UAV Autonomous Landing using Visual Servo Control based on Aerostack

Guanzheng Wang
College of Intelligence and Technology
National University of Defense Technology
Changsha 410072, China
guanzhengw@163.com

Zhihong Liu*
College of Intelligence and Technology
National University of Defense Technology
Changsha 410072, China
zhliu@nudt.edu.cn

Xiangke Wang
College of Intelligence and Technology
National University of Defense Technology
Changsha 410072, China
xkwang@nudt.edu.cn

ABSTRACT

The autonomous landing for unmanned aerial vehicles (UAVs) plays an important role in driving the unmanned systems towards practical applications. However, the traditional position-based landing control method relies heavily on satellite navigation and positioning systems, which cannot be applied in environments with weak satellite signals such as indoors and valleys. In order to solve these problems, this paper proposes a visual servo control method for the UAV autonomous landing based on Aerostack software framework. The method takes the real-time on-board camera image as input and extracts tag features on the ground. In addition, the autonomous landing visual servo law is then constructed to obtain the velocity control signals which are able to guide the drone to land accurately. The experimental results in the Gazebo simulation environment show that the proposed algorithm can achieve robust and accurate autonomous landing without satellite navigation signals. Most specifically, the distance deviation is less than 0.1 meters and the yaw deviation is less than 0.1 radians.

CCS CONCEPTS

• Theory of computation • Theory and algorithms for application domains • Computing methodologies • Model development and analysis • Computing methodologies • Real-time simulation

KEYWORDS

Aerostack, Autonomous landing, Visual Servoing, UAVs

CSAE 2019, October 22–24, 2019, Sanya, China

ACM ISBN 978-1-4503-6294-8/19/10$15.00
https://doi.org/10.1145/3331453.3361667

1 Introduction

In recent years, the techniques of unmanned aerial vehicles have been experiencing a rapid growth in both academia and industry due to their potential value in practical applications. The autonomous landing of UAVs is one of the techniques of vital importance in advancing the development of UAVs. Many applications, such as disaster rescue, cargo transportation, desire the accurate autonomous landing of UAVs for accomplishing missions [1,2].

The traditional position-based landing control method relies heavily on satellite navigation and positioning systems [3]. However, existing satellite navigation and positioning systems, such as GPS and BeiDou, are not sufficient accurate in the civilian field, and the navigation signal is easily lost when there is occlusion above the UAV, let alone in the indoor environment and valleys. To this end, visual servo control [4], a control method that produces control signal based on the visual feedback, is very attractive to solve this problem. Besides, since the sensors used in visual servo control are mainly visual sensors, the measurement of the environment is non-touch [5]. At the same time, compared with traditional navigation control method, a larger amount of environment information can be obtained, and the control precision is effectively improved [6,7].

Many approaches have been proposed to tackle the autonomous landing of UAVs. Santamaria-Navarro et al. derived an uncalibrated image-based visual servo method (UIBVS) for Urban Air Mobility (UAM), which does not require prior knowledge of the focal length and contains mild assumptions about the principle point and skew values of the camera [8]. Mittal et al. proposed a computationally efficient system, which is able to reliably identify safe landing sites [9]. Ananthakrishnan et al. combined neural networks with autonomous landings using visual sensors and described an offline neural network backpropagation controller to provide visual servoing for the landing operation [10].

Nevertheless, these approaches do not consider the scalability of integrating other missions such as path following and target tracking. Note that, to provide the high level of autonomy, the scalability of supporting multi-missions is desired. Aerostack, an

architecture framework of unmanned aircraft system, is proposed. This architecture is based on the concept of modularity that other missions and other kinds of platforms can be easily extended.

Therefore, based on Aerostack framework, this paper proposes a UAV autonomous landing visual servo control approach. Our approach not only realizes a fully autonomous and precise landing, but also leverage the scalability of Aerostack. We extend the Aerostack framework and the simulation results show that our approach can achieve robust and accurate autonomous landing. More specifically, the distance deviation is less than 0.1 meters and the yaw deviation is less than 0.1 radians.

The remainder of this paper is organized as follows. The background of the proposed approach is given in Section 2. Section 3 gives the overview of the system design. The detail of the control algorithm is elaborated in Section 3. The simulation experiment results are presented and analyzed in Section 4. Concluding remarks are finally given in Section 5.

2 Background

2.1 Aerostack Framework based on ROS

Robot Operating System (ROS) is an open-source operating system designed for robots. ROS provides libraries and tools to help software developers create applications for robots. This operating system provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more [11].

ROS provides mainly two communication methods: Topic and Service. When using the method of Topic to communicate, the Publisher (probably a node) publishes messages to a specific topic and then the Subscribers can subscribe the messages from the topic. This is not a point to point communication way. In contrast, Service allows a node to send a request and get the response, thus this approach is also known as Remote Procedure Call (RPC).

Aerostack is an open-source software architecture of unmanned aircraft system (UAS), which is built on the ground of ROS. It can help developers to design and build the control architecture of their own UAS. It also integrates a variety of different computing solutions [12]. This framework is suitable for systems of different kinds of autonomy and different numbers of the drones in the mission. One of the outstanding advantages of the Aerostack framework is its open-source property. Using this framework can save developers a lot of time, allowing them to focus on algorithm design of functionality expansion.

There are five layers in Aerostack framework, which are social layer, reflective layer, deliberative layer, executive layer and reactive layer from top to bottom. In order to specify the flight mission, it provides three method including Behavior Tree, TML language (Task-based Mission Specification Language) and Python language.

2.2 Related Coordinate Systems

The coordinate systems used in this paper mainly include Geodetic Coordinate System {O}, drone's Body Coordinate System {B} and Camera Coordinate System {C}. These coordinate systems belong to the right-handed coordinate system and are attached to the earth, the drone and the downwards camera.

Assuming that the yaw angle of the drone is φ, the pitch angle is θ, and the roll angle is γ, then the rotation relation between {O} and {B} is

$$\begin{pmatrix} x_B \\ y_B \\ z_B \end{pmatrix} = \Phi(\gamma)\Phi(\theta)\Phi(\varphi)\begin{pmatrix} x_O \\ y_O \\ z_O \end{pmatrix}. \quad (1)$$

The three rotation matrices are

$$\Phi(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix}, \Phi(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix},$$

and $\Phi(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

According to (1), the relationship (including rotation and translation) between {B} and {C} is

$$\begin{pmatrix} x_B \\ y_B \\ z_B \end{pmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}\begin{pmatrix} x_C \\ y_C \\ z_C \end{pmatrix} - \begin{pmatrix} 0 \\ 0.5 \\ 0 \end{pmatrix}. \quad (2)$$

When solving some variables, the coordinates in different coordinate systems are generally converted to drone's Body Coordinate System, and then the relevant calculations are performed, which is clearer and simpler.

In addition to the three coordinate systems described above, pixel coordinate, focal length and horizontal field of view are used. The focal length refers to the distance between the optical center of the lens and the focus at which the light is concentrated when the light is incident. The horizontal field of view refers to the angle formed by the edges of the largest range of the object image in the horizontal direction, with the camera lens as the fixed point. The hfov of the camera used in this paper is 100 degrees and resolution is 960x720. There is a relationship between the camera's focal length, image width and the horizontal field of view, which is

$$focal\_length = \frac{image\_width}{2\tan(\frac{hfov}{2})}. \quad (3)$$

Assume the pixel width is ρ_u, the pixel height is ρ_v, the units of which are meters. Besides, $\rho_u = \rho_v$, then the focal length can be calculated by

$$focal\_length = \frac{960\rho_u}{2\tan(5\pi/18)}. \quad (4)$$

Assume that there is a point P, which coordinates in {C} is (X, Y, Z), then the coordinates of the point p (P's image point) in the image plane is

$$x = focal\_length \times \frac{X}{Z}, y = focal\_length \times \frac{Y}{Z}. \quad (5)$$

In (5), the units of x and y are meters. Thus, the pixel coordinates of p can be obtained by

$$u = \frac{x}{\rho_u} + u_0, v = \frac{y}{\rho_v} + v_0. \quad (6)$$

Let $\overline{u} = u - u_0, \overline{v} = v - v_0$, then $(\overline{u}, \overline{v})$ is the pixel coordinate of point p relative to the main point. All the pixel coordinates' units are dimensionless.

3 System Design

3.1 Overview

Flight mission of the drone generally includes three steps, which are taking off, carrying out the specific task and then landing. The overall process of visual servo control for UAV autonomous landing is shown in Figure 1.

In order to test the performance of autonomous landing more scientifically, this paper manipulates the drone to fly to the target area before landing, and ensures that the drone's position coordinates obey a uniform random distribution in the area.

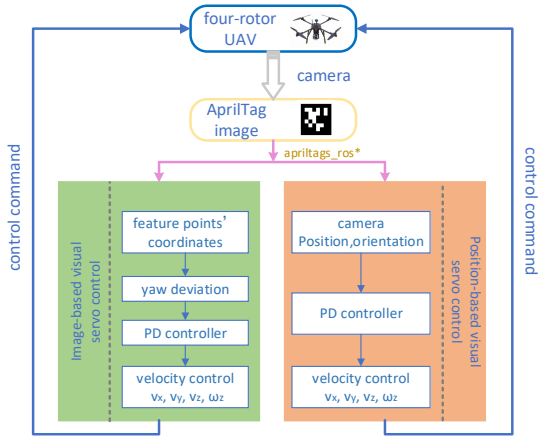

Figure 1: Visual Servo Control Process.

Throughout the project, the downwards camera is the main data source for autonomous landing. There are two ways to realize visual servo control, which are Position-Based Visual Servo (PBVS) and Image-Based Visual Servo (IBVS). It must be mentioned that the control algorithms in these two modes are not limited.

The ideal successful landing goal is that the support bar touches the ground and the downwards camera is facing the tag on the ground. If the camera is mounted directly under the drone, the support bar will block the camera's field of view and the tag will not be recognized, so mount the camera under the right arm of the drone. Therefore, when the drone lands successfully, the drone is not directly above the tag, which has little impact on the evaluation of the mission and control algorithm.

3.3 Architecture Design

Based on the Aerostack framework, this paper extends the functionality of visual servo control module and the simulation implementation in Gazebo [13]. Figure 2 shows the architecture design for realizing the UAV autonomous landing based on the visual servo control.

In the autonomous landing process, the downwards camera captures the image and the feature extractor extracts the features of the tag on the ground. The motion controller generates the velocity control signals for the drone to move based on the feature information, thereby manipulating the drone to accurately land to the ground target position.

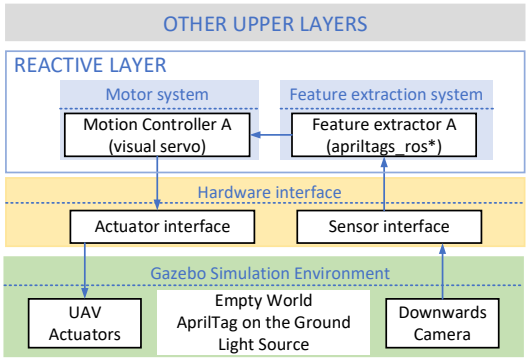

Figure 2: Architecture Design under Aerostack.

The visual servo control module belongs to the Reactive Layer of Aerostack framework. The feature extractor is obtained on the basis of the open-source package apriltags_ros. In order to obtain the control variables required for image-based visual servo control, we added the function of publishing the pixel coordinates of the center point and four corner points of the tag, thus, the package is named apriltags_ros* in this paper. It is worth mentioning that the ROS node communication method used in the extended function modules is Topic. The controller in the Motor system can use different features provided by the feature extractor to generate motion control signals for the drone using different control algorithm.

4 Visual Servo Control Algorithm based on Aerostack

4.1 Image Feature Extraction

This article uses AprilTag (similar to the QR code used daily) as the ground target point for the drone's autonomous landing, which is a visual fiducial system that uses the open-source package apriltags_ros to calculate the 3D position, orientation of the tag relative to the camera [14].

In the simulation experiments of this paper, the images captured by the downwards camera are sent to the apriltags_ros package (Modified) for processing to obtain the required feature point information, including the center point and the four corner points [15].

Motion Controller receives image feature messages from Feature Extractor. These features include, but are not limited to, the position and orientation of the center of the tag, and the pixel coordinates of the feature points. The tag used in this paper is in TAG36H11 family and its ID is 0.

4.2 Visual Servo Control Law Design

Proportional-integral-derivative controller is a control loop feedback mechanism widely used in a variety of applications requiring continuous modulated control [16]. The PID controller includes proportional control, integral control that can eliminate the offset error, and differential control that can improve the dynamic response. Therefore, the PID controller is also called a three-item controller [17].

The controller used in this paper is a PD controller, which relationship between the input and output is

$$u(t)=K_p e(t)+K_d \dot{e}(t). \quad (7)$$

In (7), K_p and K_d refer to the proportional and derivative parameters, and $u(t)$ refers to the velocity of the UAV. The input $e(t)$ refers to the deviation of position or yaw angle. These deviations can be calculated according to the information provided by the package apriltags_ros (Modified) when the tag is able to be detected.

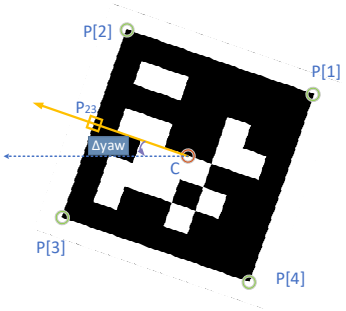

Figure 3: The Deviation of the Yaw.

In Figure 3, P[1] ~ P[4] are four corner points, C is the centroid point and P_{23} is the midpoint of P[2] and P[3]. Based on these pixel coordinates, the yaw angle deviation of drone can be calculated. The algorithm ensures that the drone rotates the minimum angle to the target yaw angle.

Four-rotor UAV is an underactuated system which position and attitude are achieved by v_x, v_y, v_z and ω_z. The overall control process is shown in Algorithm 1.

Algorithm 1 Visual Servo Control

Require: start the mission in Aerostack GUI
Ensure: accurate landing on the tag
01: subscribes camera image
02: detect AprilTag
03: **if** tag is detected **do**
04: extract feature
05: calculate deviation of position
Δ = target - current
06: **if** IBVS **do**
07: calculate deviation of yaw
$u_{23} = (u_2 + u_3)/2,\ v_{23} = (v_2 + v_3)/2$
$\Delta u = u_{23} - u_c, \Delta v = v_{23} - v_c$
if $\Delta u > 0$ **do** $\Delta yaw = \arctan(\Delta v / \Delta u) - \pi$
else if $\Delta u < 0$ **do** $\Delta yaw = \arctan(\Delta v / \Delta u)$
else do $\Delta yaw = 0$
end if
08: **end if**
09: calculate velocity control signal
$u(t)=K_p e(t)+K_d \dot{e}(t)$
10: control UAV using $v_x, v_y, v_z, \omega_z (u(t))$
11: **end if**

5 Simulation Experiments

Simulation experiment is an important part of testing the reliability and effectiveness of algorithm. In this paper, the experiments are implemented in Gazebo, as shown in Figure 4.

5.1 Construction of Simulation Environment

In Gazebo, the models that need to be created are mainly empty world, AprilTag on the ground, light sources, UAV and downwards camera. The light sources used in this paper are point sources and are placed directly above the tag.

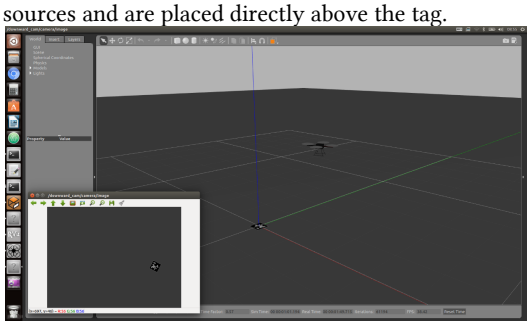

Figure 4: An Experiment in Gazebo.

5.2 Results and data analysis

Table 1: PD Controller Parameters

PBVS	K_p	K_d	IBVS	K_p	K_d
x	0.9	0.005	u	0.001	0.00001
y	0.9	0.001	v	0.001	0.00001
yaw	0.3	0.001	yaw	0.01	0.0001

The main content of the simulation experiment is to load the top-level mission design written in Python language, simulate in Gazebo environment, and obtain experimental data. In the

process of autonomous landing, the vertical speed of the UAV is set to -0.2 m/s, thus the time taken for the landing phase is approximately 20 seconds. The parameters of the PD controller are shown in Table 1, which can be adjusted according to the actual situation.

5.2.1 Position-Based Visual Servo. We performed ten simulation experiments using PBVS under different light conditions, and deviation of landing position is shown in Figure 5.

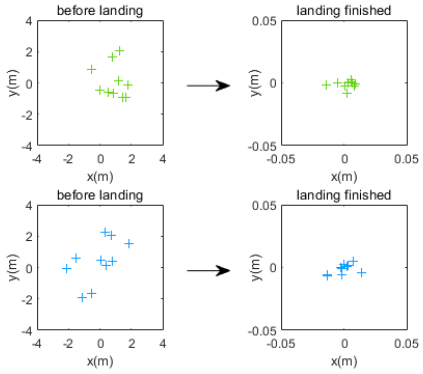

Figure 5: Landing Position Deviation (PBVS). The Green Cross Symbols Indicate the Experimental Results under 1 Time Light Intensity, while the Blue Ones for 4 Times Light Intensity.

According to the Pythagorean theorem, it can be calculated that under 1 time light intensity, the average position deviation at the completion of landing is 0.0069 meters, and the variance is 1.0924×10^{-5} square meters. Under 4 times the average position deviation is 0.0070 meters, and the variance is 3.0768×10^{-4}.

The yaw angle at the beginning and end of the landing is shown in Figure 6. Under 1 time light intensity, the average landing yaw deviation is 0.0147 rad, and the variance is 3.1363×10^{-4}. On the contrary, the average landing under 4 times light deviation is 0.0111 rad and the variance is 1.5730×10^{-4}.

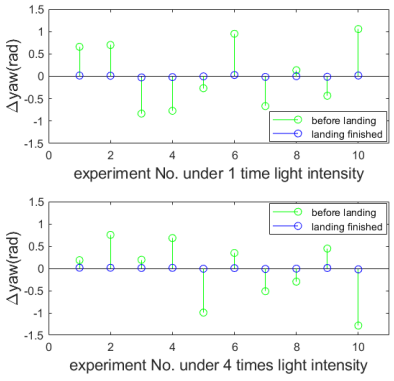

Figure 6: Landing Yaw Deviation.

In order to explain the landing process more intuitively, the position and yaw angle deviations are shown in Figure 7. This article shows three landings here.

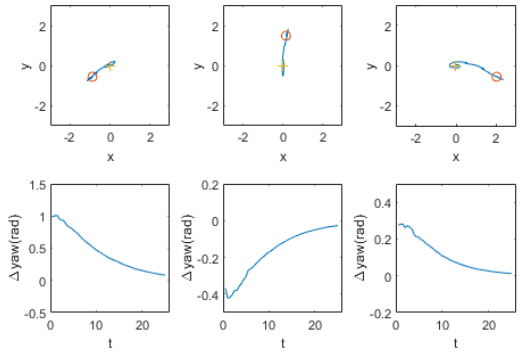

Figure 7: The Position and Yaw Angle Deviations in the Landing Process on Time Series (PBVS). The circles Refer to the Start Points and the '+' Symbols Refer to End Points.

5.2.2 Image-Based Visual Servo. Under the same simulation conditions as 5.2.1, we performmed 10 experiments under different light intensity conditions.

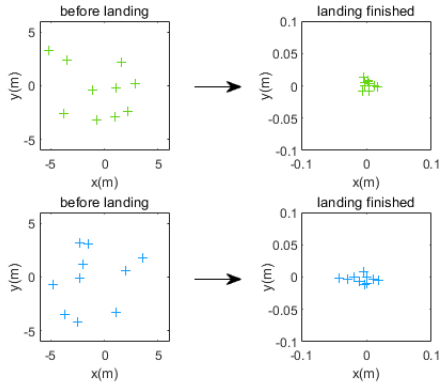

Figure 8: Landing position deviation (IBVS). The green cross symbols indicate the experimental results under 1 time light intensity, while the blue ones for 4 times light intensity.

Consistent with the previous section, we can obtain the position deviation using Pythagorean theorem. As shown in Figure 8, under 1 time light intensity, when landed, the position deviation is 0.0093 meters, and the variance is 2.2089×10^{-5} square meters. Under 4 times light intensity, the two results are 0.0167 meters and 1.4128×10^{-4} square meters.

The yaw angle deviation at the beginning and ending of landing is shown in Figure 9. Under 1 time light intensity, the average yaw deviation is 0.0044 rad and the variance is 3.4383×10^{-5}. Under 4 times light intensity, the average yaw deviation is 0.0062 rad and the variance is 4.3906×10^{-5}.

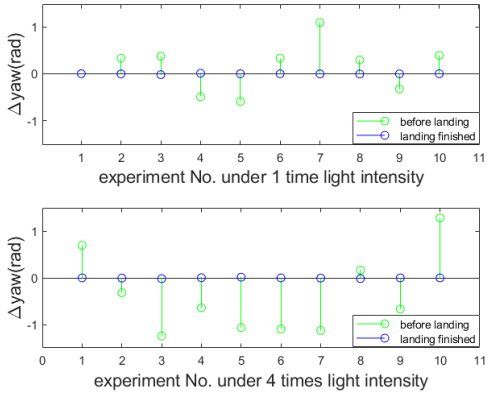

Figure 9: landing Yaw Deviation.

To better describe the landing process, similarly, Figure 10 shows the position and yaw angle deviation changes during the three landings.

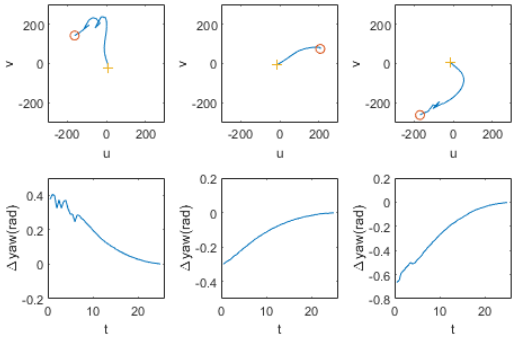

Figure 10: The Position and Yaw Angle Deviations in the Landing Process on Time Series (IBVS). The Circles Refer to the Start Points and the '+' Symbols Refer to End Points.

In the simulation experiment, the position-based and image-based visual servo control algorithms show good robustness and accuracy in the autonomous landing of the drone, which indicates that autonomous landing are achieved without satellite navigation and inertial navigation sensors.

But it must be noted that there are still many challenges in real applications. After all, the simulation environment cannot fully simulate the real environment. For example, in Gazebo, complex light and airflow cannot be simulated, which can have a large impact on the robustness and reliability of autonomous landings.

6 Conclusions

In order to solve the existing problems of inaccurate landing position and high dependence on satellite navigation during the landing of UAVs, we have designed and implemented a completely autonomous visual servo control approach for UAVs' landing based on the Aerostack framework in this paper.

Our main work includes: top-level mission design, construction of simulation environment, expanding the function of apriltags_ros package, adding position-based and image-based visual servo control node in Aerostack and building communication channel between Aerostack and Gazebo. Based on this, we have conducted a series of simulation experiments to verify the feasibility of the overall design.

The results show that the proposed visual servo control algorithms can achieve robust and accurate landing goal without satellite navigation. And the accuracy has reached the centimeter level.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China under Grant 61906209.

REFERENCES

[1] Maza, I., Caballero, F., Capitán, J., Martínez-de-Dios, J. R., & Ollero, A. (2011). Experimental results in multi-UAV coordination for disaster management and civil security applications. *Journal of intelligent & robotic systems, 61*(1-4), 563-585.

[2] Erdelj, M., & Natalizio, E. (2016, February). UAV-assisted disaster management: Applications and open issues. In *2016 ICNC)* (pp. 1-5). IEEE.

[3] Nemra, A., & Aouf, N. (2010). Robust INS/GPS sensor fusion for UAV localization using SDRE nonlinear filtering. *IEEE Sensors Journal, 10*(4), 789-798.

[4] Chaumette, F., & Hutchinson, S. (2006). Visual servo control. I. Basic approaches. *IEEE Robotics & Automation Magazine, 13*(4), 82-90.

[5] Bourquardez, O., Mahony, R., Guenard, N., Chaumette, F., Hamel, T., & Eck, L. (2009). Image-based visual servo control of the translation kinematics of a quadrotor aerial vehicle. *IEEE Transactions on Robotics, 25*(3), 743-749.

[6] Smolyanskiy, N., Kamenev, A., Smith, J., & Birchfield, S. (2017, September). Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness. In *2017 IEEE/RSJ IROS* (pp. 4241-4247). IEEE.

[7] Weckesser, P., Dillmann, R., Elbs, M., & Hampel, S. (1995, August). Multiple sensor processing for high-precision navigation and environmental modeling with a mobile robot. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots* (Vol. 1, pp. 453-458). IEEE.

[8] Santamaria-Navarro, A., Solà, J., & Andrade-Cetto, J. (2019). Visual Servo. In *Visual Guidance of Unmanned Aerial Manipulators*(pp. 57-77). Springer, Cham.

[9] Mittal, M., Valada, A., & Burgard, W. (2018). Vision-based Autonomous Landing in Catastrophe-Struck Environments. *arXiv preprint arXiv:1809.05700.*

[10] Ananthakrishnan, U. S., Akshay, N., Manikutty, G., & Bhavani, R. R. (2017). Control of Quadrotors Using Neural Networks for Precise Landing Maneuvers. In *Artificial Intelligence and Evolutionary Computations in Engineering Systems* (pp. 103-113). Springer, Singapore.

[11] TullyFoote, Documentation – ROS Wiki, http://wiki.ros.org/.

[12] Sanchez-Lopez, J. L., Fernández, R. A. S., Bavle, H., Sampedro, C., Molina, M., Pestana, J., & Campoy, P. (2016, June). Aerostack: An architecture and open-source software framework for aerial robotics. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)* (pp. 332-341). IEEE.

[13] Rodríguez-Ramos, A., Sampedro, C., Carrio, A., Bavle, H., Fernández, R. A., Milošević, Z., & Campoy, P. (2016, October). A monocular pose estimation strategy for uav autonomous navigation in gnss-denied environments. In *Proceedings of the International Micro Air Vehicle Conference and Flight Competition, Beijing, China* (pp. 17-22).

[14] Olson, E. (2011). AprilTag: A robust and flexible visual fiducial system.*2011 IEEE International Conference on Robotics and Automation.* IEEE.

[15] Wang, J., Sadler, C., Montoya, C. F., & Liu, J. C. (2016, December). Optimizing ground vehicle tracking using unmanned aerial vehicle and embedded apriltag design. In *2016 International Conference on Computational Science and Computational Intelligence (CSCI)* (pp. 739-744). IEEE.

[16] Franklin, G. F., Powell, J. D., Emami-Naeini, A., & Powell, J. D. (1994). *Feedback control of dynamic systems* (Vol. 3). Reading, MA: Addison-Wesley.

[17] Åström, K. J., & Hägglund, T. (1995).*PID controllers: theory, design, and tuning*(Vol. 2). Research Triangle Park, NC: Instrument society of America.