Localization of a Drone for Landing Using Image-Based Visual Servoing with Image Moments

Mostafa Hegazy, Riby Abraham Boby, and Alexandr Klimchik

1 Introduction

Unmanned aerial vehicles (UAVs) have many useful applications ranging from surveillance [1], search [2], agriculture [3] to border patrol [4], and mapping [5]. In these applications, fully autonomous UAVs play an important and key role, because they perform tasks without the guidance of humans. The majority of drones come with a variety of sensors, including Inertial Measurement Units (IMUs), GPS, compasses, barometers, monocular, stereo cameras, and LIDAR sensors, among others. These sensors are used to localize the drone and to gather information about the surroundings in order to map or avoid the various obstacles around the drone. The key point is to ensure that the drone has a high autonomy level with the use of robust and reliable navigation systems [6] and accurate localization. An important requirement to achieving this level of autonomy is to guarantee the takeoff as well as the landing phases to be totally autonomous. More importantly during the landing phase, the vehicle has to land safely on the landing platform [7]. There are two kinds of landing sites, the first one is that which is known to the vehicle before hand and the vehicle should reach it with the local positioning information [8]. The second kind is that

Riby Abraham Boby and Alexandr Klimchik were supported by RSF research grant No. 22-41-02006.

M. Hegazy (✉)
Innopolis University, Tatarstan, Russia
e-mail: m.hegazy@innopolis.university

R. A. Boby
Mechanical Engineering, IIT, Jodhpur, Rajasthan, India
e-mail: ribyab@gmail.com

A. Klimchik
School of Computer Science, University of Lincoln, Lincoln, UK
e-mail: alexandr.klimchik@gmail.com

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2024

S. K. Ghoshal et al. (eds.), *Recent Advances in Industrial Machines and Mechanisms*, Lecture Notes in Mechanical Engineering,
https://doi.org/10.1007/978-981-99-4270-1_13

where the vehicle has to land in unseen or unknown environments [9]. In the first kind, the landing site is made from easily recognizable marker or markers so that it can be easily detected when the vehicle has reached the boundary of the landing site. The landing phase then is activated when the camera detects the special characteristics of the marker. In the second kind, the criteria of flatness, spaciousness, and surface robustness are used to determine the best landing places. Landing platforms are used for the first case where we know before hand the special characteristics of the platform and use it to perform the landing strategy. One of the most promising methods for extending the operational range of UAVs with the least amount of vehicle modification is to use landing platforms. Additionally, automatic landing platforms could carry out tasks like picking up or loading goods, data exchange and processing, etc., in addition to the battery charge or replacement function [10]. There are two types of landing platforms which are the stationary type and the moving type. This work will focus on the first kind of landing site, which will be a set of markers known beforehand, and the platform will be stationary.

There are solutions in the literature based on motion capture systems [11] or other sensors [12]. Additionally, several articles on moving targets focused on how a flying aircraft and a ground vehicle worked together to plan the landing maneuver [14, 15]. In this work however, it is assumed that the vehicle has no communication with the platform. Another article [16] demonstrates an onboard computer vision system for estimating a UAV's pose in relation to a landing target based on a coarse-to-fine approach using a monocular camera. Different methods can be thought of to land the drone after detecting the marker of the landing platform such as image-based visual servoing [17, 18] which involves the use of computer vision data to regulate a robot's movements. Though popularly used methods rely on calculating the pose of the drone after detecting the landmark and then transforming the pose from 3D camera coordinates into 3D world coordinates, it might lead to unnecessary errors if the camera is not calibrated properly. Some of these steps can be skipped with the help of visual servoing. The control needs to be modified to get the 2D image coordinates as input; this will make the process more robust to tiny calibration errors, which will be significant while transforming into world coordinates.

2 Methodology

In Image Based Visual Servoing (IBVS) [17, 18], the time variation $\dot{s}$ of the visual features $\mathbf{s}$ can be linearly expressed with respect to the relative velocity of the camera $\mathbf{v_c}$. The control is created to ensure that the visual features decouple exponentially to reach the required value $\mathbf{s}^*$, where $\mathbf{L_s}$ is the interaction matrix associated with $\mathbf{s}$, and considering an eye-in-hand system observing a static object, then

$$\mathbf{v_c} = -\lambda \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger} (\mathbf{s} - \mathbf{s}^*), \;\; \dot{s} = \mathbf{L_s} \mathbf{v_c}, \tag{1}$$

where $\widehat{\mathbf{L}_\mathbf{s}^\dagger}$ is an approximation of $\mathbf{L_s}$ and its pseudo-inverse, λ is a positive gain responsible for the time to convergence. Given the mounting details and the UAV's center of mass, this velocity twist vector needs to be rotated and translated.

The key here is the selection of the visual features to ensure the convergence of the controller to the desired position, one might select the individual corner points of an aruco markers as visual features, but the image moments of that marker were chosen instead. The reason is that image moments provide general representation of any object that can be segmented in an image. They are also more intuitive and meaningful than just the corners of an object [19, 20]. For a discrete set of n image points, the moments m_{ij} and and the centered moments μ_{ij} are defined by [21]:

$$m_{ij} = \sum_{k=1}^{n} x_k^i y_k^j,\ \ \mu_{ij} = \sum_{k=1}^{n} (x_k - x_g)^i (y_k - y_g)^j, \tag{2}$$

where n is the number of image points, $x_g = \frac{m_{10}}{n}$, $y_g = \frac{m_{01}}{n}$ and $m_{00} = n$. It is known that these centered moments are invariant to 2D translational motion.

Next, interaction matrix may be defined. If planar objects are taken into account and for each object point, the degenerate case when the camera optical center is on the plane is excluded $1/Z = Ax + By + C$. For any point in the image, the velocity $\boldsymbol{x_k}$ is given from [17]. Using 1 to set $s = \boldsymbol{x_k}$ we obtain:

$$\begin{aligned} \dot{x_k} &= -(Ax_k + By_k + C)v_x + x_k(Ax_k + By_k + C)v_z + x_k y_k \omega_x \\ &\quad - (1 + x_k^2)\omega_y + y_k\omega_z \end{aligned} \tag{3}$$

$$\begin{aligned} \dot{y_k} &= -(Ax_k + By_k + C)v_y + y_k(Ax_k + By_k + C)v_z - x_k y_k \omega_y \\ &\quad + (1 + y_k^2)\omega_x + x_k\omega_z. \end{aligned} \tag{4}$$

The first two components of the interaction matrix that relate x_g and y_g:

$$L_{x_g} = \left[-\frac{1}{Z_g}\ 0\ x_{g_{vz}}\ x_{g_{\omega x}}\ x_{g_{\omega y}}\ y_g\right],\ L_{y_g} \quad = \left[0\ -\frac{1}{Z_g}\ y_{g_{vz}}\ y_{g_{\omega x}}\ y_{g_{\omega x}}\ -x_g.\right] \tag{5}$$

From the observations taken from [21], the centered moments will be invariant to tranlsational motions if $A = B = 0$ that happens when the object and the image plane are parallel to each other.

In [19, 22], to control the three translational motions these visual features have been selected: x_g, y_g the center of gravity and the area of the object in the image a. Based on [21] the interaction matrix can be expressed as

$$\begin{aligned} L_{x_g} &= \left[-C\ 0\ Cx_g\ \epsilon_1\ -(1+\epsilon_2)\ y_g\right],\ L_{y_g} \quad = \left[0\ -C\ Cy_g\ 1+\epsilon_3\ -\epsilon_1\ -x_g\right], \\ L_a &= \left[0\ 0\ 2a\delta C\ 3a\delta y_g\ -3a\delta x_g\ 0.\right] \end{aligned} \tag{6}$$

A modification to this interaction matrix will be made by adding normalization in the form $a_n = z^*\sqrt{\frac{a^*}{a}},\ x_n = a_n x_g,\ \text{and} y_n = a_n y_g$, where a^* is the desired area of the object in the desired image, and z^* is the desired depth between the camera and the object in the image. The resulting interaction matrix elements after the modification will be

$$\begin{bmatrix} L_{x_n} \\ L_{y_n} \\ L_{a_n} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & a_n\epsilon_{11} & -a_n(1+\epsilon_{12}) & y_n \\ 0 & -1 & 0 & a_n(1+\epsilon_{21}) & -a_n\epsilon_{22} & -x_n \\ 0 & 0 & -1 & -a_n\epsilon_{31} & a_n\epsilon_{32} & 0. \end{bmatrix} \tag{7}$$

This new interaction matrix has a decoupling property to control the three translational velocities, and the three features have the same dynamics. For discrete objects, since $\mu_{20} + \mu_{02}$ is invariant to 2D rotation and translation [21] $a = \mu_{20} + \mu_{02}, \ a^* = \mu^*_{20} + \mu^*_{02}$. Since the high-level control of the drone can only control 4 dofs, v_x, v_y, v_z and yaw rate, so we will keep only those columns from the full interaction matrix resulting in

$$\begin{bmatrix} L_{x_n} \\ L_{y_n} \\ L_{a_n} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & y_n \\ 0 & -1 & 0 & -x_n \\ 0 & 0 & -1 & 0. \end{bmatrix} \tag{8}$$

In [19, 22], to control the three rotational motions, these visual features have been selected: the orientation of the object in the image $\alpha = \frac{1}{2}\arctan 2(2\mu_{11}, (\mu_{20} - \mu_{02}))$ and two moment invariants c_i, c_j chosen from combination of image momments that are invariant to 2D translation, rotation, and scale. A modification to the general form of the orientation of the object has been made to be specific to the orientation of the fiducial marker. The algorithm detects the four corners with a fixed order, the new angle will be the angle between the first corner, and the third corner of the detected marker: $\alpha = \arctan 2((y_2 - y_1), (x_2 - x_1))$. The interaction matrix has the following form:

$$\begin{bmatrix} L_{c_i} \\ L_{c_k} \\ L_{\alpha} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & c_{i_{\omega x}} & c_{i_{\omega y}} & 0 \\ 0 & 0 & 0 & c_{j_{\omega x}} & c_{j_{\omega y}} & 0 \\ 0 & 0 & 0 & \alpha_{\omega x} & \alpha_{\omega_y} & -1 \end{bmatrix} \tag{9}$$

Since we only need to control the yaw rate, only the last row with the 4th and 5th columns removed will be used i.e., $L_\alpha = \begin{bmatrix} 0 \ 0 \ 0 \ -1 \end{bmatrix}$. Finally, the following interaction matrix will be used in the IBVS approach based on our 4 DOFs high-level control:

$$\begin{bmatrix} L_{x_n} \\ L_{y_n} \\ L_{a_n} \\ L_{\alpha} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & y_n \\ 0 & -1 & 0 & -x_n \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1. \end{bmatrix} \tag{10}$$

The full set of new visual features $s = [x_n, \ y_n, \ a_n, \ \alpha]$ and the corresponding error vector $e = \begin{bmatrix} x_n - z^* x^*_g \ y_n - z^* y^*_g \ a_n - z^* \ \alpha - \alpha^*. \end{bmatrix}^T$

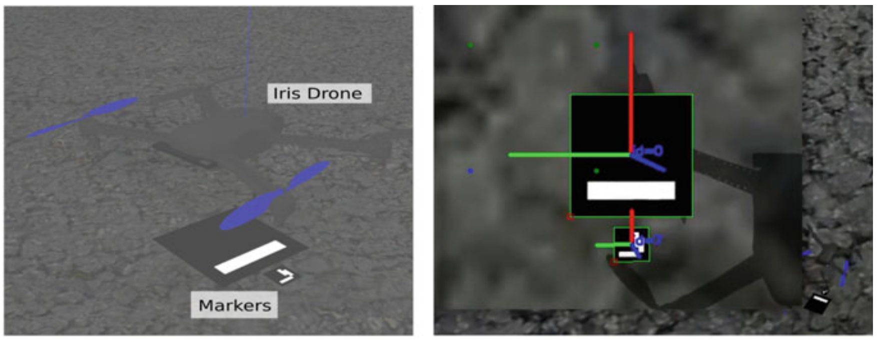
The following approximation to the interaction matrix has proven to have a good performance in practice [18, 23] $\widehat{\mathbf{L}_\mathbf{s}^\dagger} = \frac{1}{2}(L_{s(s)}^\dagger + L_{s(s^*)}^\dagger)$, where $L_{s(s)}^\dagger$ is the interaction matrix computed from the measured features, while $L_{s(s*)}^\dagger$ is computed from the desired features.

According to the findings of [26], the velocity components created by the controller employing this interaction matrix do not have significant oscillations and give a smooth trajectory in the image and in three dimensions. The gain λ in the proposed controller is adaptive, which means the error value controls the magnitude of the gain $\lambda = (\lambda_{\max} - \lambda_{\min})(\frac{||e||}{||e_{\max}||}) + \lambda_{\min}$, where $\lambda_{\max}$, $\lambda_{\min}$ are the maximum and minimum values of the gain respectively. At time t, $||e||$ is the norm of the error vector and at the start of the program, $||e_{\max}||$ is defined as the max. error in the control loop according to [25], where the authors show the effect and smoothness of adaptive gains in different visual servoing scenarios. This was adopted in this work.

3 Evaluation and Discussion

The current environment contains an Iris drone [27] inside a Gazebo world, equipped with a down-facing camera and two ArUcO markers with sizes 0.176 and 0.05 m, respectively, the code was written as a ROS package [24]. The environment is depicted in Fig. 1. The first marker will localize the drone at a $h = 0.6$ m and centered with respect to the marker, and the second marker will localize it with $h = 0.25$ m and centered. The threshold of the norm of error was empirically chosen as 0.1 m and 0.015 m.

The algorithm will do the following: Firstly, detect the larger ArUcO marker in order to localize the drone at the desired $h = 0.6$ m. Once detected, the IBVS module will initialize and recursively guide the drone to the desired location until the error threshold is reached. Second, once the first error threshold is reached and the second marker is detected, the IBVS module will initialize again with the new parameters



(a) Iris Drone in the simulation environment (b) Fiducial markers detected using camera

Fig. 1 Iris drone and fiducial markers

to recursively guide the drone to the next desired location at $h = 0.25$ m. Later, once the second error threshold is reached, the motors are turned off for landing.

Algorithm 1 Landing algorithm with IBVS

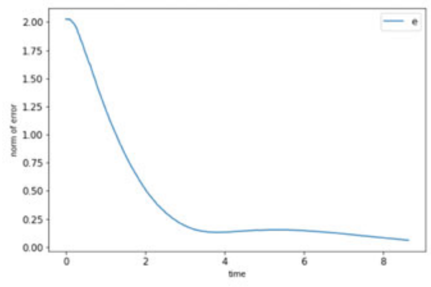
```
Require: Aruco marker 1 Detection
Ensure: $a^* =$ desired area 1, $z^* =$ desired height 1
  $n \leftarrow 4$
  while $e \geq$ error threshold 1 do
     1) Compute $m_{ij}$ based on Eq. (3)                    ▷ Start of IBVS Block
     2) Compute $x_g$, $y_g$ and $\mu_{ij}$ based on Eq. (4)
     3) Calculate $a$ based on Eq. (18)
     5) Compute $a_n$, $x_n$, $y_n$ based on Eq. (16)
     6) Compute $\boldsymbol{L_s}$ based on Eq. (26)
     7) Calculate $e$ based on Eq. (27)
     8) Generate $\mathbf{v_c}$ based on Eq. (2)             ▷ End of IBVS Block
  end while
Require: Aruco marker 2 Detection
Ensure: $a^* =$ desired area 2, $z^* =$ desired height 2
  while $e \geq$ error threshold 2 do
     Repeat IBVS block
  end while
  Turn off Motors and Land
```

In the main experiment, the value of the parameters: $\lambda_{\max} = 1.0$ and $\lambda_{\min} = 0.5$. The drone starts at the world position $x = \begin{bmatrix} 0.2 \ 0.2 \ 2 \end{bmatrix}$, with the positions measured in meters. The desired height for the first marker is $h = 0.6$ m and the final position of the drone is $x = \begin{bmatrix} 0.205 \ 0.203 \ 0 \end{bmatrix}$. The results of the IBVS algorithm for each marker in the experiment are shown in Fig. 2a–c for marker 1 and Fig. 2d–f for marker 2.

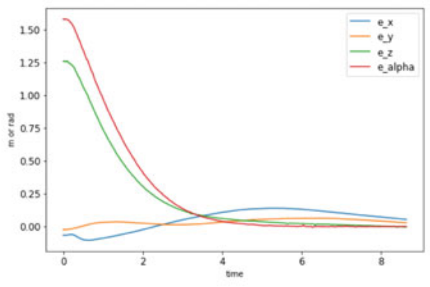
In case of rough localization, we can see that the controller converges with decoupled error and reaches the error threshold in about 8 s with the low maximum gain, the trajectories are also smooth and there are no irregularities in the control signal. While using second marker in the stage of fine localization, from the data in the graphs we can notice the noisy nature of the drone and the problem becomes much harder for the controller due to the low error signal and the fine-positioning required to land. The system is converged in about 6 s.

4 Conclusion

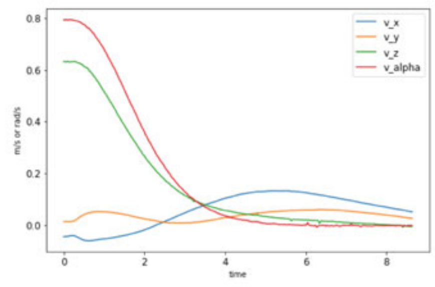
This article proposed an image-based visual servoing controller for UAV that was able to converge and land the vehicle accurately using only the image moments as the visual features from two fiducial markers. This could be done without any local position or global position information helping the controller. The only step needed before the deployment of the controller is the learning step of the desired height and image moments that will be used as a reference for the controller. Once the desired image moments and height are known, the controller will stabilize and guide the



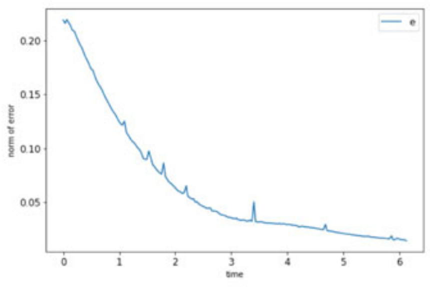
(a) Error convergence for the first marker



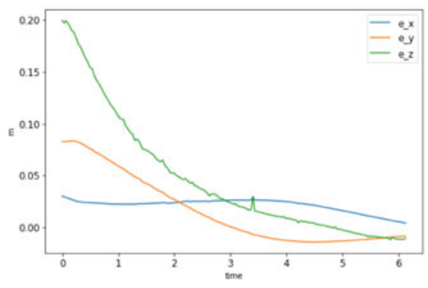
(b) The errors along the axes



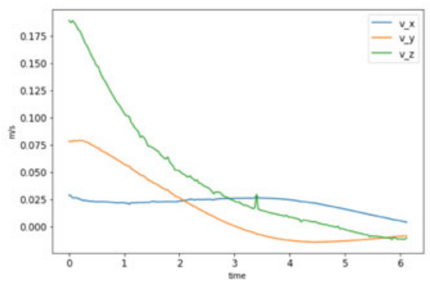
(c) High level control individual signals



(d) Error convergence for the second marker



(e) The errors along the axes



(f) High level control individual signals

Fig. 2 Simulation results

drone to land on the predefined set of markers. The convergence of the UAV to the desired position has been verified through simulations. In the future, the methodology may be extended to handle a moving platform and land on it.

References

1. Geng L, Zhang Y, Wang J, Fuh J, Teo A (2013) Mission planning of autonomous UAVs for urban surveillance with evolutionary algorithms. In: 10th IEEE international conference on

control and automation (ICCA), pp 828–833
2. Waharte S, Trigoni N (2010) Supporting search and rescue operations with UAVs. In: Emerging security technologies (EST), pp 142–147
3. Herwitz S, Johnson L et al (2004) Imaging from an unmanned aerial vehicle: agricultural surveillance and decision support. Comput Electron Agric 44(1):49–61
4. Girard AR, Howell AS, Hedrick JK (2004) Border patrol and surveillance missions using multiple unmanned air vehicles. In: 43rd decision and control, vol 1, pp 620–625
5. Nagai M, Chen T, Shibasaki R, Kumagai H, Ahmed A (2009) UAV-borne 3-d mapping system by multisensor integration. IEEE Trans Geosci Rem Sens 47(3):701–708
6. Kendoul F (2012) Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. J Field Rob 29(2):315–378
7. Kong W, Zhou D, Zhang D, Zhang J (2014) Vision-based autonomous landing system for unmanned aerial vehicle: a survey. In: 2014 MFI, pp 1–8
8. Lange S, Sunderhauf N, Protzel P (2009) A vision based onboard approach for landing and position control of an autonomous multirotor UAV in Gps denied environments. In: 2009 international conference on advanced robotics, pp 1–6
9. Mukadam K, Sinh A, Karani R (2016) Detection of landing areas for unmanned aerial vehicles. In: ICCUBEA, pp 1–5
10. Feng Y, Zhang C, Baek S, Rawashdeh S, Mohammadi A (2018) Autonomous landing of a UAV on a moving platform using model predictive control. Drones 2:34
11. Mellinger D, Shomin M, Kumar V (2010) Control of quadrotors for robust perching and landing. In: International powered lift conference, pp 205–225
12. Sharp CS, Shakernia O, Sastry S (2001) A vision system for landing an unmanned aerial vehicle. ICRA 2:1720–1727
13. Herisse B, Russotto F, Hamel T, Mahony R (2008) Hovering flight and vertical landing control of a vtol unmanned aerial vehicle using optical flow. In: IROS, pp 801–806
14. Daly JM, Ma Y, Waslander SL (2015) Coordinated landing of a quadrotor on a skid steered ground vehicle in the presence of time delays. Auton Robots 38(2):179–191
15. Ghamry KA, Dong Y, Kamel MA, Zhang Y (2016) Real-time autonomous take-off, tracking and landing of UAV on a moving UGV platform. In: 24th mediterranean conference on control and automation, pp 1236–1241
16. Patruno C, Nitti M, Petitti A, Stella E, D'Orazio T (2019) A vision-based approach for unmanned aerial vehicle landing. J Intell Robot Syst 95
17. Hutchinson S, Hager G, Corke P (1996) A tutorial on visual servo control. IEEE Trans Robot Autom 12:651–670
18. Chaumette F, Hutchinson S (2007) Visual servo control. I. Basic approaches. Robot Autom Mag 13:82–90
19. François C (2004) Image moments: a general and useful set of features for visual servoing. IEEE Trans Robot 20(4):713–723
20. Chaumette F (2002) First step toward visual servoing using image moments. IROS 1
21. Omar T, Francois C (2005) Point-based and region-based image moments for visual servoing of planar objects. IEEE Trans Robot 21(6):1116–1127
22. Corke Peter I, Hutchinson Seth A (2001) A new partitioned approach to image-based visual servo control. IEEE Trans Robot Autom 17(4):507–515
23. Malis E (2004) Improving vision-based control using efficient second-order minimization techniques. In: ICRA, vol 2
24. Anis K (ed) (2017) Robot operating system (ROS), vol 1. Springer, Cham
25. Olivier K, François C (2013) Dealing with constraints in sensor-based robot control. IEEE Trans Robot 30(1):244–257
26. Bastourous M et al (2020) Image based visual servoing for multi aerial robots formation. In: 28th mediterranean conference on control and automation (MED)
27. Pisa S et al (2018) Evaluating the radar cross section of the commercial IRIS drone for anti-drone passive radar source selection. In: 22nd international microwave and radar conference