Towards 3D printed modular unmanned aerial vehicle development: The landing safety paradigm

Tilemahos Mitroudas, Konstantinos A. Tsintotas, Nicholas Santavas, Athanasios Psomoulis, and Antonios Gasteratos

2022 IEEE International Conference on Imaging Systems and Techniques (IST) | 978-1-6654-8102-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IST55454.2022.9827665

Abstract— **Unmanned aerial vehicles (UAVs) are at the forefront of this century's technological shift, becoming ubiquitous in research and market areas. Similarly, nowadays, 3D printing is a fast-emerging, widely used technology that allows individuals to design prototypes that fulfil their needs. This paper presents an autonomous UAV designed and implemented to be fully modular and 3D printable. Furthermore, suitable areas for landing are recognized using a lightweight deep learning architecture while a Gazebo model for simulation purposes is also given to the research community. Finally, its fly and surface recognition processes are evaluated exhaustively in real-world and simulation scenarios.**

I. Introduction

Small autonomous unnamed aerial vehicles (UAVs) have become very popular in the last decade [1]. The ones based on quadrotors, *i.e.,* small agile UAVs controlled by the rotational speed of their four rotors, constitute a popular category. Their vertical take-off and landing ability as well as their agility and precise movement differentiate them from other aerial platforms. Their rotors' arrangement to the coordinate system of the aircraft's body discriminates them into two different types, namely the "x" and the "+" configurations [2].

UAVs require lightweight materials with great mechanical strength. Unfortunately, polymers such as polylactic acid (PLA) and glycol-modified version of polyethylene terephthalate (PETG), which are the primary materials used in fused deposition modeling (FDM) [3], do not exhibit such properties [4]. Nevertheless, the FDM manufacturing processes, such as 3D printing, is a technology that has made prototyping more accessible [5], [6] and has seen tremendous growth the previous years [7]. This is because lightweight structures with complex inner features can be produced directly without a mould need, giving additive manufacturing (AM) an advantage over conventional manufacturing [8]. This fact permitted the development of many quadrotor implementations through open-source designs [9]. For example, a scriptable way of producing small quadrotors quickly and efficiently using an origami-inspired method was introduced in [10], where precision cut sheets of plastic film were folded to achieve desired geometries. At the same time, the authors in [11] proposed a way to produce a small, lightweight, and cheap platform on a single printed circuit board (PCB) aiming for robotic research [11].

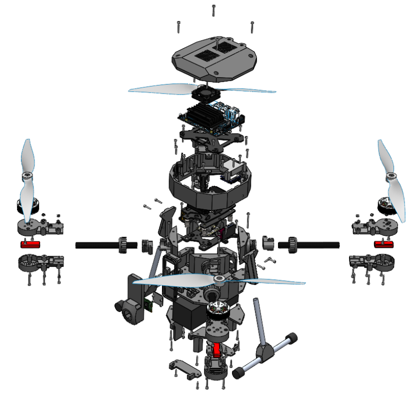

Fig. 1: A view of the quadrotor during the design phase.

Due to the wide use of quadrotors in applications [12], such as surveillance [13]–[15], search and rescue operations [16], [17], inspection [18], [19], mapping [20]–[23], and media production [24], flight safety is of the highest importance in autonomous navigation because of the possible hazards to people or disasters to their equipment in cases of malfunction [25]. A crowd detection method based on light and fast convolutional neural networks (CNNs) [26], [27] is demonstrated in [28]. Similarly, a novel pipeline that performs cue extraction with algorithms based on the structure and functionality of the retina and the visual cortex of the mammalian visual system is introduced in [29]. A complete UAV flight and landing safety pipeline robot operating system (ROS)-based implementation is proposed in [30]. The quadrotor's mission comprises three phases: take-off, trajectory tracking, and precise landing, nevertheless, the latter is among the most challenging tasks when the global navigation satellite system (GNSS) information is missing [31].

This paper presents the development of a 3D printed modular aerial quadrotor platform that can be easily extended and customized to each application's needs by choosing different onboard computers, cameras, sensors, and even different types of motors' electronic speed controls (ESCs) and flight computers. Our system is built from scratch through 3D printing for quick and easy part creation from the ground up (see Fig. 1). PETG is used as the construction material since it combines good mechanical properties, and significant temperature resistance, while it is recyclable [32]. Furthermore, a surface recognition algorithm for safety landing is proposed aiming to protect the system from circumstances of

Authors are with the Department of Production and Management Engineering, Democritus University of Thrace, 12 Vas. Sophias, GR-671 32, Xanthi, Greece {tilemitr, ktsintot, nsantava, apsomoul, agaster}@pme.duth.gr

978-1-6654-8102-1/22/$31.00 ©2022 IEEE

lost GNSS signal. As a final note, aiming to help the robotics community, we design and port our aircraft's model to the Gazebo simulator for easier integration within ROS.

The main contributions proposed by the paper at hand are summarized as follows:

- An open-source, fully 3D printable quadrotor construction.
- A low complexity and lightweight vision-based surface recognition system, which detects people and avoids crowded areas.
- A quadrotor integration with ROS by porting the implemented model to the Gazebo environment. The simulated aircraft is designed to be as close to the actual one, including the same set of sensors and flight characteristics. This way, new applications can easily be simulated and experimented with before real-world scenarios.

The structure of this work is as follows. Section II introduces the system's design and the model's port to Gazebo and thus ROS. Section III describes the 3D printed UAV construction, while Section IV presents the surface recognition module. Our experimental protocol, both in gazebo and actual quadrotor, is given in Section V. Lastly, conclusions and future work come in Section VI.

II. Quadcopter framework

The quadcopter layout constitutes the most suitable one for our approach due to its simple flight properties [33]. The combination of even weight distribution and the possible slow flight speed facilitate the design process as no substantial computational fluid dynamics (CFD) analysis is required. A fully 3D printed modular quadcopter airframe presents many advantages; more specifically, its structure proves to be very practical in the flowing cases:

- Design revisions do not require the whole airframe to be re-printed during the UAV development.
- The UAV configuration can be easily modified without the need for radical changes in its design layout.
- The damage is confined to a minimum when an accident occurs as only the affected parts call for repair.

Due to the reasons mentioned above, the cost needed for restoring the UAV and the MTTR (MTTR stands for mean time to recovery) is the least possible. Therefore the design can be easily replicated, and the affected-systems can be easily and quickly replaced.

A. *UAV's design & architecture*

Apart from the airframe, the general system architecture of a quadcopter is standard [34]. The basic building blocks comprise the flight controller, the onboard computer, the power distribution module including batteries, the payload consisting of the sensors needed for the UAV's mission, and the thrust system, which consists of the ESCs, the brush-less electric motors, and their propellers.

A Pixhawk 1.8.2 (fmuv3) is selected as a flight controller for the proposed system due to its low cost and variety of features. Concerning its software, it is flashed with PX4, which is an autopilot running on top of the real-time operating system (RTOS) NuttX. Furthermore, it provides access to the aircraft's parameters, while much support exists since it is widely adopted from the research and individual communities [35]. As a final note, MAVLink, a communication protocol used to monitor and control UAVs through an onboard computer, constitutes the last component of our architecture.

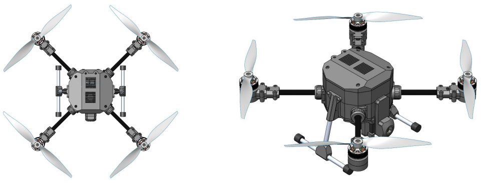

Fig. 2: A Solidworks view of our fully assembled quadrotor.

Next, an Nvidia Jetson nano equipped with ROS is chosen as an onboard computer, while the PX4 to ROS communication is achieved via MAVROS, an extendable node with ROS that converts MAVLink messages to ROS topics [36]. The rest of the components, such as the thrust and power distribution systems, are not critical components as the former should lift the quadcopter and maintain its altitude at 50% of throttle input, while the latter is responsible for providing the suitable voltages reliably to each subsystem. Finally, our platform perceives the environment using a conventional Raspberry Pi's IMX219 rolling-shutter camera.

The main guidelines during the design phase of the quadrotor are the following:

- Even weight distribution of rendering the platform as passively stable as possible.
- A robust aircraft structure capable of withstanding harsh landings.
- The flight controller is mounted on vibration isolation dampeners.
- The enclosed components are protected from dust and other external factors.

Similarly, we considered the following FDM's limitations:

- Parts need to be designed in such way that big overhangs are avoided.
- The maximum area of printed parts is within our print volume limits.

The final design of the proposed quadrotor is illustrated in Fig. 2 while an exploded view, where the majority of the components are visible, can be found in Fig. 1.

B. *Gazebo simulator model*

As robot simulators allow to design, simulate, and test robotic applications in relevant physical environments independently of the availability of actual hardware and, at the same time, permit time-saving during development and cost, we modeled our quadrotor for integration into the Gazebo

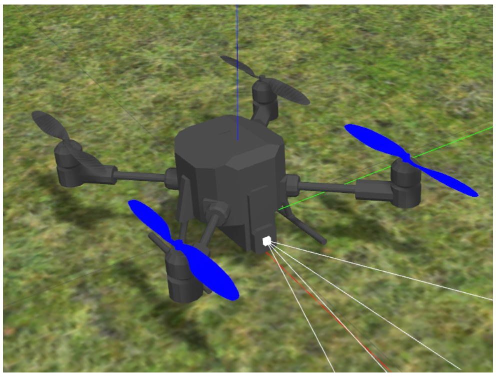

Fig. 3: The Gazebo model of our 3D printed modular unmanned aerial vehicle (UAV) quadrotor as illustrated in the simulated environment.

environment. As a result, a compatible model[1] is generated with the quadrotor's 3D system and data such as weight, dimensions, and thrust characteristics. At first, the .dae or .stl files from the urdf Solidworks plug-in are exported, while the platform's characteristics, *e.g.,* collision, visual-inertial sensors, in the .sdf file are configured subsequently. We are based on the IRIS, a quadcopter pre-installed in Gazebo, and aiming to accomplish a simulation as close to the real-world. Therefore, we calculate the thrust characteristics of the quadcopter. In particular, *libgazebo_motor_model.so* is the plugin responsible for simulating the motor-propeller system, while the critical parameters that need adjustment are the *maxRotVelocity¿*, *motorConstant*, and *momentConstant*. Our calculations are based on [37] and the PX4's Gazebo software in the loop (SITL) source code. As a final note, the perception sensor, *i.e.,* a camera, is added to the model in the same position as its real counterpart. This way, the dynamic environments our platform may encounter are reproduced in Gazebo, providing us with vital information that would protect the aircraft from unexpected behavior during real scenarios. Fig. 3 shows our aircaft as simulated in the Gazebo environment.

III. UAV CONSTRUCTION

As the most common materials used for 3D printing are the PLA, PETG, and ABS, our construction is based on PETG and an FDM 3D printer. Moreover, it is the best solution since it combines high-temperature resistance, good mechanical ABS properties, and the ease of printing, *i.e.,* no need for high printing surface temperatures and a heated chamber. The filament manufacturer provides the appropriate printing settings. Last but not least, Carbon fibre tubes were used for the motor arms and landing gear since they are lightweight, robust, and readily available. Our construction is shown in Fig. 4.

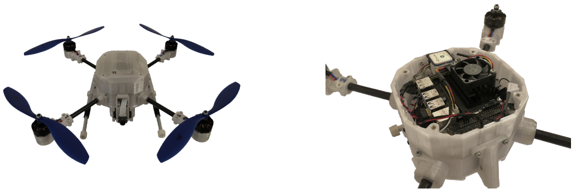

Fig. 4: Our 3D printed modular unmanned aerial vehicle (UAV). In the left picture, the complete quadrotor without the batteries is demonstrated, while in the right one, the Nvidia Jetson nano and the global navigation satellite system (GNSS) module are shown.

IV. LANDING SURFACE RECOGNITION

In case of a UAV malfunction, a safe landing is of utmost importance because an impact due to loss of control will place both people and equipment at high risk. Therefore, our solution is based on detecting people in the area where the system tries to land, aiming to avoid them. However, crowd detection is challenging due to variations in peoples' pose, appearance, size, scale, and orientation. Nevertheless, even if several techniques try to solve the problem through deep learning, they focus on the frontal camera view. Yet, the detection process should be achieved from the above in our case.

YOLO [38], an open-source state-of-the-art object detector performing in real-time, is chosen as the backbone of our pipeline. It approaches object detection as a regression problem by applying a single CNN while bounding boxes are generated to the image for the object's classification and position. In our paradigm, YOLO V3 tiny is adopted [39]. It is a YOLO v3 network with a decreased depth of the convolutional layer; therefore, it is significantly faster while its detection accuracy is slightly reduced. However, its performance gains make it a good fit for our Nvidia Jetson.

The cross-section area of a person from above is tiny as the person's appearance from above is very different from the front. Its accurate detection is difficult in conjunction with the UAV's high altitude and abrupt camera movements. Due to this fact, we opted to use the Okutama-Action dataset [40], which comprises 75093 images captured by UAVs flying at different altitudes and different angles. In order to detect people, each frame taken by the quadrotor's camera is divided into an S × S grid with N number of cells. If the entity's center is laid within a grid cell, then this specific cell is "responsible" for the object's detection. At this point, each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores correspond to the model's confidence that an object is contained in the box and the accuracy of the box that it predicts. Confidence is formally defined as:

$$\text{Conf}(\%) = (P_r(\text{object}) \times IoU_{\text{pred}}^{\text{truth}}) \times 100\% \qquad (1)$$

where Conf is the confidence level expressed in a percentage score, P_r is the predicted box containing the object, and IoU is the intersection over union between the predicted box and

[1] Our model and the installation instructions can be found on https://github.com/telemc97/BigBrotherSamp_SITL−gazebo.git.

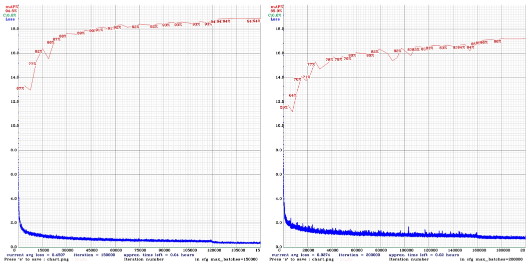

Fig. 5: Left: the learning diagram of YOLO v3 with a maximum mAP of 94% Right: the learning diagram of YOLO v3 tiny with a maximum mAP of 86%.

Training Results				
model	**TP**	**FP**	**FN**	**aver. IoU**
YOLO v3	70244	5396	4113	74.75%
YOLO v3 tiny	65269	10204	9088	66.41%
YOLO v4	71282	5646	3075	75.94%

TABLE I: Training results. Note that there were 74357 truth positives while the confidence threshold was 25%.

the ground truth. Ideally if the object is contained in the predicted box, the Conf $= IoU$. If no object is present then ideally Conf $= 0$.

A. Detection network

We tested YOLO V3 and YOLO V3 tiny. Both models were not pre-trained. Concerning their training, the dataset was split 80% for training and the rest for validation. Both YOLO and YOLO tiny were trained in the same unit. The latter has an input size of 640×640 while the former a size of 416×416. YOLO's achieved mAP was approximately 94%, while YOLO tiny achieved an mAP of 86%. YOLO was trained for $N = 150000$ iterations but after $N = 100000$ iterations the learning process was gradually slowing down and finally at $N = 120000$ iterations the accuracy stopped improving, meaning that our network stopped learning at this point. Similarly YOLO tiny was trained for 200000 iterations. However, at $N = 80000$ iterations, the learning process was slowing down and at $N = 165000$ the learning reached its maximum accuracy. The learning diagrams are shown in Fig. 5, while the training results are given in Table I. Lastly, both networks were trained using 2 Nvidia RTX3090 GPUs.

B. Landing

In the event that the quad-rotor is switched into Land mode and at least one person is detected, it aborts landing and returns in the last safe position before Land mode initiation or in a predefined altitude. The described solution is very simple and lightweight and in essence it adds an additional safety parameter that needs to be checked for the quad-rotor to complete the landing procedure.

V. Performance Evaluation and Results

In this section, we perform a quantitative evaluation of the performance of both our quadrotor and our surface recognition paradigm. Firstly we assess the airworthiness of our platform. Next, the evaluation of the recognition algorithm follows. Finally, tests are performed in the Gazebo simulator and in the real-world. However, it is worth noting that our experimental protocol was performed in a private area in the latter case.

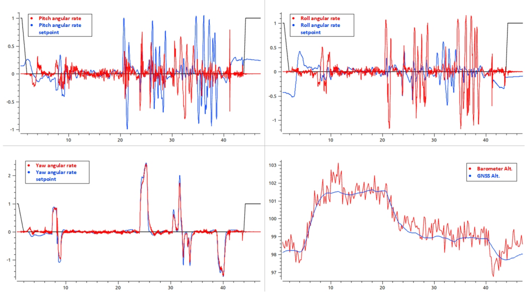

Fig. 6: Top left: the pitch angular rate of the quadcopter is annotated with red and the setpoint (calculated by the flight controller) is annotated with blue. Top right: Roll angular rate and roll setpoint rate. Bottom left: Yaw angular rate and setpoint rate. The four direction changes due to the four way-points are visible to thus subfigure. Bottom right: The flight altitude recorded both by the barometer and GPS. The black line in the first three plots indicates the land status of the quadrotor.

FPS Evaluation			
RTX 2060	**YOLO V3**	**YOLO V4**	**YOLO V3 tiny**
Gazebo	27.3 FPS	26.8 FPS	66.7 FPS
Real-World	30.1 FPS	27.2 FPS	93.3 FPS
Jetson Nano	**YOLO V3**	**YOLO V4**	**YOLO V3 tiny**
Real-World	1.1 FPS	1.0 FPS	4.7 FPS
2X RTX 3090	**YOLO V3**	**YOLO V4**	**YOLO V3 tiny**
Real-World	105.3 FPS	93.5 FPS	606.3 FPS
RTX 3080	**YOLO V3**	**YOLO V4**	**YOLO V3 tiny**
Real-World	93.3 FPS	84.4 FPS	514 FPS

TABLE II: Evaluation of network's performance on Gazebo. Video is captured in 1280×720 The network's input size is 640×640

A. Assessing the platform's airworthiness

Before the test flight, the PID controller must be tuned to our specific airframe so that oscillations and non-linearities are avoided. The procedure is covered thoroughly in PX4's documentation, while a simple mission was planned for the main test comprising four waypoints, which formed a rectangle. The four changes in the aircraft's direction are visible in the yaw chart of Fig. 6.

B. Evaluating the surface recognition pipeline

Two tests were performed; the first was the Gazebo, and the second was the real-world. The latter followed the first, and it was dependent on its success. Concerning the Gazebo, an instance with our model was initiated. We chose Baylands world for the test as it was very close to a real-world scenario. Both YOLO V3 and YOLO V3 tiny were

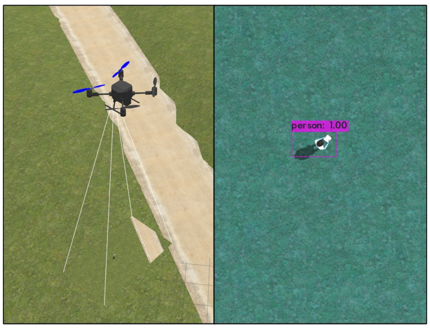

Fig. 7: Left: our quadrotor model hovering above a person in Gazebo environment. Right: YOLO v3 output as it recognizes the person underneath it with a confidence of 98%.

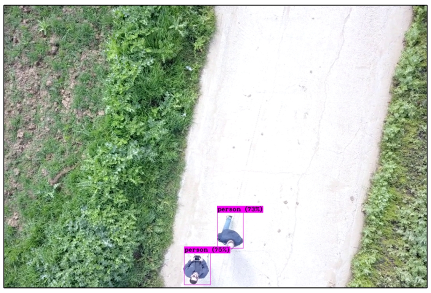

Fig. 8: Person detection in real-world scenario.

evaluated using an Nvidia RTX 2060. Our system achieved to detect people in the flying area quite effectively despite the non-realistic Gazebo's graphics. The results are given in Table II. Regarding the second case, the Jetson Nano lacks the performance of desktop's GPU; nevertheless, it achieved 4.7 FPS when using YOLO V3 tiny (see Table II). Safety landing needs accuracy rather than speed, therefore, the lower framerate is acceptable. We have to note that performance can be improved by reducing the network's input size, but that would impact the overall system's accuracy. A snapshot of the algorithm in action both in Gazebo and in real-world can be found in Fig. 7 and Fig. 8 respectively.

VI. Conclusion and Future Work

In concluding this work, a low-cost, easily repairable, and 3D printable quadrotor capable of carrying an onboard computer is presented. The factors that led to its design and the system's architecture are described in detail. At the same time, a surface recognition pipeline that permits the UAV to land when people are not included in the landing area is also proposed. Using YOLO v3 tiny on an Nvidia Jetson nano, we achieve real-time performance during the aircraft's autonomous navigation. Finally, a Gazebo model capable of being integrated into ROS is implemented and given to the research community. A more efficient surface recognition pipeline for detecting suitable landing areas with lower computational complexity belongs to our plans, while regarding the UAV development, carbon fiber sheets in the aircraft's body will reduce the use of heavy plastics.

Appendix

For calculating the gazebo's parameter's we know that:

$$force = real\,motor\,velocity * real\,motor\,velocity* \\ motor\,constant \tag{2}$$

Similarly to [41], we include:

- ρ, as the air density.
- T, as the thrust produced by the propeller.
- Q, as the torque produced by the propeller.
- P, as the power needed for the revolution of the propeller.
- n, as the revolutions per minute.
- υ_0, as the vertical speed of the propeller in its centreline.
- D, as the propeller's diameter.

To simulate the forces and moments on a quad-copter, we need to predict how T and Q vary with n. This is accomplished using four dimensionless parameters:

- $J = \upsilon_0/Dn$, the advance ratio.
- $C_T = T/\rho n^2 D^4$, the thrust coefficient.
- $C_Q = Q/\rho n^2 D^5$, the torque coefficient.
- $C_P = P/\rho n^3 D^5$, the power coefficient.

next we calculate the the thrust and torque coefficients:

$$T = C_T(J)\rho n^2 D^4 \tag{3}$$

$$Q = C_Q(J)\rho n^2 D^5 \tag{4}$$

The power and torque coefficients for all propellers are related by the following equation:

$$C_Q = C_P/2/pi \tag{5}$$

Using the above we convert the 2 to:

$$T = \omega^2(motor\_constant\_) = (2\pi n^2)(motor\_constant\_) \\ T = C_{T0}\rho n^2 D^4 \tag{6}$$

Thus,

$$motor\_constant\_ = \frac{C_{T0}\rho D^4}{(2\pi)^2} \tag{7}$$

where C_{T0} is the static thrust coefficient at $J = 0$ Again from the PX4 Gazebo SITL source code in line 243 we get that the Gazebo model computes the rotor torque magnitude as: $force * moment\_constant\_$ with 3 and 4 we have that:

$$moment\_constant\_ = \frac{C_{Q0}\rho n^2 D^5}{C_{T0}\rho n^2 D^4} = \frac{C_{Q0}}{C_{T0}}D \tag{8}$$

Looking at the UIUC Propeller Database [42], [43] we get that:

$$C_{T0} = 0.098$$

$$C_{P0} = 0.04$$

in $20°C$ the air density is $\rho = 1.2041kgm^{-1.3}$, the diameter of the propeller is $D = 0.28m$ from 5 we get that $C_{Q0} = 0.00637$ From 7 and 8 we get that:

$$motor\_constant\_ = 1.8 * 10^{-5}kgm$$

$$moment\_consant\_ = 0.0182m$$

By adding the above values to our model's .sdf file we get a more unique approach to our Gazebo model.

Acknowledgement

This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code:T2EDK-00592).

References

[1] B. Canis, "Unmanned aircraft systems (UAS): Commercial outlook for a new industry," 2015.

[2] S. N. Ghazbi, Y. Aghli, M. Alimohammadi, and A. A. Akbari, "Quadrotors unmanned aerial vehicles: A review," *Int. J. Smart Sensing and Intelligent Systems*, vol. 9, no. 1, 2016.

[3] K. Kun, "Reconstruction and development of a 3D printer using FDM technology," *Procedia Engineering*, vol. 149, 2016.

[4] A. Galatas, H. Hassanin, Y. Zweiri, and L. Seneviratne, "Additive manufactured sandwich composite/abs parts for unmanned aerial vehicle applications," *Polymers*, vol. 10, no. 11, p. 1262, 2018.

[5] T. Rayna and L. Striukova, "From rapid prototyping to home fabrication: How 3D printing is changing business model innovation," *Technol. Forecasting and Social Change*, vol. 102, pp. 214–224, 2016.

[6] D. Bak, "Rapid prototyping or rapid production? 3D printing processes move industry towards the latter," *Assembly Automation*, vol. 23, no. 4, pp. 340–345, 2003.

[7] N. Shahrubudin, T. Lee, and R. Ramlan, "An Overview on 3D Printing Technology: Technological, Materials, and Applications," *Procedia Manufacturing*, vol. 35, 2019.

[8] G. D. Goh, S. Agarwala, G. Goh, V. Dikshit, S. L. Sing, and W. Y. Yeong, "Additive manufacturing in unmanned aerial vehicles (uavs): Challenges and potential," *Aerosp. Sci. Technology*, vol. 63, pp. 140–151, 2017.

[9] A. Manecy, N. Marchand, F. Ruffier, and S. Viollet, "X4-mag: a low-cost open-source micro-quadrotor and its linux-based controller," *Int. J. Micro Air Vehicles*, vol. 7, no. 2, pp. 89–109, 2015.

[10] A. M. Mehta, D. Rus, K. Mohta, Y. Mulgaonkar, M. Piccoli, and V. Kumar, "A scripted printable quadrotor: Rapid design and fabrication of a folded MAV," in *Robotics Research*, pp. 203–219, Springer International Publishing, 2016.

[11] C. Lehnert and P. Corke, "μav-design and implementation of an open source micro quadrotor," *Robot. and Automation*, 2013.

[12] K. A. Tsintotas, L. Bampis, and A. Gasteratos, "The revisiting problem in simultaneous localization and mapping: A survey on visual loop closure detection," *IEEE Trans. on Intelligent Transportation Systems*, 2022.

[13] K. A. Tsintotas, L. Bampis, S. Rallis, and A. Gasteratos, "SeqSLAM with bag of visual words for appearance based loop closure detection," in *in proc. Int. Conf. Robotics in Alpe-Adria Danube Region*, pp. 580–587, 2018.

[14] A. V. Savkin and H. Huang, "Asymptotically optimal deployment of drones for surveillance and monitoring," *Sensors*, vol. 19, no. 9, p. 2068, 2019.

[15] V. Balaska, L. Bampis, M. Boudourides, and A. Gasteratos, "Unsupervised semantic clustering and localization for mobile robotics tasks," *Robot. Auton. Systems*, vol. 131, p. 103567, 2020.

[16] E. Lygouras, A. Gasteratos, K. Tarchanidis, and A. Mitropoulos, "ROLFER: A fully autonomous aerial rescue support system," *Microproc. Microsystems*, vol. 61, pp. 32–42, 2018.

[17] E. Lygouras, N. Santavas, A. Taitzoglou, K. Tarchanidis, A. Mitropoulos, and A. Gasteratos, "Unsupervised human detection with an embedded vision system on a fully autonomous UAV for search and rescue operations," *Sensors*, vol. 19, no. 16, p. 3542, 2019.

[18] F. K. Konstantinidis, I. Kansizoglou, K. A. Tsintotas, S. G. Mouroutsos, and A. Gasteratos, "The role of machine vision in industry 4.0: A textile manufacturing perspective," in *in proc. IEEE Int. Conf. Imaging Systems and Techniques*, pp. 1–6, 2021.

[19] V. Balaska, L. Bampis, and A. Gasteratos, "Self-localization based on terrestrial and satellite semantics," *Eng. Appl. Artificial Intelligence*, vol. 111, p. 104824, 2022.

[20] D. Day, "Drones for transmission infrastructure inspection and mapping improve efficiency," *Natural Gas & Electricity*, vol. 33, no. 12, pp. 7–11, 2017.

[21] I. T. Papapetros, V. Balaska, and A. Gasteratos, "Multi-layer map: Augmenting semantic visual memory," in *in proc. Int. Conf. Unmanned Aircraft Systems*, pp. 1206–1212, 2020.

[22] D. Moreno-Jacobo, G. Toledo-Nin, A. Ochoa-Zezzatti, V. Torres, and F. Estrada-Otero, "Evaluation of drones for inspection and control in industry 4.0," in *Technological and Industrial Applications Associated with Intelligent Logistics*, Springer International Publishing, 2021.

[23] K. A. Tsintotas, L. Bampis, S. An, G. F. Fragulis, S. G. Mouroutsos, and A. Gasteratos, "Sequence-based mapping for probabilistic visual loop-closure detection," in *in proc. IEEE Int. Conf. Imaging Systems and Techniques*, pp. 1–6, 2021.

[24] K. A. Tsintotas, L. Bampis, and A. Gasteratos, "Tracking-DOSeqSLAM: A dynamic sequence-based visual place recognition paradigm," *IET Comput. Vision*, vol. 15, no. 4, pp. 258–273, 2021.

[25] C. Symeonidis, E. Kakaletsis, I. Mademlis, N. Nikolaidis, A. Tefas, and I. Pitas, "Vision-based UAV safe landing exploiting lightweight Deep Neural Networks," in *in proc. Int. Conf. Image and Graphics Processing*, pp. 13–19, 2021.

[26] I. Kansizoglou, L. Bampis, and A. Gasteratos, "Deep feature space: A geometrical perspective," *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2021.

[27] I. Kansizoglou, L. Bampis, and A. Gasteratos, "Do neural network weights account for classes centers?," *IEEE Trans. Neural Networks and Learning Systems*, 2022.

[28] G. Castellano, C. Castiello, C. Mencar, and G. Vessio, "Crowd detection for drone safe landing through fully-convolutional neural networks," in *in proc. Int. Conf. Current trends in theory and practice of informatics*, pp. 301–312, 2020.

[29] M. Pethő and T. Zsedrovits, "UAV obstacle detection with bio-motivated computer vision," in *in proc. Int. Work. Cellular Nanoscale Networks and their Applications*, pp. 1–4, 2021.

[30] 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*, vol. 54, no. 9, pp. 1–37, 2021.

[31] K. A. Tsintotas, L. Bampis, A. Taitzoglou, I. Kansizoglou, and A. Gasteratos, "Safe UAV landing: A low-complexity pipeline for surface conditions recognition," in *in proc. IEEE Int. Conf. Imaging Systems and Techniques*, pp. 1–6, 2021.

[32] P. Franciszczak, E. Piesowicz, and K. Kalniņš, "Manufacturing and properties of r-PETG/PET fibre composite–Novel approach for recycling of PETG plastic scrap into engineering compound for injection moulding," *Composites Part B: Engineering*, vol. 154, pp. 430–438, 2018.

[33] O. M. Agudelo and B. Moor, "Computergestuurde regeltechniek exercise session case study: Quadcopter," 2014.

[34] P. Parihar, P. Bhawsar, and P. Hargod, "Design & development analysis of quadcopter," *Compusoft*, vol. 5, p. 2128, 2016.

[35] L. Meier, D. Honegger, and M. Pollefeys, "PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *in proc. IEEE Int. Conf. Robotics and Automation*, pp. 6235–6240, 2015.

[36] M. contributors, "MAVROS-MAVLink extendable communication node for ROS.." https://github.com/mavlink/mavros/tree/master, 2022. [Online; accessed 4-April-2022].

[37] Vernacchia, "Gazebo motor & propeller model notes," 2019.

[38] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[39] P. Adarsh, P. Rathi, and M. Kumar, "YOLO v3-Tiny: Object Detection and Recognition using one stage improved model," in *in proc. IEEE Int. Conf. Advanced Computing and Communication Systems*, pp. 687–694, 2020.

[40] M. Barekatain, M. Martí, H.-F. Shih, S. Murray, K. Nakayama, Y. Matsuo, and H. Prendinger, "Okutama-action: An aerial view video dataset for concurrent human action detection," in *in proc. IEEE Int. Conf. Computer Vision and Pattern Recognition Workshops*, pp. 28–35, 2017.

[41] B. W. McCormick, *Aerodynamics, aeronautics and flight mechanics*. Wiley, 1979.

[42] J. B. Brandt, *Small-scale propeller performance at low speeds*. PhD thesis, University of Illinois at Urbana-Champaign, 2005.

[43] J. Brandt and M. Selig, "Propeller Performance Data at Low Reynolds Numbers," in *AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, 2011.