

# Team Cigogne technical description paper : IMAV 2025

Kaci Maxime, Gass Siméon, Guy Emmanuel, Hadjiat Abderrahim, Pivert Maya, Pavot Thomas, Youbi Lounes, Kiefer Renaud, Schmitt Théo, and Binder Aurore\*  
 INSA Strasbourg, 24 bld de la victoire, Strasbourg, FRANCE

## ABSTRACT

This paper presents the design and implementation of the MAV platforms developed by **Team Cigogne** for the IMAV 2025 outdoor competition. All airframes, including four quadrotors and a flying wing, were entirely manufactured in-house using a hybrid composite made of balsa wood, fiberglass, and carbon fiber. This combination provides an optimal trade-off between lightness, rigidity, and ease of manufacturing, enabling the rapid production of modular and durable structures.

Each vehicle integrates a Pixhawk 4 flight controller coupled with a Raspberry Pi 5 for onboard computation and vision processing. A dedicated Python framework was developed to communicate with the flight controller via MAVLink, enabling real-time visual servoing and autonomous mission execution. This software allows the drones to interpret visual data for tasks such as target tracking, mapping, and autonomous navigation.

The integration of custom-built lightweight structures with embedded vision-based control demonstrates a complete home-made workflow from mechanical design to autonomous operation, tailored for experimentation and competition in aerial robotics.

## 1 INTRODUCTION

The International Micro Air Vehicle (IMAV) competition promotes research in autonomous aerial robotics through realistic mission scenarios inspired by disaster-response operations.

The team's participation focuses on four key missions. **Mission 1** addresses autonomous aerial mapping. **Mission 2** involves visual search and localization of a person based on a text description. **Mission 3** explores cooperative payload transportation. **Mission 4** focuses on water transportation. Together, these missions demonstrate the use of MAVs in a disaster environment.

\*Email addresses:

theo.schmitt1@insa-strasbourg.fr aurore.binder@insa-strasbourg.fr  
 maxime.kaci@insa-strasbourg.fr @insa-strasbourg.fr  
 @insa-strasbourg.fr @insa-strasbourg.fr

The following sections describe the drone design, the methodologies implemented for each mission, and the experimental results obtained in preparation for IMAV 2025.

## 2 DESCRIPTION OF THE MAVS

In this section, we will describe the mechanical design of our drones. We will also explain the hardware and software components used.

### 2.1 Mechanical design

The design of the drone was guided by the need to achieve a balance between **lightness, rigidity, and robustness**. Two composite plates were manufactured from a **balsa wood core** reinforced with **fiberglass** and **carbon fiber**. Balsa was selected for its very low density, while the fibers provide mechanical strength and structural stability.

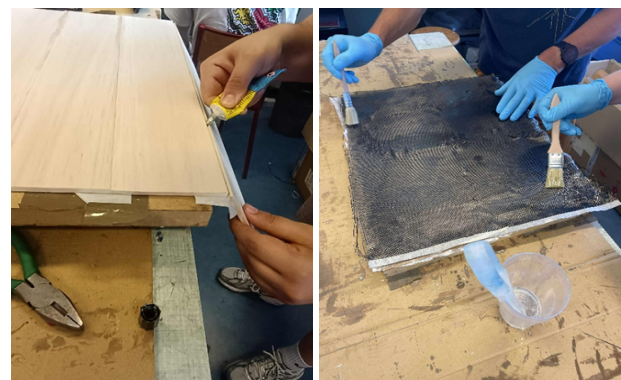


Figure 1: Left: Gluing of the balsa plates. Right: Placement of the fiberglass and carbon fiber layers.

After curing at 45 °C, the composite plates were **machined using a CNC** to cut the body and arms of the drone. This process allowed for a modular, lightweight, and precise structure, easily assembled and reproduced.



Figure 2: Left: CNC machining process. Right: Example of a finished drone part.

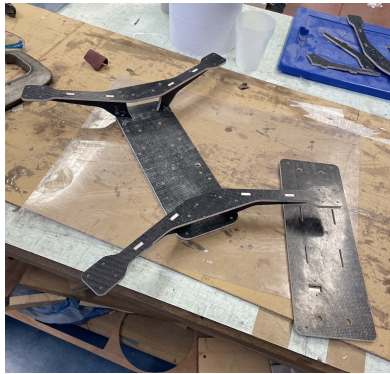


Figure 3: Completed drone frame.

## 2.2 Drone assembly

The drone arms were assembled by gluing the CNC-cut parts and equipping each arm with **brushless motors** mounted on **3D-printed supports** specifically designed to match the arm geometry and evenly distribute mechanical stress.

Additional 3D-printed parts were integrated into the structure:

- a front mount for the camera,
- a rear mount for the GPS module and LiDAR sensor,
- landing skids ensuring ground stability,
- protective parts under the motors to protect the arms in case of impact.

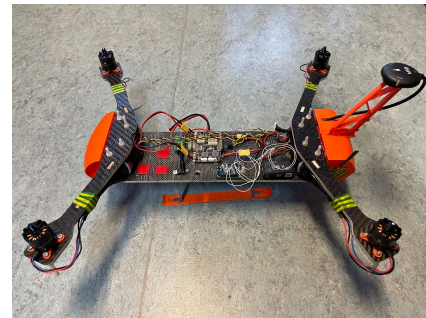


Figure 4: Mounting of the motors and 3D-printed parts.

The arms are attached to the central plate using **plastic screws**, reducing weight, limiting stress on composite materials in the event of a shock, and preventing unwanted electrical conduction between structural elements.

## 2.3 Wiring and electronic integration

The wiring of the drone was designed to ensure both **reliable power distribution** and **efficient data communication** between modules. The architecture is centered around the **Pixhawk 4 flight controller** and the **Holybro PM07 power management module**.

The **4S LiPo battery** is connected to the PM07, which distributes raw voltage to the **ESCs (Electronic Speed Controllers)** through XT60 connectors and provides regulated power to the Pixhawk via its PWR outputs.

Each brushless motor is connected to a 20 A ESC, receiving power from the PM07 and a PWM control signal from the Pixhawk.

A **Raspberry Pi 5** is integrated for onboard processing and camera management. It is powered by a **BEC**, providing a stable voltage for its operation. Communication between the Raspberry Pi and the Pixhawk is established via a USB cable, enabling MAVLink data exchange and coordination of autonomous missions.

For communication and sensors, the drone includes:

- an XBee module for long-range wireless communication, using either 802.15.4 protocol or ZigBee, depending on available modules.,
- a GPS module for positioning and telemetry,
- an RC receiver connected to the SBUS port of the Pixhawk,
- and a camera directly linked to the Raspberry Pi.



Figure 5: Integration of the Pixhawk, PM07 and Raspberry Pi.

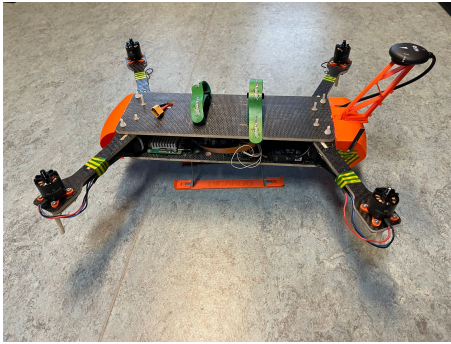


Figure 6: Fully assembled quadrotor drone.

After full assembly, **preliminary tests** were conducted by holding the drone by hand to observe its behavior and control stability using the remote controller. Subsequently, an **initial manual flight test** was performed in an open field to validate system responsiveness, followed by an **autonomous mission** that confirmed the overall functionality of the drone and communication between all onboard modules.

### 3 MISSION 1: MAPPING OF A DISASTER AREA

#### 3.1 Context

The goal of this mission is to simulate a situation where, right after a disaster, mapping of an area is required for rescue and assessment efforts. The drone must autonomously survey a predefined zone, collecting visual data to generate either a 2D panoramic image or a 3D model of the terrain. At the start of the mission, the drone will either take off from the designated take-off area or be lifted by a second drone and released in mid-air for additional points. In both cases, it will then fly to the mapping zone and begin its task. Once the team determines that sufficient data has been collected, the drone must return and land in the take-off zone. The map may be generated in real time for bonus points or delivered within 30 minutes after landing.

#### 3.2 Flight parameters

The mapping can't work if the flight parameters are not correctly determined. The smooth acquisition of the data must be ensured. As mentioned above the drones are fitted with Raspberry pi 5 and pi camera 2 or 3. To ensure that the mapping goes smoothly the goal is to take a large number of pictures and to overlap each other with a rate of at least 60%. The first thing to determine is the area on the ground covered by the camera (with respect to the altitude (alt) at which the photos are taken). The formulas for the sides ( $L$  and  $l$ ) of the rectangular area are:

$$L = 2 * alt * \tan\left(\frac{FOV_L}{2}\right)$$

$$l = 2 * alt * \tan\left(\frac{FOV_l}{2}\right)$$

$FOV_L$  and  $FOV_l$  are values that represents the field of view of the camera ( $102^\circ$  and  $67^\circ$  for the camera pi 3 for example). The altitude is chosen so that the photos still have details on them (not too high) and so that the space covered is large enough (not too low).

The overlap rate ( $R$ ) is given by the following formula :

$$R = 100 - \frac{L_{line} * 100}{\min(L, l)}$$

$L_{line}$  is the space between the lines that the drone is following during missions and we need to compute its value.

We use the minimum between  $L$  and  $l$  because the camera's field of view is not square. This choice ensures that the overlap rate remains sufficient in both directions (longitudinal and lateral). By taking  $\min(L, l)$ , we guarantee that even in the direction where the camera covers a smaller ground area, the overlap will still meet the threshold for successful image stitching during mapping.

Once this is settled, what is left to find is the time between two photos  $t_{photos}$  which is depending on the speed of flight of the drone  $V_{mav}$ , one of the free parameters. The speed is chosen with respect to the exposure of the camera, the wind speed and direction and the battery capacity of the drone.

The formula for  $t_{photos}$  is

$$t_{photos} = \frac{L_{line}}{V_{mav}}$$

$L_{line}$  is here because using the minimum of  $L$  and  $l$  means approximating the rectangle that the camera sees into a square to ensure a good overlap rate. Thus  $L_{line}$  is reused here to keep the same approximation.

#### 3.3 Flightplan

The flightplan is conceived with Ardupilot MissionPlanner, The MAV will follow a gridlike pattern with the space between each line of the grid being  $L_{line}$ .



Figure 7: Grid used for a mapping with  $L_{line} = 16m$

One can see on the picture the grid and the waypoints at the end of each lines. The altitude of flight, the takeoff instruction and the landing instruction are also set with MissionPlanner. It is also possible to add instructions in order to adapt speed. this is useful in order to plan a flight accordingly to the wind direction.

#### 3.4 Taking Photos and Mapping

The photos are taken using a pi camera 2 mounted on the drone coupled with a Raspberry pi 5 which will stock the picture and geotag them, ensuring that each photo gets the right coordinates.

The mapping is done once all the photos have been taken using the opendronemap software to create an ortophotography of the area.

## 4 MISSION 2: FIND A PERSON

### 4.1 Context

This mission aims to assess the capability of an autonomous unmanned aerial vehicle to locate and reach a casualty in a post-disaster search scenario using solely a textual description. The protocol simulates an aerial surveillance operation in which the operator team receives, prior to flight, a brief visual description of the person of interest (clothing, accessories, posture) together with an indicative GPS position. A mannequin representing the casualty is placed within a 20-meter radius of that position. The system is required to execute fully autonomously the phases of takeoff, navigation to the target area, visual identification of the individual matching the description, and landing in the surrounding area without collision.

### 4.2 Proposed solutions

The first approach relies on convolutional neural networks, specifically the YOLO (You Only Look Once) architecture, which performs object detection in a single pass over the image. Lightweight variants such as YOLOv8n or YOLOv8m enable real-time inference on embedded platforms like the Raspberry Pi, providing a good balance be-

tween accuracy and efficiency. Despite its robustness and versatility, this approach remains computationally demanding and may require fine-tuning for specific scenarios.

The second approach employs color-based segmentation in the HSV color space, where thresholding is applied to isolate regions matching predefined color ranges. This method is simple, fast, and well suited for resource-constrained systems, as it requires no training and minimal processing power. However, its performance is strongly affected by lighting variations and background similarities, making it less reliable in complex or dynamic environments.

### 4.3 Color-based segmentation method

This method is based on image analysis in the HSV (Hue, Saturation, Value) color space, allowing the isolation of pixels corresponding to target colors through predefined threshold ranges. The process involves converting the image from BGR to HSV, applying binary masks for the selected color ranges, merging these masks to retain only relevant regions, and extracting the main contour to estimate its centroid.

The detected center is then used to infer the target's relative position in the image and guide the drone's movement accordingly. Thanks to its low computational cost, this approach is well suited for resource-constrained embedded platforms such as the Raspberry Pi, although its performance is sensitive to lighting variations and background color similarities.



Figure 8: Example photo taken by the drone

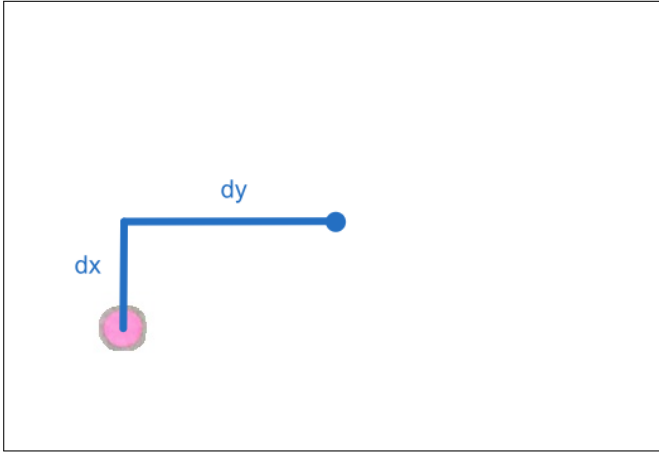


Figure 9: Target detected after filtering

The target centroid in image coordinates  $(u, v)$  is converted to a ground-frame displacement by estimating the ground sampling distance  $p$  (meters per pixel) from altitude  $h$  via the experimentally calibrated linear relation:

$$p = \frac{\alpha h}{H}$$

where  $\alpha = 0.9767$  and  $H$  is the image height in pixels. The metric offsets are then computed as:

$$dx = (c_x - u) \cdot p, \quad dy = (c_y - v) \cdot p$$

where  $(c_x, c_y)$  denotes the image center.

We then consider two types of control: a position-based command, which directs the drone to the target using GPS coordinates, and a velocity-based command, which provides direct  $v_x$  and  $v_y$  velocity instructions to move the drone toward the target.

For the first approach, to compute a new GPS position from a reference position and small displacements in the East-North-Up (ENU) frame, we first convert the geodetic coordinates of the reference point  $(\phi_0, \lambda_0, h_0)$  into Earth-Centered, Earth-Fixed (ECEF) coordinates  $(X_0, Y_0, Z_0)$ . This transformation is performed using the following relations:

$$\begin{aligned} N &= \frac{a}{\sqrt{1 - e^2 \sin^2 \phi_0}} \\ X_0 &= (N + h_0) \cos \phi_0 \cos \lambda_0 \\ Y_0 &= (N + h_0) \cos \phi_0 \sin \lambda_0 \\ Z_0 &= ((1 - e^2)N + h_0) \sin \phi_0 \end{aligned}$$

where  $a$  represents the equatorial radius of the WGS-84 ellipsoid, and  $e$  is its eccentricity.  $\phi_0$  and  $\lambda_0$  denote the

latitude and longitude of the reference point, while  $h_0$  is the altitude.

Once the reference coordinates are expressed in ECEF, the ENU displacements  $(dx, dy, dz)$  are mapped into the ECEF frame using the rotation matrix:

$$\begin{bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{bmatrix} = \begin{bmatrix} -\sin \lambda_0 & \cos \lambda_0 & 0 \\ -\sin \phi_0 \cos \lambda_0 & -\sin \phi_0 \sin \lambda_0 & \cos \phi_0 \\ \cos \phi_0 \cos \lambda_0 & \cos \phi_0 \sin \lambda_0 & \sin \phi_0 \end{bmatrix} \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix}$$

The new ECEF position is then obtained by adding these displacements to the reference ECEF coordinates:

$$X = X_0 + \Delta X, \quad Y = Y_0 + \Delta Y, \quad Z = Z_0 + \Delta Z.$$

Finally, the resulting ECEF coordinates  $(X, Y, Z)$  are converted back into geodetic coordinates  $(\phi, \lambda, h)$ .

The longitude  $\lambda$  is computed directly from the  $X$  and  $Y$  coordinates using  $\lambda = \arctan 2(Y, X)$ , where  $\arctan 2$  is the two-argument arctangent function that returns the correct quadrant for  $\lambda$ .

The distance  $p$  from the Earth's rotation axis to the projection of the point onto the equatorial plane is  $p = \sqrt{X^2 + Y^2}$ .

An initial approximation for the latitude  $\phi_0$  can be obtained using  $\phi_0 = \arctan \left( \frac{Z}{p \cdot (1 - e^2)} \right)$ , where  $e$  is the first eccentricity of the WGS-84 ellipsoid, defined by  $e^2 = 1 - \frac{b^2}{a^2}$ , with  $a$  the equatorial radius and  $b$  the polar radius of the ellipsoid.

The geodetic latitude can then be refined with the relation  $\phi = \arctan \left( \frac{Z + e^2 N \sin^3 \phi_0}{p} \right)$  and the altitude  $h$  above the ellipsoid is computed as  $h = \frac{p}{\cos \phi} - N$ .

where  $a$  and  $b$  are the semi-major and semi-minor axes of the WGS-84 ellipsoid,  $e^2$  is its first eccentricity squared,  $p$  is the projection of the point onto the equatorial plane, and  $N$  is the prime vertical radius of curvature.

For the second approach, a positional deadzone  $|dx| < 0.5\text{m}$  and  $|dy| < 0.5\text{m}$  prevents unnecessary motion. A proportional controller maps these errors to commanded velocities:

$$v_x = \text{clamp}(k_p dx, -V_{max}, V_{max})$$

$$v_y = \text{clamp}(k_p dy, -V_{max}, V_{max})$$

where  $\text{clamp}()$  limits the commands to  $\pm V_{max}$ , and  $k_p$  is the proportional gain. Motion durations are chosen proportionally to distance:

$$t_x = \frac{|d_x|}{|v_x|}, \quad t_y = \frac{|d_y|}{|v_y|}$$

An empirical scaling factor may be applied to account for drone dynamics and control latency. This framework, combining altitude-to-pixel-scale calibration, pixel-to-metric conversion, proportional velocity control with saturation, and distance-proportional timing, provides a lightweight and stable method for centering the drone above a detected target.

#### 4.4 Convolutional neural networks

In addition, a more advanced detection method was implemented using the YOLO (You Only Look Once) model, which is widely recognized for its efficiency in real-time object detection. The model is applied to images captured by the drone to identify all persons present in the scene.

The detection process begins with loading the pre-trained YOLO network, followed by inference on each image to generate bounding boxes for detected objects. Only detections corresponding to the "person" class are retained for further analysis.

To refine the selection, a color-based comparison is performed within each bounding box by referencing a preprocessed color-filtered image. This allows the system to identify the individual whose clothing best matches the textual description provided prior to the mission. The target with the highest proportion of matching pixels is then selected as the primary target.

By combining semantic recognition through YOLO with color-based segmentation, this method significantly improves detection reliability and accuracy, particularly in scenarios where multiple individuals or objects are present, surpassing the limitations of purely color-based approaches.

### 5 MISSION 3: TRANSPORTATION CHALLENGE

#### 5.1 Context

This mission focuses on employing a swarm of three or more micro aerial vehicles (MAVs) to cooperatively transport a payload in a disaster response scenario, such as delivering medical supplies or evacuating a small object (e.g., an infant).

#### 5.2 Proposed solution

In the proposed framework, one MAV acts as the master unit, broadcasting its position via Bluetooth. The follower MAVs compute their desired positions based on predefined compass bearings and distance offsets relative to the master. A proportional-integral (PI) control law is then applied to generate the velocity commands required to maintain formation.

Due to time constraints, this approach was not implemented or experimentally validated.

### 6 MISSION 4: LOAD AND RELEASE

#### 6.1 Context

In the context of natural disasters such as earthquakes, this mission aims to use one or more MAVs (Micro Aerial Vehicles) to transport and release water, similar to how helicopters perform such operations. The MAVs must collect water from a circular pool located at a given GPS position and release it over a designated target zone indicated by another GPS coordinate.

#### 6.2 Proposed Solution

To achieve the "Load and Release" mission, a complete electromechanical system was developed to autonomously collect and discharge a defined volume of water between two GPS coordinates. The system consists of a mechanical winding mechanism, a custom electronic control board, and a 3D-printed structural support designed to fit under the quadrotor frame.

#### Operational Sequence

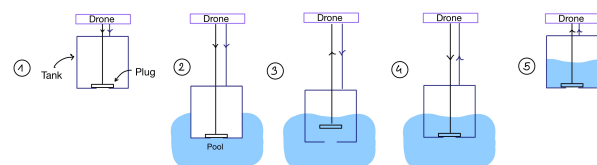


Figure 10: Mission 4 sequence diagram

To complete the mission, we use a tank suspended by a cable, capable of drawing in and storing water. The tank features a hole and a plug : when submerged, water enters through the hole and the plug seals it, making the tank watertight. To release the water, the plug is connected to a cable that can be pulled to open it. The cables for both the tank and the plug are wound on the same spool, driven by a single motor, with the plug's cable slightly shorter so it lifts automatically when the tank is lowered.

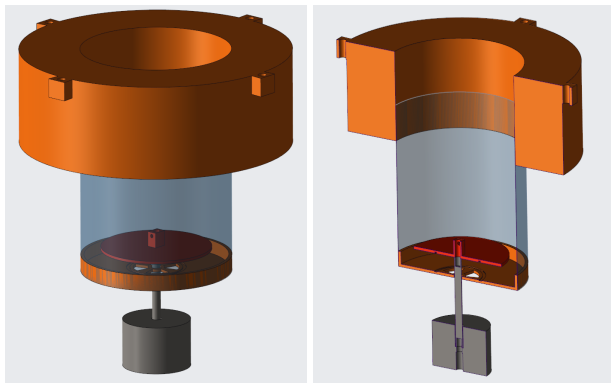


Figure 11: CAD of the float, the tank, and the plug

### Mechanical Design

The lifting system relies on a drum-based cable winder actuated by a servo motor. The cable is attached to a small container capable of holding approximately 600 g of water. The mechanism must lift the load vertically from a pool during the collection phase and maintain it securely during the flight.

The winding drum was dimensioned to ensure sufficient torque while minimizing energy consumption, since flight endurance is limited by the drone's battery capacity. Analytical modeling showed that a 1 cm drum radius offered an optimal compromise between speed, torque, and current draw.

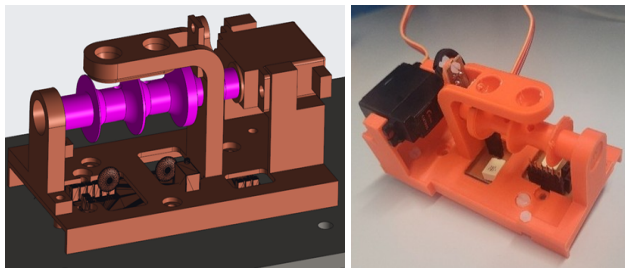


Figure 12: Winding system

An interruption sensor based on a magnetic switch was added to count the number of turns and limit the winding length to 2 m, preventing cable entanglement. The container uses a mechanical plug system that seals automatically when submerged and opens during release by a short cable offset of the same winding spool.

The entire mechanism—including the frame, drum, and mounting brackets—was modeled in CREO and manufactured via 3D printing.

### Electronic System

A dedicated PCB was designed under Proteus and built around a PIC16F15214 microcontroller, chosen for its compactness and low power consumption. It interprets a PWM input from the flight controller (Pixhawk) to trigger winding or unwinding commands.

The circuit integrates:

- a Hall-effect sensor to detect drum rotation,
- a PIC16F15214 for the regulation,
- a 6 V power stage for the servo motor and a 3.3 V logic supply derived from the drone's 15 V bus.

The PCB was fabricated on a single-sided metal substrate to reduce mass.

## 7 OUTDOOR SURVEILLANCE CHALLENGE

Long-endurance flight is a key aspect of the Outdoor Surveillance Challenge, where autonomous aerial monitoring over extended periods is required. To address this objective, Team Cigogne designed and built a fully custom flying wing platform optimized for endurance and lightweight efficiency.

The airframe was entirely manufactured in-house. It consists of an extruded polystyrene foam core covered with layers of fiberglass and carbon fiber, providing both rigidity and structural strength while maintaining an exceptionally low mass. The central fuselage, designed in 3D printing, houses the avionics, propulsion system, and payload.

The aircraft features a wingspan of 1.2 m and a total take-off weight of approximately 800 g, including a 4S 2000 mAh LiPo battery. This configuration enables an estimated flight endurance of around one hour. Propulsion is provided by a 400 W peak brushless motor, offering efficient cruise flight at 15 m/s with a current consumption of about 2.5 A.

The flight control system is based on a Pixhawk autopilot running the PX4 firmware, providing reliable stabilization, navigation, and mission execution. This setup allows full integration with the same Python-based framework developed for the quadcopters, enabling unified mission planning, telemetry, and potential visual servoing capabilities.

Thanks to its aerodynamic efficiency, structural lightness, and open-source control architecture, the flying wing complements the quadrotor fleet by extending mission duration and coverage area for outdoor surveillance operations.



Figure 13: Photograph of the fixed-wing drone developed by the team

## 8 CONCLUSION

This paper presented the complete development process of the aerial platforms designed by Team Cigogne for the IMAV 2025 competition. All vehicles, from the quadrotors to the fixed-wing aircraft, were entirely conceived and manufactured in-house using lightweight composite materials and 3D-printed components. The use of balsa, fiberglass, and carbon fiber enabled the creation of robust yet lightweight structures tailored to the mission requirements.

The integration of a unified control architecture based on Pixhawk controllers running PX4, coupled with a Python framework for high-level coordination and vision-based autonomy, demonstrates the team's capability to design both hardware and software in a fully coherent system.

Future work will focus on enhancing the onboard autonomy and perception algorithms, as well as improving coordination between multiple MAVs to perform more complex cooperative missions in real-world environments.