1st International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI' 2019),
20-22 March 2019, Barcelona, Spain

Oral

Topic: *Intelligent System Architectures*

AirCSI – Remotely Criminal Investigator

P. Araújo Jr., M. Mendonça and L. Oliveira
Intelligent Vision Research Lab
Federal University of Bahia, Brazil
Tel.: +55 71 3283-9472
E-mails: pompilio.araujo@ufba.br, lrebouca@ufba.br

Summary: Once a location associated with a committed crime must be preserved, even before criminal experts start collecting and analyzing evidences, the crime scene should be recorded with minimal human interference. In this work, we introduce an autonomous system for investigation of crime scene using a drone. Our proposed intelligent system recognizes objects considered as important evidence of the crime scene, and defines the trajectories through which the drone performs a detailed search to record evidences of the scene. We used our own method, called Air-SSLAM, to estimate drone's pose, as well as proportional–integral–derivative (PID) controllers for aircraft stabilization, while flying through the paths defined by the environment recognition step. We evaluated the performance of our system in a simulator, also preparing a real-drone system to work in a real environment.

Keywords: criminal scene investigation, object detection, intelligent drones, SLAM, autonomous drone, self localization, object detect.

1. Introduction

When a crime is committed, no matter how careful the criminal is, evidences are spread throughout the crime scene, and must be recorded and collected by a team of experts. Evidences are not perennial, decreasing in quantity and quality as they spatially and temporally come far from the crime scene. The goal of collecting and recording evidences is to preserve the maximum amount of information so that experts, prosecutors and judges can analyze the dynamics of the facts, deciding in the courts on the culpability of those ones involved [1]. The gathering and recording of evidences are tasks of the criminal expert, who is designated by the state to analyze the site, as well as any material evidence that may clarifies the crime. The expert must use all technological resources available to store the evidence, since the fragile elements can be lost after releasing from the crime scene.

Very few works propose methods of autonomous drones to search for crime evidence. In [2], the authors make a comparison between the ways of acquiring data from the scene with laser and with images. The goal is to compare traditional methods using theodolite to geo-refer the evidence. The data collected through these 3D techniques are free of time-consuming problems and can be used at any time, while sharing between different operators. The authors propose two case studies, discussing practical aspects of data acquisition from a crime scene. However, authors perform data acquisition, manually. An environmental data acquisition system is presented in a police context [3], specifically in external environments focused on environmental crimes. Some aspects of data captured with aerial images are used, but there is no calculation of trajectory and autonomous flight. A detection of objects with images captured in a drone with an NVIDIA Jetson TX2 module for GPU processing on board the aircraft is addressed in [4]. In [5], you only look once (YOLO) method is used to detect objects, achieving the bet performance at that time.

Here we introduce AirCSI, an image collection and recording system of crime scene evidence, which autonomously operates onboard in an unmanned aerial vehicle (drone). The AirCSI uses a stereo camera installed on the drone, which has the goal of capturing the images for the aircraft positioning system in real time with our AirSSLAM, a simultaneous localization and mapping method [6]. A downward-facing monocular camera, also equipping the drone, is used to detect and help estimate the object coordinates at the crime scene. Although AirCSI can use any object detector as a baseline to find the evidences, in our experiments, we used YOLO-v3 [7], which was specially trained for our proposed goals. The YOLO was chosen because it uses a single CNN network to

classify and locate the object, which provides faster detection.

AirCSI brings three important contributions, as follows: (i) The introduction of a method to calculate trajectories for finding objects, considering the objects already found in a crime scene, (ii) a new methodology to increase the accuracy of object detection based on multiple perspectives, and, finally, (iii) an evaluation in a realistic environment using the AirSim simulator [8].

2. Outline of the AirCSI

Air-CSI creates a coordinate system that originates from the drone starting point. Figure 1 summarizes our proposed method described in five steps, as follows:

(1) The drone initiates the movement in the vertical direction and stabilizes at the height $h < hmax$;
(2) Using the monocular camera at the bottom of the drone, each detected object is classified as a type of evidence, which has a relevance coefficient ρ defined by the user;
(3) Trajectory calculation is performed according to the coefficient ρ of the detected evidences; a coverage radius is created for each detected evidence and the drone should pass through the coverage area of all evidences;
(4) The control module performs stabilization and displacement of the aircraft in the trajectory defined by the system; there are eight proportional–integral–derivative (PID) controllers: two cascades in each direction of the quadrotor drone movements, one for velocity, and another one for the position;
(5) From the data collected, AirCSI creates reports with sketches, highlighted evidence and images gathered during the scanning.

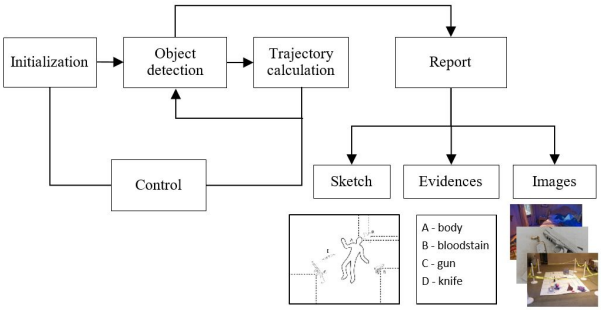

Fig. 1. Initialization - the drone takes off from a position near the crime scene, being positioned at a height **h**; **object detection** - the camera at bottom captures images and detects suspicious objects; **trajectory calculation** - a trajectory is calculated from the positioning of the detected objects, and the drone performs the trajectory for refined object detection; **report** - the result of the scan is presented in a report.

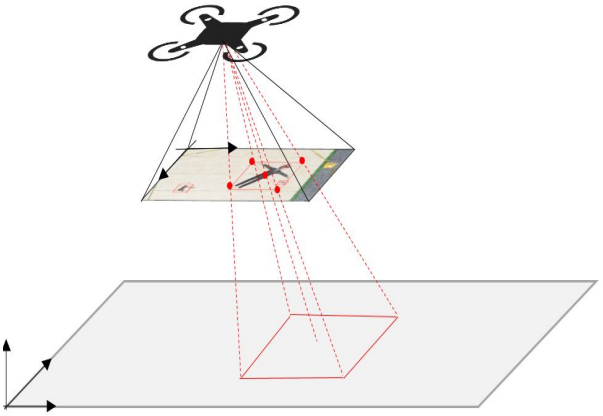

Fig. 2. The five points of the bounding box of the monocular camera image are translated to the world coordinate system by multiplying the target vector by the inverse of the pose matrix.

2.1. Evidence detection

YOLO was used as a baseline detector. Although this detection method is not the most accurate, it is one of the fastest. YOLO was also the best choice, since precision has less relevance than the detection rate. This is so because the object will be detected from more than one perspective, and only objects that has their detections confirmed in all perspectives will be considered. In other words, after the first detection, object position is recorded, demanding the drone to detect the object again, in a different pose. This situation makes the object detection module to have higher mAP as the drone approaches to the object.

YOLO applies a single neural network to every whole image. The network divides the image into regions and provides bounding boxes and probabilities for each region. The bounding boxes are weighted by the predicted probabilities. YOLO was trained for the detection of the following objects: human body, revolver, pistol, machine gun and knife. The following parameters were used to train the object detector: batch = 64, momentum = 0.9 and decay = 0.0005. Images were preprocessed by changing their resolution to 608 x 608 from the original images acquired. To train the YOLO detector, the MS-COCO dataset with 3000 additional weapon images were used.

After detecting the objects in the scene, the object bounding box is projected onto the ground plane, providing two dimensional information of the object location (see Fig. 2). The stereo camera is used to estimate the distance between the drone and the evidence, as well as the drone and the ground. These two distances provides also an estimation of the object height.

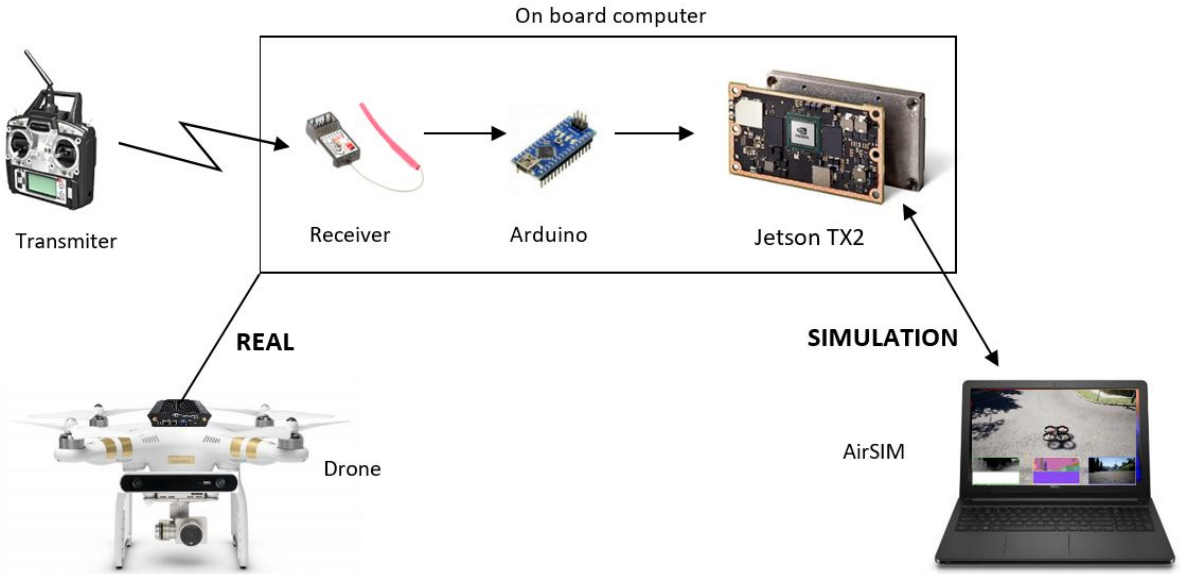

Fig. 3. Simulation tests - AirCSI runs within an NVIDIA Jetson TX2 module. Another computer runs the AirSim program, which transmits the pose to drone on board computer. Real projected situation - AirSSLAM and AirCSI run within an NVIDIA Jetson TX2 module. A transmitter and receiver as well as an Arduino are used for manual control.

Our proposed system considers six degrees of freedom that determines the pose of the drone $[x\ y\ z\ \varphi\ \chi\ \psi]^T$, where x, y and z are the coordinates of the drone position, and ψ is the yaw rotation. The angles φ and χ (roll and pitch) are considered null values when the drone is in equilibrium, while these values are very low during the drone movement. These constraints are completely suitable, because the drone moves at very low velocity.

To internally represent a detected evidence, five points of the detected bounding boxes (the four vertices and the geometric center of the rectangle) are used. Each point (p_i) in the bounding box is defined with the coordinates $X_{Ci} = [x_i\ y_i\ z_i]^T$ with respect to the camera coordinate system. Using this camera pose P, points are translated to the world coordinate system X_{Wi}.

$$X_{Ci} = P^{-1} \cdot X_{Wi} \tag{1}$$

Each time an evidence is found, its position is stored and a counter is incremented. At the end of the scan, each evidence will have recorded the number of times it was detected. The accuracy of the deviation is proportional to the value recorded by the counter, because drone overlaps the evidence many times in different perspectives.

3. Self localization

To perform the control of the drone pose in a real situation, a GPS-independent, self-locating method is exploited by using only images from the stereo camera (refer to [2] for more details on AirSSLAM). AirSSLAM relies on good features to-track (GFTT) [9] to extract keypoints, which are lately described by rotated-binary robust-independent elementary features (rBRIEF) [10],[11]. To estimate drone pose, the keypoints of the two views are matched in order to calculate the transformation matrix between two consecutive timed frames. The keypoints provide an initial map that is used as a reference to be tracked posteriorly. Air-SSLAM performs a periodic map maintenance around image patches, which are also used as quality indicators to improve keypoint tracking. An optimization procedure algorithm is applied to include new keypoints in the continuously updated map. This procedure is necessary to minimize the error between the current keypoint and the map (already inserted) keypoints. After that, the pose of the drone is periodically recalculated by a bundle adjustment optimization method. All map keypoints and the drone pose calculated are used to perform this refinement. Air-SSLAM presents a novel method of point matching, starting the search at a probable point location, then gradually increasing the search area. As each keypoint is found, the probable location for the subsequent points are updated and corrected. Air-SSLAM applies a Kalman filter to stabilize the calculated camera pose. This method allows real-time location of the drone in environment.

In the simulator, AirSSLAM is not used. Another computer runs the AirSim program, which transmits the pose to the drone onboard computer (NVIDIA Jetson TX2 [12]). AirSim is a simulator created on the Unreal Engine that offers physically and visually realistic

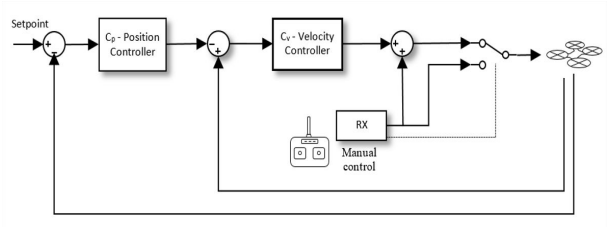

Fig. 4. Control system: the control in each direction is made by two controllers: C_V (velocity) and C_P (position). The output of the velocity controller goes into a switch that alternate the control from manual to automatic.

simulations designed to operate on high frequency real-time looping hardware simulations. The AirSim was experimentally tested with a quadrotor as a stand-alone vehicle, comparing the software components with real-world flights. A change was made in the original code of the *reportState* method [12], which was carried out to transmit the drone pose, via network, with the UDP protocol in order to have faster transmission boudrates. This communication channel is also used to stream the controller commands to the simulator. Figures 3 illustrates the configuration to perform the simulation and how it was designed to work in real situation.

4. Controlling the drone

A double control is applied in each direction of the drone coordinates, [x y z ψ]. For each variable, we used two controllers, one for velocity and another for position. Disturbances influencing the position are corrected by the first controller, which does not allow a great interference on the velocity (see Fig. 4) [13]. In addition, the phase delay in the secondary part of the process is measurably reduced by the secondary loop. This improves the response of the velocity in the primary mesh [14].

The input of the controller C_V is the velocity error $e_{\dot{x}_c}$, given by

$$e_{\dot{x}_c} = \left(\frac{x_{c(n)} - x_{c(n-1)}}{T}\right) - P^{-1}\dot{x}_{ws}\,, \qquad (2)$$

where $x_{c(n)}$ and $x_{c(n-1)}$ is the position of the drone in the camera coordinate system in the current and previous samplings, respectively; T is the sampling period, P is the drone pose matrix and $\dot{x}_{ws}$ is the reference velocity in the global coordinate system that is received from the position controller output. The input of the controller C_P is the position error e_{x_w} , which is defined by

$$e_{X_w} = X_w - X_{ws}\,, \qquad (3)$$

where X_w is the position of the current drone and X_{ws} is the desired position.

The PID controllers are used by the transfer function:

$$u(t) = K_p e(t) + K_i \int e(t)\,dt + K_d \frac{de}{dt} \qquad (4)$$

where K_p, K_i, K_d are the proportional, derivative integral constants, respectively. The method 2p2z was implemented with sampling period of 160ms. The output is given by:

$$y[n] = e[n]\,b_0 + e[n-1]\,b_1 + e[n-2]\,b_2 + y[n-1] \quad (5)$$

where y[n] is the control signal at the output of the controller, and e[n] is the error in the controlled variable (position or velocity). The constants b_0, b_1 and b_2 are:

$$\begin{aligned} b_0 &= K_p + \frac{K_i \cdot T}{2} + \frac{K_d}{T} \\ b_1 &= K_p + \frac{K_i \cdot T}{2} - \frac{2 \cdot K_d}{T} \\ b_2 &= \frac{K_d}{T} \end{aligned} \qquad (6)$$

The controllers were tuned by the Ziegler-Nichols closed-loop method. The values of the constants of the velocity and position controllers are listed in Table 1.

Table 1. Coefficients of the controllers used in the tests

	x		***y***		***z***		**ψ**	
	Cp	***Cv***	***Cp***	***Cv***	***Cp***	***Cv***	***Cp***	***Cv***
Kp	*5*	*30*	*5*	*30*	*10*	*40*	*10*	*20*
Ki	*1*	*2*	*1*	*2*	*2*	*2*	*5*	*5*
Kd	*4*	*3*	*4*	*3*	*3*	*3*	*0.08*	*0.01*

In our experiment, a remote control is used to manually control the drone during the tuning phase, in order to avoid accidents. A switch key was implemented to switch from manual to automatic control. A routine was implemented to keep the drone in the current position whenever the key is triggered. This allows the user to set an initial pose for the drone manually.

Figure 6 illustrates the result of the tests with the controllers running in the simulator. The controllers were evaluated by adjusting the set point with a variation of eight meters in each direction x, y and z, and a variation of 90º in the angle yaw (ψ). An average accommodation time of 20 ms in all directions was obtained. With respect to the angle yaw (ψ), the accommodation time was 10s. This time can be considered as satisfactory, because a small value of velocity is needed to give time to perform the detection of objects. A large overshoot was observed only in the x direction.

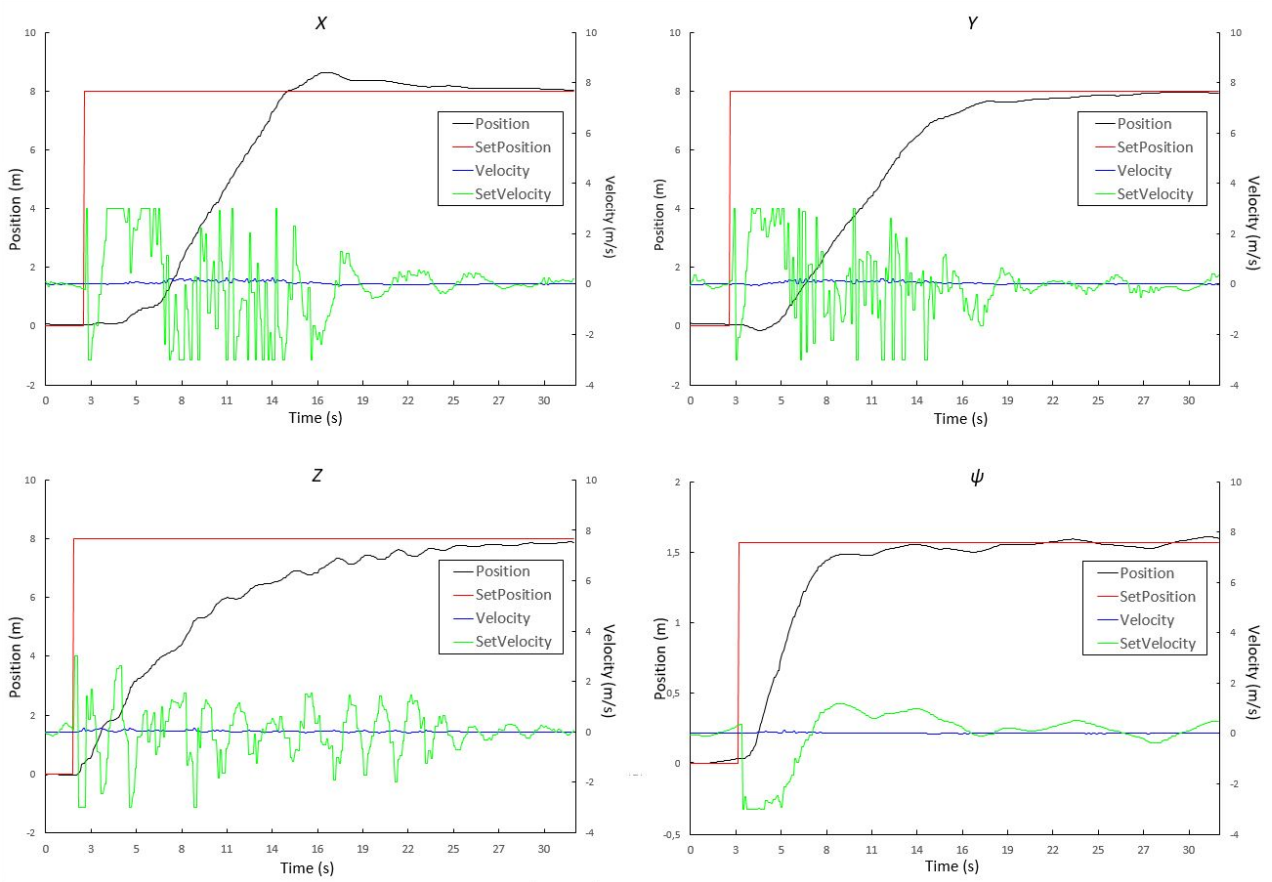

Fig. 5. The plots show the position and velocity variation over a period of 30 seconds. The average accommodation time of 20 seconds was observed in the directions x, y, x, and 10 seconds in the angle ψ.

The main reason may not have to do with the system itself, but to some delay in transmitting the position over the ethernet network. This hypothesis was confirmed when the tests were repeated, and the problem did not occur all the time.

To stabilize the calculated camera pose, a Kalman filter was applied only on the position controllers, using the following parameters: number of states = 2, measure states = 1, and time of measurement = 0.500 s.

4. Trajectory calculation

After detecting the first evidence from a height $\boldsymbol{h}$, AirCSI flies down to perform a detailed search for more evidences. Each type of evidence has a span radius value ρ_i that defines a scan area. In the tests performed, we used the following radius for some evidence samples: human body (ρ_1 = 3 m), revolver (ρ_2 = 2 m), pistol (ρ_3 = 2 m), machine gun (ρ_4 = 2 m) and knife (ρ_5 = 1 m).

The scan is performed in-line following a path that fills the rectangle R that circumscribes the circle of radius ρ_i (see Fig. 6). That rectangle is calculated using the method described in [15]. This method builds a rectangle of minimum area enclosing an n-vertex convex polygon. To define the in-line scan, the drone moves following a zig-zag path, according to a sequence of points, which is defined as follows: Let V_1, V_2, V_3 and V_4 be the vertices of the rectangle circumscribing the circles, h the height of the drone during scanning and θ the horizontal aperture angle of the camera, the trajectory follows the points T_i as follows

$$T_i = V_t + \left(\frac{1}{2} + j\right) \bullet \left(\frac{V_{t+2} - V_t}{\| V_{t+2} - V_t \|}\right) \cdot h \bullet \tan tan \frac{\theta}{2}, \quad (7)$$

where i is the index of each point of the trajectory j is:

$$\begin{aligned} j &= (i-1) \ div \ 2 \\ t &= 1 + [(i-1) \ div \ 2] \ mod \ 2 \end{aligned} \quad (8)$$

With this trajectory the drone manages to sweep all areas close to the evidence, guaranteeing that there would be no point without going through the vision of the camera. The example shown in Fig. 7 shows a trajectory of a crime scene, where four evidences were found: a human body, a revolver, a knife and a machine gun.

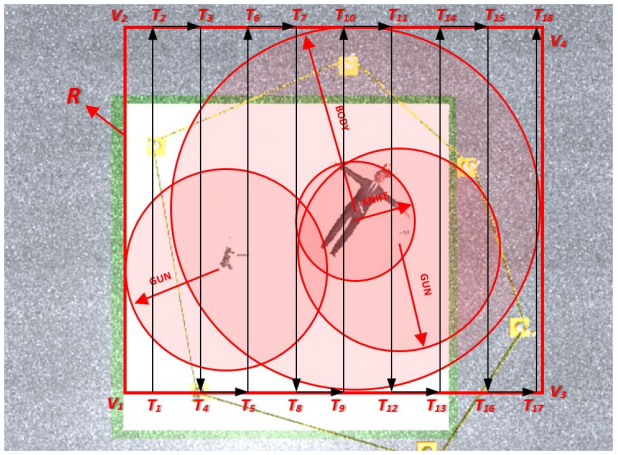

Fig. 6. Drone trajectory calculated as a function of the weight coefficient ρ_i for each evidence. The human body is more relevant than other evidences, so it has a larger scanning area.

5. Conclusion

AirCSI uses a drone to sweep an area that contains evidence of a crime. The proposed system uses YOLO as a real-time object detector to find evidences in a scene. In search for crime evidences, a low false negative value is wanted, since a human analysis will always be done by an expert after the automatic search. The system control is designed for the drone to move steadily. The cascaded controllers allowed an explicit velocity adjustment, which did not allow very fast scrolling to impair the capture of images. Thus, the found accommodation time of 20s can be considered suitable for scanning the tested distance of 8m. As a future work, we are working on a real implementation of AirCSI in a drone. Also, the goal of our research is to training a detector to search for other classes of evidence such as ammunition cases, projectiles, bloodstains and other types of objects used in crime.

Acknowledgements

The authors would like to thank FUNDAÇÃO DE AUXÍLIO À PESQUISA DO ESTADO DA BAHIA (FAPESB) to support with the project under grant APP0015/2016, and the Federal Police of Brazil, for the time granted to the project.

References

[1] Fish, J. T., Miller, L. S., Braswell, M. C., & Wallace Jr, E. W. Crime scene investigation. *Routledge*. 2003.

[2] Bucheli, S. R., Pan, Z., Glennie, C. L., Lynne, A. M., Haarman, D. P., & Hill, J. M. (2014). Terrestrial laser scanning to model sunlight irradiance on cadavers under conditions of natural decomposition. International journal of legal medicine, 128(4), 725-732.

[3] Lega, M., Ferrara, C., Persechino, G., & Bishop, P. (2014). Remote sensing in environmental police investigations: aerial platforms and an innovative application of thermography to detect several illegal activities. Environmental monitoring and assessment, 186(12), 8291-8301.

[4] Tijtgat, N., Van Ranst, W., Volckaert, B., Goedemé, T., & De Turck, F. (2017). Embedded real-time object detection for a UAV warning system. In ICCV2017, the International Conference on Computer Vision (pp. 2110-2118).

[5] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

[6] Araújo, P., Miranda, R., Carmo, D., Alves, R., & Oliveira, L. (2017). Air-SSLAM: A visual stereo indoor slam for aerial quadrotors IEEE *Geoscience and Remote Sensing Letters*, 14, 1643–1647.

[7] Redmon, J., & Farhadi, A. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*. 2018.

[8] Shah, S., Dey, D., Lovett, C., & Kapoor, A. (2017). Airsim: High-230 fidelity visual and physical simulation for autonomous vehicles. *In Field and Service Robotics*. https://arxiv.org/abs/1705.05065.arXiv:arXiv:1705.05066.

[9] J. Shi and C. Tomasi, "Good features to track," in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 1994, pp. 593–600.

[10] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in Intl. Conf. on Comput. Vision (ICCV), 2011, pp. 2564–2571.

[11] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in Europ. Conf. on Comput. Vision (ECCV), 2010, pp. 778–792.

[12] "NVIDIA Jetson TX2," http://www.nvidia.com/object/ embedded-systems-dev-kits-modules.html.

[13] Saha, P. K. Process Control and Instrumentation-Web course.

[14] Marlin, T. E., & Marlin, T. E. (1995). Process control: designing processes and control systems for dynamic performance (Vol. 2). New York: McGraw-Hill.

[15] Arnon, D. S., & Gieselmann, J. P. (1983). A linear time algorithm for the minimum area rectangle enclosing a convex polygon.