short distances in the range [0.57–1.67]m, whereas Table 1(b) assesses higher distances ranging in [1.51–3.01]m.

All the relative errors have been derived by means of equation (5) as following reported

$$\varepsilon_r = \frac{|a_v - e_v|}{e_v} 100 \tag{5}$$

where a_v stands for the actual value returned from the processing algorithm and e_v is the expected value, namely the information received by the ground truth.

The maximum relative error observable in Table 1(a) corresponds to 0.72%, that is, a maximum error of about 0.012 m has been done by considering short distances. The absolute error between the total expected displacement of 1.10 m and the actual one of 1.115 m is found equal to 0.015 m. On the contrary, by looking at Table 1(b), one can notice that the highest relative error is 0.82%, corresponding to an error of 0.021 m. However, an absolute error of 0.014 m is obtained by considering the entire actual displacement of 1.514 m and the expected one of 1.50 m. The computed absolute errors seem to be almost equal in both tests concerning short and long distances, although the relative errors slightly increase their values in long distance tests. However, this experiment has proven that the relative errors are always below 1%. Therefore, our algorithm is able to perform well in estimating distances as further proven by the low root mean squared error (RMSE) values.

Other similar tests have been carried out to assess the vision system in estimating angle variations as well. The validation is obtained by imposing controlled rotations to both the panel and the camera. More specifically, the rotations have been performed separately for each axis by keeping a distance of about 0.50 m. In this way, we are able to evaluate how the method works for the single rotational movements.

Table 1. Relative errors in distance estimations by considering (a) short distance range and (b) long distance range between the camera and the helipad target.[a]

Nominal distances D (m)	Actual distances D (m)	Relative errors ε_r (%)
(a)		
0.57	0.5673	0.48
0.67	0.6683	0.26
0.77	0.7653	0.61
0.87	0.8695	0.06
0.97	0.9718	0.19
1.07	1.0723	0.22
1.17	1.1736	0.31
1.27	1.2727	0.21
1.37	1.3733	0.24
1.47	1.4766	0.45
1.57	1.5778	0.50
1.67	1.6821	**0.72**
	RMSE (m): 0.005	
(b)		
1.51	1.5063	0.24
1.61	1.6019	0.50
1.71	1.7105	0.03
1.81	1.8095	0.03
1.91	1.9137	0.19
2.01	2.0226	0.63
2.11	2.1070	0.14
2.21	2.2184	0.38
2.31	2.3128	0.12
2.41	2.4185	0.35
2.51	2.5306	**0.82**
2.61	2.6271	0.65
2.71	2.7248	0.55
2.81	2.8265	0.59
2.91	2.9312	0.73
3.01	3.0204	0.35
	RMSE (m): 0.012	

RMSE: root mean squared error.
[a]A displacement step of 0.10 m has been considered in both tests. The relative RMSEs, obtained by considering the nominal and the actual values, are reported as well. The values in bold represent the highest relative errors related to each table.

Table 2. Relative errors in rotation estimations between the camera and the helipad target evaluated along (a) the *X*-axis, (b) the *Y*-axis, and (c) the *Z*-axis along with their RMSE values.

Nominal angles roll (°)	Actual angles roll (°)	Relative errors ε_r (%)
(a)		
148.00	148.0311	0.02
153.00	152.9814	0.01
158.00	157.5740	0.27
163.00	162.7612	0.15
168.00	167.1835	**0.49**
173.00	172.6792	0.19
178.00	178.1026	0.06
183.00	182.9871	0.01
188.00	188.5639	0.30
193.00	193.7862	0.41
198.00	198.4732	0.24
203.00	203.5272	0.26
208.00	208.5394	0.26
	RMSE (°): 0.459	
(b)		
−30.50	−30.4319	0.22
−25.50	−26.0256	2.06
−20.50	−20.8997	1.95
−15.50	−15.7111	1.36
−10.50	−10.3334	1.59
−5.50	−5.6491	**2.71**
−0.50	−0.4897	2.06
4.50	4.5859	1.91
9.50	9.5498	0.53
14.50	14.4338	0.46
19.50	18.9765	2.69
24.50	24.0708	1.75
29.50	29.0477	1.53
	RMSE (°): 0.305	
(c)		
−30.90	−30.8658	0.11
−25.90	−25.8811	0.07
−20.90	−20.8613	0.19
−15.90	−15.8904	0.06
−10.90	−10.9106	0.10
−5.900	−5.9535	0.91
−0.90	−0.8964	0.40
4.10	4.0615	**0.94**
9.10	9.1056	0.06
14.10	14.0974	0.02
19.10	19.1034	0.02
24.10	24.0929	0.03
29.10	29.0907	0.03
	RMSE (°): 0.024	

RMSE: root mean squared error.

As shown in Table 2, relative errors concerning the *roll* and *yaw* variations are below 1%. On the contrary, the analysis of *pitch* rotations shows relative errors limited to about 3%. Hence, the errors in Table 2(b) are slightly higher than other ones. This behavior might be due to the accuracy of corner extraction from H mark. In fact, as already described in the previous section, some of corner candidates could be located near the actual corner. Therefore, the method merges these points providing only their centroid. As a consequence, inaccuracies in the estimation of corner locations might slightly affect the actual pose estimation.

Nevertheless, both the small relative errors and RMSE values obtained in the experiments enable to affirm that our algorithm is able to achieve high accuracies in estimating the target pose. In this regard, this method performs better than the one proposed in Lin et al.[21] in terms of accuracy. More specifically, the average RMSE values found in the other work were 0.029 m and 1.786° for the distance and the angle estimations, respectively. Instead our average RMSE values are equal to 0.0085 m and 0.263°. Hence, one can highlight how our methodology returns more accurate measurements.

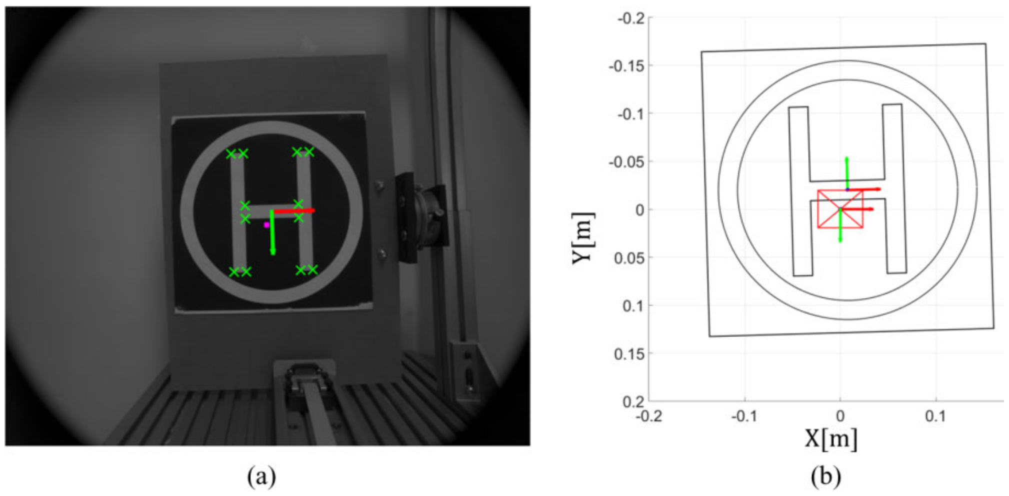


Figure 8. Output image after the processing and (b) 3-D view perspective of target with respect to camera reference system. In this case, the working distance was about twice the heliport size.

In order to compare our outcomes with the study by Sanchez-Lopez et al.,[16] further experiments have been carried out to estimate the reliability of algorithm in the event that stationary frames were acquired. Specifically, we considered two different orientations of target with respect to the camera: In the former test, we forced the panel and the camera plane to be parallel, whereas in the latter, we considerably tilted the target plane. In both the tests, we acquired a sequence of 300 frames corresponding to an acquisition time of about 70 s. Furthermore, different working distances between the camera and the heliport have been taken into account. As the heliport size is 29 cm, the distances have been chosen by considering approximately multiples of such size.

Figure 8 shows a test with the helipad parallel to the acquisition system, whereas in Table 3 some statistics related to the processing are reported.

Table 3(a) reports the average values of the 3-D positions (X_H, Y_H, Z_H) of the heliport origin O_H and its orientations (*roll*, *pitch*, *yaw*) with respect to the camera reference system at different distances (multiples of helipad size). Instead Table 3(b) contains the standard deviation (SD) values.

One can observe that the SDs concerning the helipad position are significantly small while the Euler angles seem to be noisier. In this regard, the roll and pitch angles show the larger variations. Nevertheless, these fluctuations increase when the distance is bigger, that is, when the drone is far from the heliport.

By comparing this work with the work of Sanchez-Lopez et al.,[16] it is possible to note how the proposed vision algorithm is able to provide more accurate information of the heliport pose. In fact, by considering a distance of about six times the heliport size (see gray rows in the Table 3(b)), our SDs are one order of magnitude smaller than the other work. The helipad size in the study by Sanchez-Lopez et al.[16] is about 20 cm.

In the second experiment, we tilted the target plane as shown in Figure 9. In this case, we can observe that the noise of Euler angles (see Table 4(b)) is noticeably reduced in comparison with the previous test (refer to Table 3(b)) by considering all the distance values. Hence, we can state that the vision algorithm provides more accurate estimates in the event that the helipad is tilted with respect to the camera, that is, when the drone might be in a dangerous pose.

In the same way, we compare the results of Sanchez-Lopez et al.[16] with ours. In this case, a distance of about three times the heliport size is considered for accomplishing the comparison. By looking at the gray rows in Table 4(b), we can note how our values are considerably smaller proving once more the reliability of method in returning the pose of camera with respect to the visual target.

Outdoor experiments

The efficiency of our methodology has been proven by performing outdoor tests as well. In fact, abrupt light changings or strong sunlight reflections might negatively affect the algorithm when it is working. Therefore, the proposed approach has been evaluated in presence of critical situations as well.

In Figure 10(a), the outdoor setup is shown. In this case, two rotational stages and one linear support have been used to provide controlled movements. Therefore, by following similar procedures to the indoor experiments, both distance and angle variations have been run.

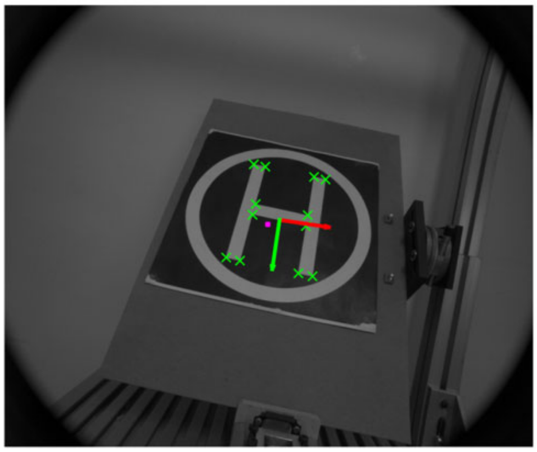
The nominal distance between the camera and the pad has been given by means of a dot laser range finder[34] having an operating range from 0.1 to 10 m and a precision of 1 mm.

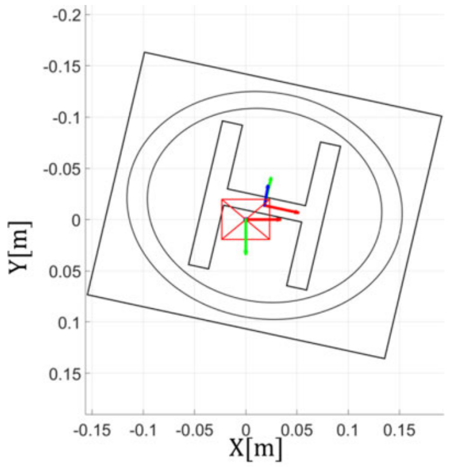
The Figure 10(b) shows some random outdoor images where the reflections, shadows, and light changings are more evident. Nevertheless, the algorithm is still able to provide the heliport pose proving once more its robustness

Table 3. (a) Mean values by taking into account approximated multiple distances of the heliport size (29 cm) and (b) SDs.[a]

Method	Distance (m)	X_H (m)	Y_H (m)	Z_H (m)	Roll (°)	Pitch (°)	Yaw (°)
(a) Average values							
Our	2*H_size	0.0074679	−0.0197432	0.5523964	179.2797522	0.2783291	−1.5853760
	3*H_size	0.0083024	−0.0076520	0.8458917	179.1623058	0.2722564	−1.6052125
	4*H_size	0.0094729	0.0047161	1.1371761	179.5363696	0.5484095	−1.5870120
	5*H_size	0.0104308	0.0166827	1.4292777	179.2149204	0.3204720	−1.6345813
	6*H_size	0.0112035	0.0285918	1.7215680	179.8616853	0.7023867	−1.6236592
	7*H_size	0.0130016	0.0397918	2.0022781	178.3100760	0.8254675	−1.5099034
Sanchez-Lopez et al.[16]	6*H_size	0.0080000	−0.0229000	1.2341500	108.6000000	162.9000000	161.6000000
(b) SD values							
Our	2*H_size	0.0000182	0.0000293	0.0001056	0.1078467	0.1477211	0.0087885
	3*H_size	0.0000364	0.0000569	0.0002694	0.3072550	0.2766787	0.0159103
	4*H_size	0.0000800	0.0000901	0.0005645	0.5363502	1.2478151	0.0425658
	5*H_size	0.0000909	0.0000943	0.0009730	0.6034604	1.3977914	0.0636385
	6*H_size	0.0000773	0.0000741	0.0012362	0.8633100	1.6640289	0.0552946
	7*H_size	0.0001352	0.0001099	0.0026254	1.066469	1.9611922	0.1482295
Sanchez-Lopez et al.[16]	6*H_size	0.0004600	0.0004250	0.0069000	12.4700000	3.9000000	11.6000000

SD: standard deviation.
[a]Each row reports the information related to the processing of 300 stationary frames. The values in gray rows are used to compare our data with the study by Sanchez-Lopez et al.,[16] having an helipad size of about 20 cm.


(a)



(b)

Figure 9. (a) Resulting image after the processing steps and (b) 3-D view of helipad. Here, the working distance is twice the heliport size.

and reliability against to external noise. Figure 10(c) reports some of the data set images employed for the controlled experiments.

In the first controlled test, the distance between the camera and the helipad has been varied by steps of 0.05 m. The other two rotational stages have been held thus enabling the analysis of the only distance movements.

Table 5(a) reports the obtained distance estimations. The relative errors have been computed by exploiting once more the equation (5). By comparing these results with those of Table 1(a, b), one can note that the relative errors are quite similar. In fact, they are still below 1% and a maximum relative error of 0.98% has been found. Therefore, the vision algorithm well performs in the distance estimation in both indoor and outdoor environments.

Slight increases of relative errors have been observed by evaluating the rotational movements. Specifically, rotations by steps of 5° have been taken into account as for the indoor experiments.

The relative errors of roll angles reported in Table 5(b) are very comparable with the one of Table 2(a). In this case, the obtained maximum error is 0.43%, which is almost negligible.

The error analysis of other two angles, that is, the pitch and yaw ones in Table 5(c, d), shows increased values with respect to those of indoor test (see Table 2(b, c)). In this regard, the maximum relative errors related to pitch angle for the indoor and outdoor experiments are 2.71% and 3.98%, respectively. Hence, a maximum error increase of about 1.3% is observable by analyzing the outdoor data.

Table 4. (a) Mean values and (b) SD values considering the helipad tilted.[a]

Method	Distance (m)	X_H (m)	Y_H (m)	Z_H (m)	Roll (°)	Pitch (°)	Yaw (°)
(a) Average values							
Our	2*H_size	0.0183273	−0.0137429	0.5485327	144.5934137	1.6274113	12.1365765
	3*H_size	0.0188922	−0.0026965	0.8400614	144.3700285	1.5151754	12.1951092
	4*H_size	0.0194399	0.0081917	1.1311983	144.1344572	1.6140310	12.1023604
	5*H_size	0.0202979	0.0192398	1.4219835	143.9592868	1.5474730	12.1899171
	6*H_size	0.0212786	0.0300728	1.7154282	143.6158734	1.4817518	12.1923212
	7*H_size	0.0217568	0.0406070	2.0085776	143.7131633	1.5981399	12.0238418
Sanchez-Lopez et al.[16]	3*H_size	−0.0433000	−0.0142000	0.6857000	72.8000000	133.0000000	7.9000000
(b) SD values							
Our	2*H_size	0.0000295	0.0000170	0.0002347	0.0560457	0.0543871	0.0337323
	3*H_size	0.0000417	0.0000240	0.0007009	0.0716890	0.0675144	0.0402707
	4*H_size	0.0000554	0.0000465	0.0009176	0.0908557	0.1119812	0.0770848
	5*H_size	0.0000928	0.0000489	0.0019848	0.1139258	0.0973499	0.0608155
	6*H_size	0.0000779	0.0000875	0.0025864	0.1434045	0.1239260	0.0758914
	7*H_size	0.0000666	0.0000773	0.0025811	0.1355283	0.1006775	0.0612105
Sanchez-Lopez et al.[16]	3*H_size	0.0004250	0.0005260	0.0016000	1.2200000	0.3800000	0.9800000

SD: standard deviation.
[a]A sequence of 300 stationary frames has been analyzed.

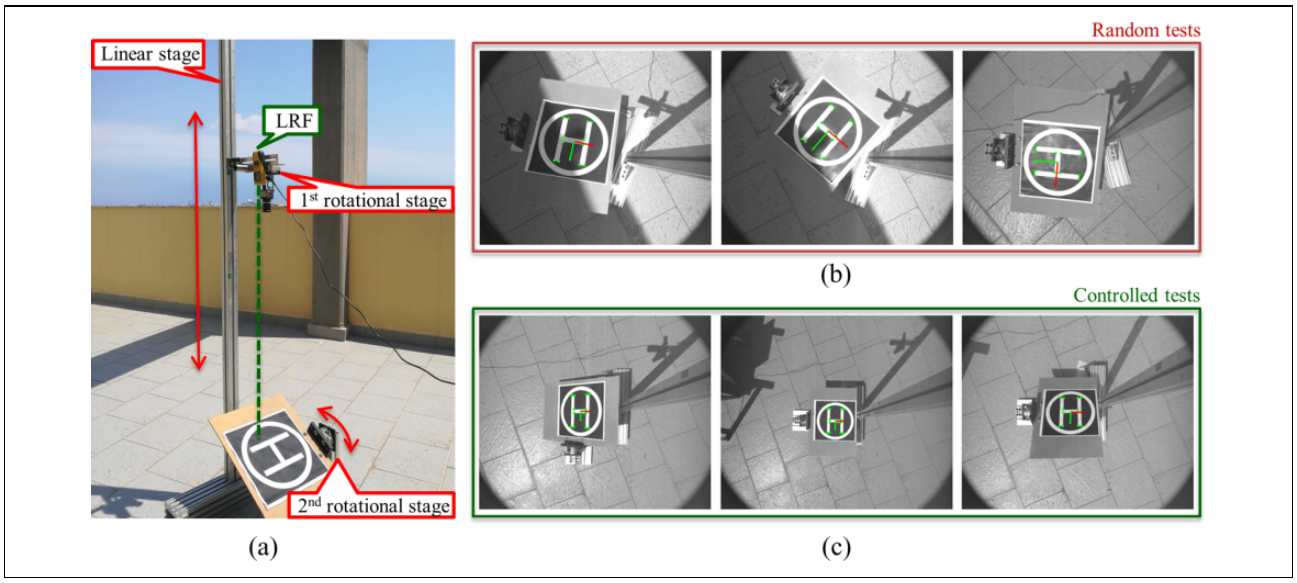


Figure 10. (a) Acquisition setup used in the outdoor experiments, (b) some of the example images used for testing the algorithm in critical situations, and (c) some of data set images employed in the accuracy assessing.

Similarly, a maximum increase of relative errors of about 0.5% is found for the yaw angle when the indoor and outdoor data are compared. In fact, the maximum obtained errors are 0.94% and 1.45%, respectively.

The errors of outdoor tests are slightly higher than the indoor ones. This is likely due to the external disturbance factors (see unavoidable reflections, shadows, blurring) that might affect negatively the algorithm during the helipad mark detection and corner extraction.

The obtained outcomes enable to affirm again that the proposed approach is still accurate and reliable in the computing of pose estimation in outdoor contexts as well. In fact, the relative errors in the distance and angle estimation are bounded to 1% and 4%, respectively. Moreover, the low RMSE values prove once more the effectiveness of the presented algorithm.

Time performance

Finally, a discussion about the processing time is due since it has an important role in this framework. The method has to provide the information of target pose in few milliseconds. As an example, by considering a working frequency of drone control system of about 5 Hz, the vision method should supply the information of interest in less than 200 ms.

In this way, the pose of drone can be quickly estimated and the control algorithm can provide continuously the

Table 5. Relative and RMSE errors concerning the (a) distance estimation, (b) roll, (c) pitch, and (d) yaw angle estimations.[a]

(a) Nominal distances *D* (m)	Actual distances *D* (m)	Relative errors ε_r (%)
0.26	0.2636	**0.98**
0.31	0.3130	0.66
0.36	0.3635	0.69
0.41	0.4130	0.49
0.46	0.4626	0.34
0.51	0.5127	0.33
0.56	0.5647	0.66
0.61	0.6140	0.49
0.66	0.6647	0.56
0.71	0.7145	0.50
0.76	0.7644	0.44
0.81	0.8149	0.49
0.86	0.8637	0.31
0.91	0.9163	0.59
0.96	0.9658	0.50
1.01	1.0170	0.59
1.06	1.0688	0.74
1.11	1.1184	0.67
1.16	1.1679	0.59
1.21	1.2176	0.55
1.26	1.2709	0.78
1.31	1.3184	0.56
1.36	1.3679	0.51
1.41	1.4230	0.85
	RMSE (m): 0.006	

(b) Nominal angles Roll (°)	Actual angles Roll (°)	Relative errors ε_r (%)
150.00	150.1326	0.09
155.00	154.3896	0.39
160.00	159.4932	0.32
165.00	164.7785	0.13
170.00	169.2717	**0.43**
175.00	174.5232	0.27
180.00	179.9280	0.04
185.00	185.3836	0.21
190.00	190.4722	0.25
195.00	195.0196	0.01
200.00	200.4174	0.21
205.00	205.4830	0.24
210.00	210.5108	0.24
	RMSE (°): 0.438	

(c) Nominal angles Pitch (°)	Actual angles Pitch (°)	Relative errors ε_r (%)
−29.60	−29.3945	0.69
−24.60	−24.4793	0.49
−19.60	−19.4186	0.93
−14.60	−14.9046	2.09
−9.60	−9.8479	2.58
−4.60	−4.7406	3.06
0.40	0.3858	3.54
5.40	5.2828	2.17
10.40	9.9858	**3.98**
15.40	15.2775	0.80
20.40	20.6931	1.44
25.40	25.0902	1.22
30.40	30.4808	0.27
	RMSE (°): 0.224	

(d) Nominal angles Yaw (°)	Actual angles Yaw (°)	Relative errors ε_r (%)
−33.50	−33.7449	0.73
−28.50	−28.6035	0.36
−23.50	−23.4668	0.14
−18.50	−18.5753	0.41
−13.50	−13.4142	0.64
−8.50	−8.5186	0.22
−3.50	−3.5506	**1.45**
1.50	1.5208	1.39
6.50	6.5317	0.49
11.50	11.3883	0.97
16.50	16.5284	0.17
21.50	21.7831	1.32
26.50	26.5818	0.31
	RMSE (°): 0.121	

[a]A displacement of 0.05 m and a rotation step of 5° has been considered in the distance and angle tests, respectively. The values in bold represent the highest relative errors related to each table.

reference signals to rotors through the data returned from the vision system.

The proposed algorithm has been programmed in Matlab which is not actually indicative of the processing time for real experiments. However, by comparing our method with other works mentioned in "Related works" section in terms of processing times, we obtain the results reported in Table 6. Our methodology in searching the helipad marks is the fastest against to others by considering the average processing time. Although the method in the study by Lee et al.[11] presents almost similar processing times, it is worth underlining that it was programmed on a dedicated onboard processing unit. Therefore, the algorithm code is already optimized to run in real-time contexts. Conversely, our algorithm performs much better than the method in the study by Prakash and Saravanan[10] which has been realized in Matlab.

Higher consumption times are evident in the study by de Oliveira et al.[18] as well. Although the lowest processing times found in that paper have been reported in Table 6, it is possible to appreciate how our method is far better. In fact, the other classifiers require more processing times to provide the classification output, namely to affirm if the helipad has been found or not.

The algorithm has been run on two machines having different hardware features. Specifically, the first computer was a laptop (PC#1) with Intel(R) Core™ 2 Extreme CPU @ 3.06 GHz, Microsoft Win7 Professional with SP-1 64 bit operating system, and 4 GB of RAM.

The installed version of Matlab was the release R2013a 64-bit. The second computer was a desktop (PC#2) having

Table 6. Processing times of the main methods in detecting the helipad and estimating its pose.[a]

Method	# Frames	Average helipad detection time	Maximum helipad detection time (ms)	Average pose estimation time (ms)	Maximum pose estimation time (ms)	Total average time (ms)
Prakash and Saravanan[10]	10	284.3	295.1	—	—	—
Lee et al.[11]	1500	82.4	95.2	—	—	—
de Oliveira et al.[18]	29	234.3	252.4	—	—	—
Our						
PC#1	2000	**80.2**	114.3	416.2	563.4	496.4
PC#2		**69.5**	85.4	367.3	445.3	437.8

[a]The average processing times of our method obtained by analyzing 2000 frames on two different computers are the lowest ones. The values in bold represent the two lowest average helipad detection times.

Intel(R) Core™ i5-3470 CPU @ 3.20 GHz, RAM of 8 GB, and Win7 Professional 64-bit operating system. It uses the Matlab release R2015a 64-bit. As observable from Table 6, the second machine offers higher performances, as predictable from the better features owned.

The results obtained with the proposed approach are very encouraging since the implementation of the processing algorithm on an embedded system using general-purpose programming languages (e.g. C, C++, C#) should reduce significantly the needed elaboration times and allow us to affirm that the proposed methodology can be used in real-time context.

Conclusions and further works

In this article, a fast, robust, and accurate vision-based algorithm for assisting UAV landing has been presented. The method aims at identifying the helipad in the image. Once the H has been found by means of cascade checks, its corners are detected by using a method based on the curvature computation. Then, the 12 corners are matched with the corresponding ones expressed in the reference system fixed on the H. Finally, the homography matrix is estimated in order to evaluate the 3-D pose of heliport with respect to the camera reference system.

A quantitative validation of the proposed algorithm has been carried out through controlled indoor and outdoor experiments performed at our institute. In this regard, outcomes show small errors, lower than 1% and 4% in the distance and angle estimations, respectively.

Further proves have been done by analyzing stationary frames of the heliport. Small fluctuations of the pose estimation have been observed, especially when the target plane and the camera plane are considerably tilted. Finally, an average helipad detection time of about 80 ms has been found, whose value is the lowest in comparison with other literature works.

Future works will lead to the implementation of the presented method on a low-cost single-board computer in order to assess the camera pose in real time. Specifically, the vision system will be attached to a UAV and actual attitude information will be the input of the control system of drone. In this way, the robustness and reliability of algorithm in supporting the UAV landing will be further proven.

Acknowledgement

The authors would like to thank Mr. Michele Attolico for his valuable contribution during the experiments and its technical support.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

References

1. Borenstein J, Everett HR, Feng L, et al. Mobile robot positioning-sensors and techniques. *Robot Syst* 1997; 14(4): 231–249.
2. Milella A, Di Paola D and Cicirelli G. RFID tag bearing estimation for mobile robot localization. In: *International conferences on advanced robotics (ICAR 2009)*, Munich, Germany, 22–26 June 2009, pp. 1–6. Washington, D.C.: IEEE.
3. Cicirelli G, D'orazio T and Distante A. Target recognition by components for mobile robot navigation. *Exp Theor Artif Int* 2003; 15(3): 281–297. DOI: 10.1080/0952813021000039430.
4. Patruno C, Marani R, Nitti M, et al. An embedded vision system for real-time autonomous localization using laser profilometry. *IEEE Trans Int Trans Syst* 2015; 16(6): 3482–3495. DOI: 10.1109/TITS.2015.2459721.
5. D'Orazio T, Lovergine FP, Ianigro M, et al. Mobile robot position determination using visual landmarks. *IEEE Trans Indus Elect* 1994; 41(6): 654–662. DOI: 10.1109/41.334582.
6. Lange S, Sünderhauf N and Protzel P. Autonomous landing for a multirotor UAV using vision. In: *In SIMPAR 2008 international conference on simulation, modeling and programming for autonomous robots*, Venice, Italy, 3–4 November 2008, pp. 482–491.
7. Lange S, Sunderhauf N and Protzel P. A vision based onboard approach for landing and position control of an autonomous multirotor UAV in GPS-denied environments. In: *International conference on advanced robotics (ICAR 2009)*, Munich, Germany, 22–26 June 2009, pp. 1–6. Washington, D.C.: IEEE.
8. Kim J, Lee YS, Han SS, et al. Autonomous flight system using marker recognition on drone. In: *Workshop on frontiers*

of computer vision (FCV), Mokpo, South Korea, 28–30 January 2015, pp. 1–4. Washington, D.C.: IEEE. DOI: 10.1109/FCV.2015.7103712.

9. James A, Amrutha V and Haridasan P. Vision based algorithm for automatic landing system of unmanned aerial vehicles: a review. *Int J Innovat Sci Eng Technol (IJISET)* 2015; 2(4): 1126–1129.
10. Prakash RO and Saravanan C. Autonomous robust helipad detection algorithm using computer vision. In: *International conference on electrical, electronics, and optimization techniques (ICEEOT)*, Chennai, India, 3–5 March 2016, pp. 2599–2604. Washington, D.C.: IEEE. DOI: 10.1109/ICEEOT.2016.7755163.
11. Lee S, Jang JW and Baek KR. Implementation of vision-based real time helipad detection system. In: *International conference on control, automation and systems (ICCAS)*, Jeju Island, Korea, 17–21 October 2012, pp. 191–194. Washington, D.C.: IEEE.
12. Saripalli S, Montgomery JF and Sukhatme GS. Visually guided landing of an unmanned aerial vehicle. *IEEE Trans Robot Autom* 2003; 19(3): 371–380. DOI: 10.1109/TRA.2003.810239.
13. Saripalli S, Sukhatme GS and Montgomery JF. An experimental study of the autonomous helicopter landing problem. In: *Experimental Robotics VIII*. Heidelberg: Springer Berlin, 2003, pp. 466–475. DOI: 10.1007/3-540-36268-1_42.
14. Saripalli S, Montgomery JF, Sukhatme GS, et al. Vision-based autonomous landing of an unmanned aerial vehicle. In: *International conference on robotics and automation (ICRA 02)*, Washington, DC, USA, 11–15 May 2002, pp. 2799–2804. Washington, D.C.: IEEE. DOI: 10.1109/ROBOT.2002.1013656.
15. Sanchez-Lopez JL, Saripalli S, Campoy P, et al. Toward visual autonomous ship board landing of a VTOL UAV. In: *International conference on unmanned aircraft systems (ICUAS)*, Atlanta, GA, USA, 28–31 May 2013, 2013. pp. 779–788. Washington, D.C.: IEEE. DOI: 10.1109/ICUAS.2013.6564760.
16. Sanchez-Lopez JL, Pestana J, Saripalli S, et al. An approach toward visual autonomous ship board landing of a VTOL UAV. *J Int Robot Syst* 2014; 74(1–2): 113–127. DOI: 10.1007/s10846-013-9926-3.
17. Silva J, Mendonça R, Marques F, et al. Saliency-based cooperative landing of a multirotor aerial vehicle on an autonomous surface vehicle. In: *International conference on robotics and biomimetics (ROBIO)*, Bali, Indonesia, 5–10 December 2014, pp. 1523–1530. Washington, D.C.: IEEE. DOI: 10.1109/ROBIO.2014.7090550.
18. de Oliveira CS, Anvar AP, Anvar A, et al. Comparison of cascade classifiers for automatic landing pad detection in digital images.
19. Wu CJ and Tsai WH. An omni-vision based localization method for automatic helicopter landing assistance on standard helipads. In: *International conference on computer and automation engineering (ICCAE)*, Singapore, Singapore, 26–28 February 2010, pp. 327–332. Washington, D.C.: IEEE. DOI: 10.1109/ICCAE.2010.5451607.
20. Premachandra C, Yoshida T and Kato K. A basic study of landing system for multicopters using Raspberry Pi. In: *International symposium on consumer electronics (ISCE)*, Madrid, Spain, 24–26 June 2015, pp. 1–2. Washington, D.C.: IEEE. DOI: 10.1109/ISCE.2015.7177806.
21. Lin S, Garrat MA and Lambert AJ. Monocular vision-based real-time target recognition and tracking for autonomously landing an UAV in a cluttered shipboard environment. *Auton Robot* 2017; 41(4): 881–901. DOI: 10.1007/s10514-016-9564-2.
22. Gonzalez RC and Woods RE. *Digital Image Processing*, 3rd ed. Upper Saddle River, New Jersey: Pearson Education, 2008.
23. Lim JS. *Two-Dimensional Signal and Image Processing*, 1st ed. Englewood Cliffs, NJ: Prentice Hall, 1990, p. 710.
24. Bradley D and Roth G. Adaptive thresholding using the integral image. *J Graph Gpu Game Tools* 2007; 12(2): 13–21. DOI: 10.1080/2151237X.2007.10129236.
25. Harris C and Stephens M. A combined corner and edge detector. In: *International conference on Alvey vision*, 1988, pp. 147–151. Romsey, UK: Plessey Research Roke Manor.
26. Shi J and Tomasi C. Good features to track. In: *Internatinal conference on computer vision and pattern recognition (CVPR '94)*, Seattle, WA, USA, 21–23 June 1994, pp. 593–600. Washington, D.C.: IEEE. DOI: 10.1109/CVPR.1994.323794.
27. Moravec HP. Obstacle avoidance and navigation in the real world by a seeing robot rover. 1980; ADA092604. Computer Science Department, Stanford University. Report No. STAN-CS-80-813.
28. Hough VCP. Method and means for recognizing complex patterns. Patent No. 3,069,654, USA, 1962.
29. Zhang Z. Flexible camera calibration by viewing a plane from unknown orientations. In: *International conference on computer vision*, Kerkyra, Greece, 20–27 September 1999, pp. 666–673. Washington, D.C.: IEEE. DOI: 10.1109/ICCV.1999.791289
30. Heikkila J and Silven O. A four-step camera calibration procedure with implicit image correction. In: *International conference on computer vision and pattern recognition (CVPR)*, San Juan, Puerto Rico, USA, 17–19 June 1997, pp. 1106–1112. Washington, D.C.: IEEE. DOI: 10.1109/CVPR.1997.609468.
31. Camera Calibration Toolbox for Matlab [Internet]. https://www.vision.caltech.edu/bouguetj/calib_doc/ (accessed 21 March 2017).
32. Hartley R and Zisserman A. *Multiple View Geometry in Computer Vision*, 2nd ed. United States of America, New York: Cambridge University Press, 2003. p. 655. DOI: 10.1017/CBO9780511811685.
33. IDS. https://www.1stvision.com/cameras/IDS/dataman/UI-2280SE-M-GL%20Rev.3.pdf (accessed 21 March 2017).
34. PREXISO. http://www.prexiso.com/media/Prexiso_X2_Manual_multilingual.pdf (accessed 16 June 2017).