Autonomous Path Follow UAV to Assist Onshore Pipe Inspection Tasks

Lucas C. Sousa[1], Yago M. R. da Silva[1], Gabriel G. R. de Castro[1], Caio L. Souza[1], Guido Berger[23], Jose´ Lima[234], Diego Brandão[1], João T. Dias[1], Milena F. Pinto[1]
[1] *Federal Center of Technological Education Celso Suckow da Fonseca - CEFET/RJ*
[2] *Research Centre in Digitalization and Intelligent Robotics (CeDRI), Instituto Polite´cnico de Braganc¸a, Portugal*
[3] *Laboratório para a Sustentabilidade e Tecnologia em Regiões de Montanha (SusTEC), Instituto Polite´cnico de Braganc¸a, Portugal*
[4] *INESC Technology and Science, Porto, Portugal*
yago.silva@aluno.cefet-rj.br

2022 7th International Conference on Robotics and Automation Engineering (ICRAE) | 978-1-6654-8918-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRAE56463.2022.10056192

Abstract—Unmanned Aerial Vehicles (UAVs) are being deployed in different applications due to their reduced time execution to perform tasks, more extensive coverage area, and more risk minimization to humans. In the Oil & Gas industry, its use for inspection activities is even more attractive due to the large structures in these facilities. Therefore, this research proposes deploying an autonomous UAV system to inspect unburied pipelines of onshore O&G facilities. The proposed UAV guiding system is based on image processing techniques Canny edge detection and Hough Transform to detect the line and on a path follower algorithm to generate the trajectory. The proposed strategy was developed in Robot Operating System (ROS) and tested in a simulated environment considering the practical operational. The same controller was tested on a physical UAV to validate the results obtained in previous simulations. The results demonstrated the effectiveness and feasibility of deploying the proposed strategy for this specific task and the cost reduction potential for real-life operations, as well as reduced potential risks to the physical integrity of the workers.

Keywords—pipeline inspection, unmanned aerial vehicle, computer vision, oil and gas industry

I. Introduction

The technological progress over the years for the Unmanned Aerial Vehicles (UAVs) had led to unprecedented use of this technology in a wide range of applications, such as inspection [1], monitoring [2], search and rescue [3], security [4], [5], photogrammetry applications [6], etc. The reason is that this kind of robot can reach places that can be a hazard to humans, has the flexibility to fly, and is easy operation. In this context, inspecting extensive constructions and areas by air is a good opportunity for UAV application. For example, the work of Biundini *et al.* [7] proposed a UAV application to assess and inspect slope and dam structures, focused on maximizing the coverage inspection trajectory. A drawback is performing a manual flight first to optimize the trajectory based on flight parameters.

GPS navigation is the most common method for path planning. However, it does not cover the constant changing in the environment, which makes the GPS-based navigation inaccurate and inefficient [8]. Several researchers have proposed systems to control quadrotor UAVs. Among them, vision-based approaches have the advantage of increasing the automation level of robotic systems, resulting in less human intervention [9]. The robot can interpret situations and scenes through cameras and associated image processing, detect objects, and navigate [10]. Note that an autonomous UAV consists of many interacting related algorithms and technologies. Over the past decade, many visual sensors and high-process image analyses have been developed. Thus, nowadays, computer vision algorithms play an essential role at the forefront of this building intelligent solutions [11].

Navigation is an important aspect of several applications involving robots [12]. The navigation's complexity depends on the sensors' accuracy, environment setup, and requirements. Several algorithms are published in the literature to guide aerial, wheeled, and underwater robots [7], [13]. Among them, the Line Follower Algorithm (LFA) is a simple strategy used in a predetermined path. According to Maciel *et al.* [14], a line follower robot is designed to follow a path or a line already selected by the user. This kind of system is widely used in many different applications. For instance, in Jiang *et al.* [15], the authors proposed a solution for guiding agricultural robots to remove weeds and fertilize crops based on the idea of lines. They applied a preprocessing step to obtain binarized images that are divided into several line segments. Then, a vertical projection method was used to estimate the position of the crop rows, and the Hough transform detected the lines. They did not show the experimentation of robots in their results. In a more advanced way, the authors of Basso *et al.* [16] proposed using an algorithm to detect the crop row and then an LFA to generate the driving parameters of the UAV.

In the Oil and Gas industry (O&G), the deployment

of intelligent solutions by using automated systems has attracted huge interest in the last years [17]. For instance, the American Petroleum Institute has published guidelines for using UAVs in this sector [18]. Therefore, this work proposes a guiding system based on an LFA to assist the navigation process for autonomous inspection of unburied pipelines of onshore O&G facilities. The proposed guiding system is based on the classical image processing techniques Canny edge detection and Hough Transform to detect the lines and generate the trajectory, as well as on an LFA to provide the position adjustment. The proposed framework was developed in Robot Operating System (ROS) and tested in simulated and real environments. As a motivation for this work, the potential risks to the physical integrity of the workers who perform unburied pipeline inspections can be reduced. Besides, automated pipeline inspection using UAVs and computer vision strategy mitigates the effort of moving specialized personnel over great distances to analyze and check possible failures. This research work's main contributions can be summarized as follows:

- Using a simple strategy to guide the UAV in an autonomous way during inspections of extensive unburied pipeline structures in the O&G industry. This solution can be implemented in devices with limited processing capability for online usage;
- Testing the proposed path based on computer vision in the Iris UAV on PX4 SITL and Gazebo software with ROS to prove its functionality. The proposed strategy was also tested on a physical UAV as a proof of concept.

II. Proposed Methodology

The main objective of this work is to provide an automatic solution to the inspection of unburied pipelines in the O&G industry. In order to achieve that, the overall idea is illustrated in the simplified diagram of Figure 1. First, the framework ROS, hardware, and algorithms are initialized. Then, communication with the Flight Control Unit (FCU) PX4 is established to start the line-follower strategy. By using ROS and Python scripts, it is possible to read the camera topic messages from the UAV, convert them to an OpenCV array with CvBrigde (i.e., ROS library), and process the information to define the robot positions setpoint. The position commands are parsed to the FCU through the MavROS package via MAVlink protocol.

A. Quadrotor Model

To fully understand our proposition, it is first necessary to identify the mechanics required to perform the flight. The quadrotor UAV model is based on [19]. The motor speed variation results in the UAV rotation on its axis. Due to the generated inclination, it results in vehicle displacement in any direction of space. To understand how the variation of each individual rotor

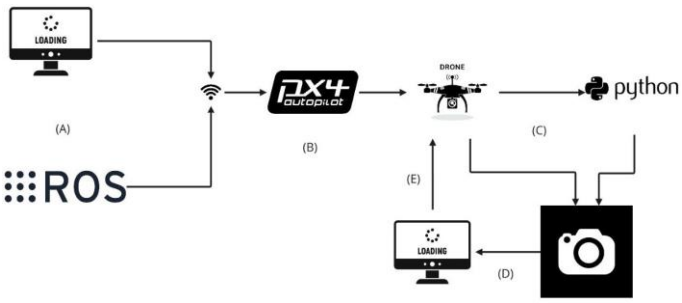


Fig. 1. Overview of the proposed methodology. (a) Initializing the ROS environment and algorithms. (b) Establishing the communication with PX4 FCU. (c) Starting the LFA. (d) Image processing algorithms. (e) UAV starts to perform the mission.

affects the vehicle's speed control, it is necessary to define the coordinates specified by the equations ${}^{e}\mathrm{q}=[{}^{e}\xi\ {}^{e}\eta]^{T}$, where ${}^{e}\xi = [x\ y\ z]^{T} \in \Re^{3}$ corresponds to the longitudinal, lateral and normal displacement in the structure $\langle e \rangle$. The variable ${}^{e}\eta = [\varphi\ \theta\ \psi]^{T} \in \Re^{3}$ corresponds to the angles of *roll*, *pitch* and *yaw* referenced in the space $\langle s \rangle$. The motors are coupled to form a 45-degree angle between the x-y coordinates. Thus, to perform any longitudinal movement, all the motors must act. The vector of forces referring to the UAV that causes the acceleration generated to different axes is represented in Equation 1.

$$F = \begin{bmatrix} fx \\ fy \\ fz \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \sum_{n=1}^{4} f_n \end{bmatrix} \tag{1}$$

The longitudinal forces cancel each other out, causing the vehicle to reach equilibrium in the x-y plane. The force on the z-axis results from the forces exerted by the four engines (Equation 2).

$$f_n = C_f \omega f^{2} \tag{2}$$

where the variable C_f is a constant that depends on parametric constants associated with the propellers and motor coupling region, and the variable ω represents the angular velocity generated by the rotation. The speed is directly controlled by the torques generated by the engines and applied to the vehicle body. Equation 3 represents the torque vector.

$$\tau = \begin{bmatrix} \tau_{\varphi} \\ \tau_{\theta} \\ \tau_{\psi} \end{bmatrix} = \begin{bmatrix} k1 & k1 & -k1 & -k1 \\ -k1 & k1 & k1 & -k1 \\ k2 & -k2 & k2 & -k2 \end{bmatrix} \begin{bmatrix} f1 \\ f2 \\ f3 \\ f4 \end{bmatrix} \tag{3}$$

where $k1$ is the distance between the reference axes and the point where each force is applied (i.e., motor positioning center). $k2$ is the determinant of the relationship between the torque generated by the rotor and

its corresponding impulse. Figure 2 shows the variables that compose the UAV movement.

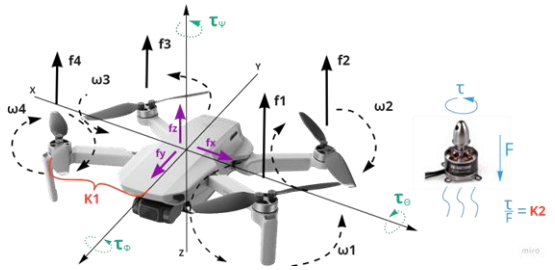


Fig. 2. Exemplified characteristics of axis rotations.

B. Line Follower Controller

Initially, it is necessary to establish the virtualized 3D environment in the Gazebo software [20] and the connection with the simulated UAV controller. The Python library CvBridge allows the reception of the image generated from the *image raw* topic of ROS. This generates compatible image data using the OpenCV package. After this procedure, the image passes through the Canny edge detection. This algorithm performs the image's filtering, locating, and corners, resulting in data to be analyzed during the parameterization through the translated Hough transform for the lines and curves detection [21]. The parameterization specifies a narrow line represented by the angle θ from the normal and delimited by the distance ρ from the origin. Equation 4 stipulates the line and Figure 3 illustrates the sequence of points obtained along the normal line.

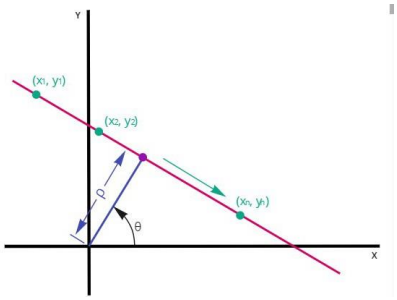


Fig. 3. Sequence of points obtained along the normal line.

$$y = (-\frac{\cos\theta}{\sin\theta})x + (\frac{\rho}{\sin\theta}) \quad (4)$$

where this distance corresponds to the geometry when applied in a way to consolidate a space of n points represented by Equation 5.

$$\rho\theta = X_n \cdot \cos\theta + Y_n \cdot \sin\theta \quad (5)$$

Each pair of polar coordinates ($\rho\theta$, θ) represents a line passing through the point (Xi, Yi). Suppose an interval is selected and observe the plot's application to each point (Xi, Yi). In that case, it is possible to generate a graph representing the family of lines that crosses the indicated point.

Figure 3 proves the possibility of identifying a line by the number of intersections. Note that the greater the number of intersections, the greater and more accurate the probability of having a line in the area of the analyzed image. In the project, we delimited the minimum amount of intersections in the curve so that the code identifies those points as straight, declaring the lines found as the parameters (θ, ρ), that is, the *threshold*. Restricting the obtained data, we ensure the relevant information to certain variables and perform the set of Equations 6 and 7.

$$X_0 = \rho \cdot \cos\theta \quad (6)$$

$$Y_0 = \rho \cdot \cos\theta \quad (7)$$

To receive and store the rounded value of the angle using only the integer result of the equation, and delimiting the start and end points of the lines to be generated, it is possible to obtain the set of Equations 8, 9, 10, and 11.

$$X_1 = X_0 + 1000 \cdot (-\sin\theta) \quad (8)$$

$$Y_1 = Y_0 + 1000 \cdot (-\cos\theta) \quad (9)$$

$$X_2 = X_0 - 1000 \cdot (-\sin\theta) \quad (10)$$

$$Y_1 = Y_0 - 1000 \cdot (-\cos\theta) \quad (11)$$

(X_1, Y_1) and (X_2, Y_2) are the points at which the program draws a line by concatenating them. The code makes markings on the image processed by the camera in order to optimize the values obtained, overwriting the line with one generated directly by the algorithm.

Using the line readings as a basis and delimiting a *threshold* to contain the continuous correction of the quadrotor direction, the angle a (Equation 12) is obtained by correlating the line generated by the code. The axis that divides the image is used to configure messages to be published through the MavROS protocol, converting the readings into modifications in the attributes that directly change the UAV displacement to neutralize a certain deviation. Essentially, it generates a compensation in the *yaw* rotation axis. The last step is the compensation method during the UAV trajectory. A parallel relationship is established with the camera's main axis to have the value obtained from the difference in the coordinates of pixels present between the parallel lines. This configures a perpendicular line that idealizes,

at the moment, in the action of the x-axis rotation to cancel the geographic error.

$$a = -1 \cdot (\arctan(\frac{(Y_2 - Y_1)}{(X_2 - X_1)}) \quad (12)$$

This method of angular fault prioritization avoids the transgression of the path trace reached to compensate for the error in the normal axis when there is a high angle. After the steps of identifying lines and obtaining consistent data, we deal with UAV lag correction by following the route, starting with the application of the start data (X_1, Y_1) and end data (X_2, Y_2) of the lines directly as coordinates in the image generation pixel matrix, overlaying it. Thus, the algorithm developed a modification that compares the last line reading obtained and the current one and an average with these data, enabling the rectification of the algorithm and smoothing of the aerial locomotion of the UAV. Equations 13 and 14 give this process.

$$X_{1n} = \frac{(X_{1n} + X_{1n-1})}{2} \quad (13)$$

$$X_{2n} = \frac{(X_{2n} + X_{2n-1})}{2} \quad (14)$$

III. Results and Discussion

To safely evaluate the proposed strategy, this work have a previous Software In The Loop (SITL) simulation in Gazebo [20] environment before the real tests. The ground station receives the UAV information from a telemetry module using the MAVLink communication protocol. The PX4 software, which runs on the PixHawk flight controller hardware, is responsible for acquiring information from peripheral sensors and modifying these values in the actuators to which it has access. The controller was developed in Python language using ROS packages [22]. The PX4 is an on-board computer with ROS compatible firmware that allows the communication of sensors and motors. The basic ROS inferface and nodes are illustrated in Figure 4. For simulation, Ubuntu 18.04.4 LTS 64-bit operating system was used and has an Intel® Core™ i7-7700HW @ 2.80GHz CPU, Geforce GTX 1050 with 8Gb of DDR4 RAM.

A. Line Follower

With the focus on carrying out tests emphasizing the line follower, a world model with standard flat terrain and a yellow line was used in the software Gazebo (Figure 5 (a)). Figure 5 (b) gives the results of the UAV path compared with the line position. Note that this good result comes from the angle fault prioritization method that avoids the transgression of the path trace. Figure 6 shows the line being detected through the image processing step.

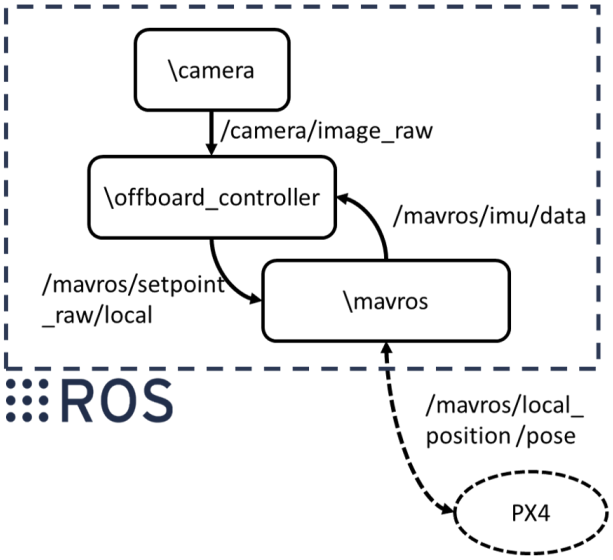
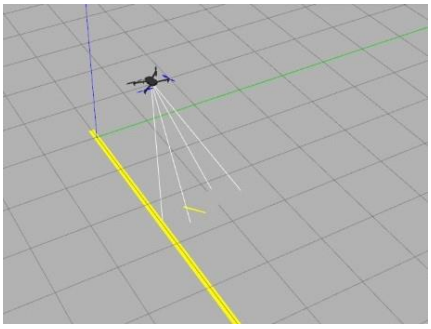
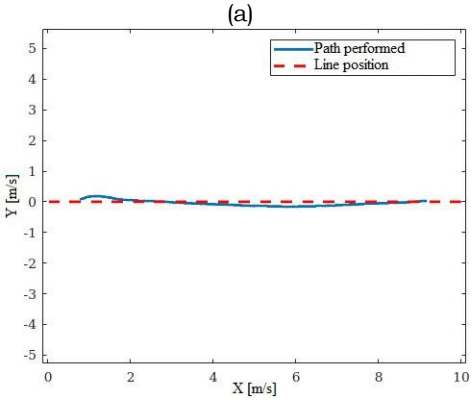


Fig. 4. ROS interfaces.



(a)



(b)

Fig. 5. UAV performing the line follower. (a) Detecting the line. (b) Path result.

B. Pipe Follower Simulation

Another ROS simulation environment was built with the software Gazebo, as shown in Figure 7. This simulation environment comprises an unburied pipeline of O&G industry facilities to demonstrate that the system is ready to deal with the practical operational aspects

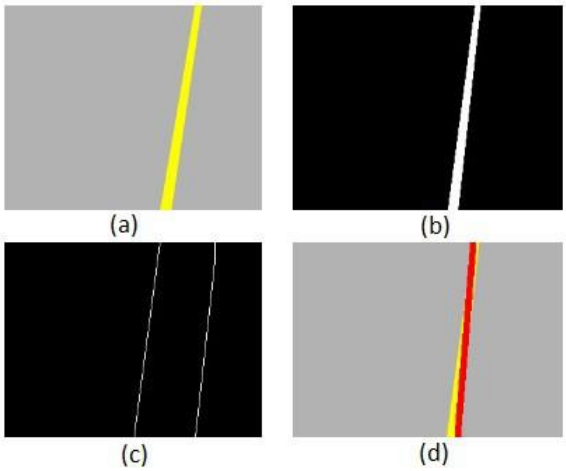


Fig. 6. Image processing steps from a simple line detection. (a) Image Acquired by the UAV. (b) Binary Threshold. (c) Canny Detection. (d) Hough Transform with the angular correction algorithm.

of this kind of environment. For the tests, the same UAV model was used: Iris UAV with PX4 FCU. Figure 8 details the image processing steps results of this SITL simulation. It is possible to verify that this simple image processing technique could adequately generate the path for the UAV, demonstrating the feasibility of testing in a real scenario.

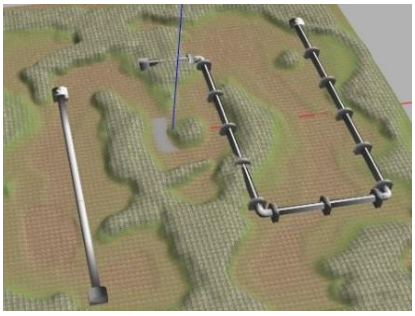


Fig. 7. World created in Gazebo Software.

C. Experimental Results

Due to space and security restrictions, the test on a real UAV was performed in a controlled environment with a fixed height of 10 meters. The FCU used in this experiment is the same as the simulation since the developed system based on MavROS was theoretically ready to be used on a real robotic system with a PixHawk Px4 FCU configured with Autopilot firmware. The UAV used in the real test is the bebop parrot with a PixHawk Px4 with the Autopilot firmware, as shown in Figure 9. The real test method starts with the UAV initializing the controller ROS node on the ground. The controller sends the autonomous takeoff command to the FCU through a native MavROS package. Then, the camera

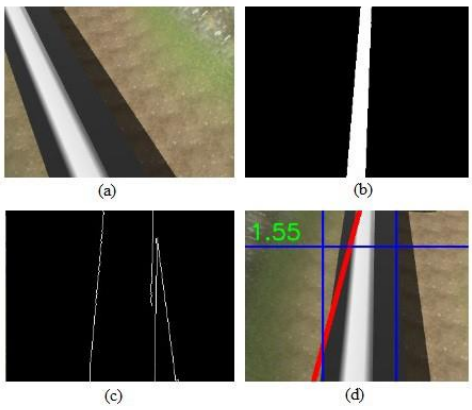


Fig. 8. Image processing steps. (a) Image Acquired by the UAV. (b) Binary Threshold. (c) Canny Detection. (d) Path follower with angle prioritization.

Fig. 9. UAV used in the experimental tests.

node detects the line of the pipes to process the information for navigation. The camera node receives the camera's image through the analog FPV receiver. Note that the velocity commands are parsed to the FCU through MavROS. The controller is designed to fix a flight height, which the FCU tries to maintain autonomously as the velocity on the (x, y) axis referred to UAV's body is modified. Figure 10 presents the results obtained.

Note that depending on the applications, the obtained path-line may be inaccurate due to inappropriate lighting. However, the UAV was positioned at a fixed height of ten meters due to safety restrictions, and, in this height, the lighting condition did not harm the process. The obtained results demonstrated the technical feasibility insight, showing that it is possible to apply such technology for autonomous inspection of unburied pipeline structures in real-time facilities, improving operational safety e reducing costs. A video of the proposed strategy working and all the coding is presented in the [1].

IV. Conclusions and Future Work

The SITL simulation and the practical experimentation succeeded in accomplishing the proposed mission, where the UAV could fly autonomously over all the pipelines. In terms of evaluation, this work opens up several future works. For instance, disturbance effects

[1]https://github.com/LucasSousaENG/Pipeline-Inspection/

Fig. 10. Experimental results obtained with Bebop Parrot. (a) Image Acquired by the UAV. (b) Hough Transform. (c) Line Follower.

on flight behavior must be analyzed before testing it in the real proposed environment. Another extension is foreseen for this research work, such as the economic gain of applying the proposed methodology and all the costs and risks associated with it in this proposed scenario.

ACKNOWLEDGMENT

The authors would like to thank the following Brazilian Agencies CEFET-RJ, CAPES, CNPq, and FAPERJ. Besides, the authors also want to thank the Research Centre in Digitalization and Intelligent Robotics (CeDRI), IPB, Laboratório para a Sustentabilidade e Tecnologia em Regiões de Montanha (SusTEC), IPB, Portugal, and INESC Technology and Science, Porto, Portugal.

REFERENCES

[1] A. G. Melo, M. F. Pinto, L. M. Honorio, F. M. Dias, and J. E. Masson, "3d correspondence and point projection method for structures deformation analysis," *IEEE Access*, vol. 8, pp. 177823–177836, 2020.

[2] M. F. Pinto, L. M. Honorio, A. Melo, and A. L. Marcato, "A robotic cognitive architecture for slope and dam inspections," *Sensors*, vol. 20, no. 16, p. 4579, 2020.

[3] M. F. Pinto, L. M. Hono´rio, A. L. Marcato, M. A. Dantas, A. G. Melo, M. Capretz, and C. Urdiales, "Arcog: An aerial robotics cognitive architecture," *Robotica*, pp. 1–20, 2020.

[4] M. F. Pinto, A. G. Melo, A. L. Marcato, and C. Urdiales, "Case-based reasoning approach applied to surveillance system using an autonomous unmanned aerial vehicle," in *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, pp. 1324–1329, IEEE, 2017.

[5] M. F. Pinto, F. O. Coelho, J. P. De Souza, A. G. Melo, A. L. Marcato, and C. Urdiales, "Ekf design for online trajectory prediction of a moving object detected onboard of a uav," in *2018 13th APCA International Conference on Automatic Control and Soft Computing (CONTROLO)*, pp. 407–412, IEEE, 2018.

[6] A. G. Melo, M. F. Pinto, A. L. Marcato, L. M. Hono´rio, and F. O. Coelho, "Dynamic optimization and heuristics based online coverage path planning in 3d environment for uavs," *Sensors*, vol. 21, no. 4, p. 1108, 2021.

[7] I. Z. Biundini, A. G. Melo, M. F. Pinto, G. M. Marins, A. L. Marcato, and L. M. Honorio, "Coverage path planning optimization for slopes and dams inspection," in *Iberian Robotics conference*, pp. 513–523, Springer, 2019.

[8] N. Dijkshoorn and A. Visser, "Integrating sensor and motion models to localize an autonomous ar. drone," *International Journal of Micro Air Vehicles*, vol. 3, no. 4, pp. 183–200, 2011.

[9] G. S. Ramos, M. F. Pinto, F. O. Coelho, L. M. Hono´rio, and D. B. Haddad, "Hybrid methodology based on computational vision and sensor fusion for assisting autonomous uav on offshore messenger cable transfer operation," *Robotica*, pp. 1–29, 2022.

[10] F. O. Coelho, M. F. Pinto, J. P. C. Souza, and A. L. Marcato, "Hybrid methodology for path planning and computational vision applied to autonomous mission: A new approach," *Robotica*, vol. 38, pp. 1000–1018, 2020.

[11] E. Kakaletsis, C. Symeonidis, M. Tzelepi, I. Mademlis, A. Tefas, N. Nikolaidis, and I. Pitas, "Computer vision for autonomous uav flight safety: An overview and a vision-based safe landing pipeline example," *ACM Computing Surveys (CSUR)*, vol. 54, no. 9, pp. 1–37, 2021.

[12] N. Al Said, Y. Gorbachev, and A. Avdeenko, "An unmanned aerial vehicles navigation system on the basis of pattern recognition applications—review of implementation options and prospects for development," *Software: Practice and Experience*, vol. 51, no. 7, pp. 1509–1517, 2021.

[13] B. Hadi, A. Khosravi, and P. Sarhadi, "A review of the path planning and formation control for multiple autonomous underwater vehicles," *Journal of Intelligent & Robotic Systems*, vol. 101, no. 4, pp. 1–26, 2021.

[14] G. M. Maciel, I. Z. Biundini, M. F. Pinto, I. C. Junior, A. L. Marcato, G. Adolfo Filho, C. F. d. E. T. Celso, and S. da Fonseca, "Design of a low cost four-channel ldr based line-follower sensor with transient and external interference compensations,"

[15] G.-Q. Jiang, C.-J. Zhao, and Y.-S. Si, "A machine vision based crop rows detection for agricultural robots," in *2010 International Conference on Wavelet Analysis and Pattern Recognition*, pp. 114–118, IEEE, 2010.

[16] M. Basso and E. Pignaton de Freitas, "A uav guidance system using crop row detection and line follower algorithms," *Journal of Intelligent & Robotic Systems*, vol. 97, no. 3, pp. 605–621, 2020.

[17] V. Mullenders, "Design of a motion monitoring system for unmanned offshore topside installation based on real-time visual object tracking using drones and fixed cameras on an sscv," *Delft University of Technology*, 2019.

[18] A. P. I. API, "Api guide for developing an unmanned aircraft systems program," 2019.

[19] A. S. Brandao, F. N. Martins, and H. B. Soneguetti, "A vision-based line following strategy for an autonomous uav," in *2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, vol. 2, pp. 314–319, IEEE, 2015.

[20] Gazebo, "Gazebo robot simulator." http://gazebosim.org/, 2021. Acessado em 26/02/2021.

[21] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[22] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, p. 5, Kobe, Japan, 2009.