IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 February 2024, accepted 11 March 2024, date of publication 18 March 2024, date of current version 27 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3378607

RESEARCH ARTICLE

An Open-Source UAV Platform for Swarm Robotics Research: Using Cooperative Sensor Fusion for Inter-Robot Tracking

SINAN OĞUZ[1,2], MARY KATHERINE HEINRICH[1], MICHAEL ALLWRIGHT[1], WEIXU ZHU[1], MOSTAFA WAHBY[1], EMANUELE GARONE[2], (Member, IEEE), AND MARCO DORIGO[1], (Fellow, IEEE)

[1]Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle (IRIDIA), Université Libre de Bruxelles (ULB), 1050 Brussels, Belgium
[2]Unité d'enseignement en Automatique et Analyse des Systémes (SAAS), Université Libre de Bruxelles (ULB), 1050 Brussels, Belgium

Corresponding author: Sinan Oğuz (sinan.oguz@ulb.be)

This work was supported in part by the Program of Concerted Research Actions (ARC) of the Université libre de Bruxelles, in part by the Office of Naval Research Global under Award N62909-19-1-2024, and in part by the China Scholarship Council under Award 201706270186. The work of Mary Katherine Heinrich and Marco Dorigo was supported by the Belgian F.R.S.-FNRS, of which they are a Postdoctoral Researcher and a Research Director, respectively.

ABSTRACT In this work, we present an open-source unmanned aerial vehicle (UAV) platform for research in swarm robotics. In swarm robotics, groups of robots collaborate using local interactions and collectively solve tasks beyond an individual robot's capabilities. Individual robots must have onboard processing, communication, and sensing capabilities to autonomously react to their neighbors and immediate environment. Most research involving UAVs in swarm robotics presents only simulation results, while key landmark studies with real UAV swarms have used UAV platforms that were custom-built for the respective study. One important reason for this is that no commercial UAV platform comes pre-equipped with the ability to identify and track the positions and poses of nearby drones using only onboard sensors and computation, and in research platforms, the relevant sensing technologies are currently under development. Our aim is to provide a platform that allows swarm robotics researchers to test their algorithms on real UAVs, without having to develop their own custom-built UAVs or to wait until more advanced sensing technology is ready off-the-shelf. We provide a well-documented, entirely open-source UAV platform—*S-drone* (Swarm-drone)—to foster and support UAV swarm research in a laboratory environment. The *S-drone* uses fiducial markers in the environment and cooperative feature-level sensor fusion for inter-robot tracking to track the presence, identity, relative 2D position, and relative 2D orientation of neighboring peers. The *S-drone* is suitable for a wide range of contexts, supports quad-camera vision-based navigation and a variety of onboard sensing, and is extensible. It is especially suited for swarm robotics research because it can operate using strictly onboard processing and sensing without the need for global positioning systems, motion capture systems, or ground stations for off-board sensing.

INDEX TERMS Autonomous aerial vehicles, autonomous systems, drones, indoor navigation, multi-robot systems, multi-UAV systems, open source hardware, open source software, swarm intelligence, swarm robotics.

The associate editor coordinating the review of this manuscript and approving it for publication was Guillermo Valencia-Palomo.

I. INTRODUCTION

In this paper, we present an open-source UAV platform for swarm robotics research. The platform includes hardware,

© 2024 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

software, simulation environment, and software support for managing experiments with real UAVs.

In recent decades, there have been many advances in swarm robotics research using real swarms of small mobile ground robots. There have also been several notable achievements in real aerial swarms, in which newly developed swarm robotics algorithms were tested with real UAVs (e.g., [1], [2], [3]). However, these studies are exceptions to the general trend: swarm robotics research with UAVs is much less common than with ground robots and most aerial swarms research is conducted in simulation and not verified with real UAVs [4], [5], [6]. In the swarm studies that have used real UAVs, the researchers mostly tested their new swarm algorithms using custom-built UAV platforms that were developed "in-house" by the same lab(s) that developed the algorithms [1], [2], [3]. This strategy is effective, but is very work-intensive: therefore, it is not feasible for many research labs and groups, and is also likely to be inefficient for the research community as a whole.

Robot swarms rely on local interactions between the robots in the swarm to self-organize their activities. Local interactions typically require robots to be able to detect their neighbors (e.g., detect the neighbors' relative distance, bearing, and/or position) and to differentiate between them, as well as to differentiate between UAVs and other objects in the environment. Such detection is highly challenging in UAVs, especially when moving at high speeds, and this is a key reason why many swarm algorithms in the literature have not yet been tested on real UAVs (cf. [4], [6]).

There are currently many UAV platforms available off-the-shelf, but the majority of these platforms do not have open hardware and software and therefore are not ideal for much of robotics research. The exceptions are commercial open development platforms and open scientific research platforms, of which the most relevant for swarm robotics are: (1) the ModalAI M500 drone[1] for GPS-denied navigation and obstacle avoidance using stereo cameras and (2) the *Palm-Sized Drone* [3] for swarm navigation in cluttered outdoor environments using ultra-wideband (UWB) distance-based localization drift correction. While both of these platforms offer advanced capabilities for swarm navigation, neither the ModalAI M500 platform nor the Palm-Sized Drone platform come pre-equipped with the ability to locally identify and track the full relative poses (i.e., positions and orientations) of nearby drones. It might be feasible to extend the software of either platform to achieve these capabilities in their targeted navigation contexts using their current off-the-shelf payloads, but this would of course be a non-trivial development task.

To the best of our knowledge, no commercial drone (whether open or fully proprietary) comes pre-equipped with the ability to identify and track the poses of nearby drones using only onboard sensors and computation. However, research on sensing technologies is advancing quickly, for example using UWB localization [3], [7], [8], [9] or visual localization supported by machine learning [10], [11], [12]. UWB state estimation is very promising, and advances are being made to address outstanding challenges, such as insufficient accuracy, complicated initialization, and consistency issues in UWB-odometry state estimation [7] as well as scalability issues related to communication throughput in UWB-odometry or UWB-visual-inertial state estimation, especially for 2D pose estimation [8]. Another very promising technology is ultraviolet direction and ranging (UVDAR) [13], [14]. UVDAR can allow drones to identify and track the positions and orientations of nearby drones using only onboard sensors and computation, and is currently being implemented as an extension [15] to the open-source MRS research platform [2]. In the mid-term future, it is likely that some of these technologies will become available on commercial drone platforms that are open and come pre-built off-the-shelf. Based on this state of the art, our motivation is to provide a UAV platform that can be used in laboratory environments in the intervening years, to make it easy and accessible for swarm robotics researchers to test their algorithms on real UAVs, without having to develop their own custom-built UAVs or to wait until more advanced sensing technology is ready off-the-shelf.

There are also many options for building a custom UAV platform using a combination of custom components and off-the-shelf open components: most commonly, the Pixhawk autopilot unit [16], [17] and PX4 open-source flight control software [18], accompanied by a large ecosystem of compatible payloads and collaborative software projects. For our UAV platform for swarm robotics, we made use of this existing ecosystem when possible and integrated custom components when necessary.

This paper contributes tools to facilitate the practical execution of swarm robotics research. In particular, we contribute an open-source UAV research platform equipped for cooperative inter-robot tracking in a laboratory environment. To support the local interactions needed in swarm robotics research, our open-source UAV platform is pre-configured to detect fiducial markers in the environment and use that sensor information to track the presence, identity, relative 2D position, and relative 2D orientation of neighboring peers, using cooperative feature-level sensor fusion for inter-robot tracking. Our UAV platform includes open hardware and software, an open simulation environment for development, and an open user interface for operators to manage their demonstrations with real UAVs. Also, our approach for cooperative feature-level sensor fusion for inter-robot tracking is not exclusive to our platform and can be implemented on other UAVs if desired, for example if a lab already has an existing UAV fleet (assuming the platform is open and customizable and has appropriate payloads). It is important to note that our platform has some limitations for swarm research: it is not currently suitable for experimentation with real UAVs outdoors and/or at high speeds. However, it is our view that many swarm behaviors that can be studied within these constraints still require extensive research,

[1] https://docs.modalai.com/m500/

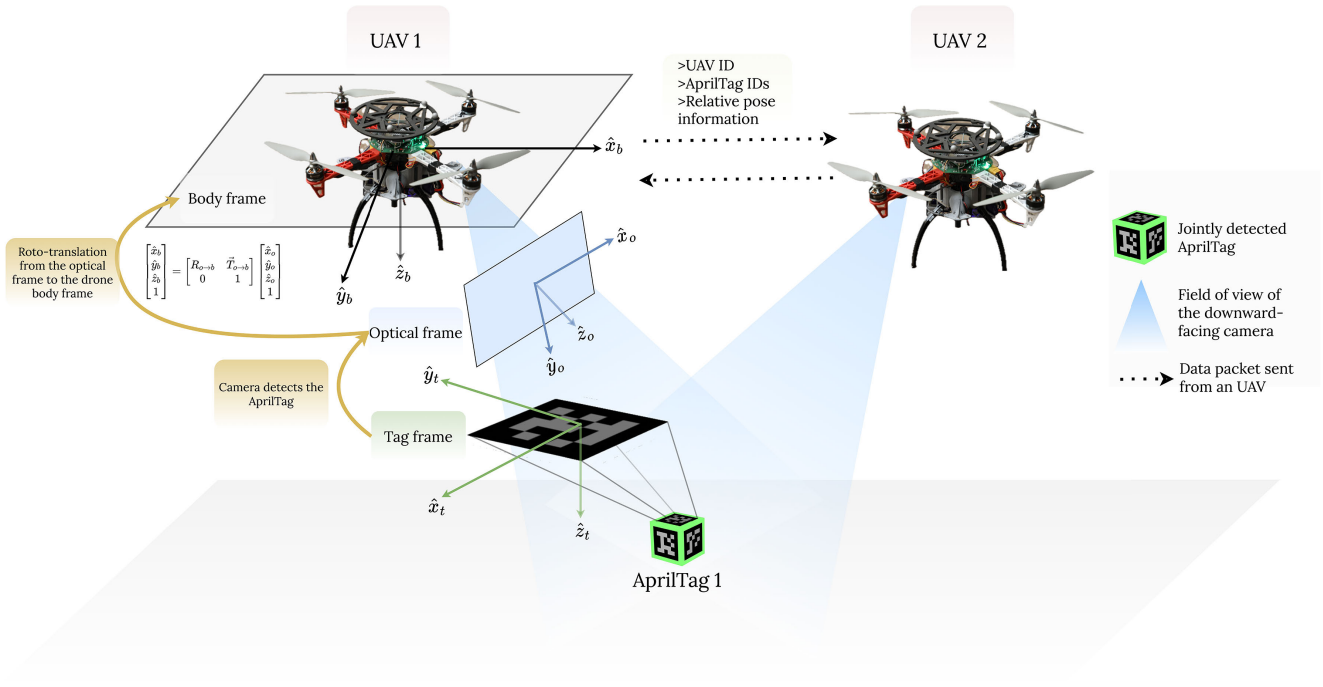


FIGURE 1. Illustration of tag-based localization and cooperative sensor fusion in the *S-drone*. When a UAV detects an AprilTag, it rotates and translates the tag's relative pose information from the camera's optical frame to the UAV body frame. If two UAVs simultaneously detect the same AprilTag, they exchange information about the tag. Using this exchanged information, each UAV calculates the relative pose of the other UAV, with respect to its own body frame. For details, please see the open-source repository (see Sec. III).

including self-organized collective perception, mapping, decision making, and learning.

A. COOPERATIVE SENSOR FUSION FOR INTER-UAV TRACKING

Our aim is to supply an approach for local inter-UAV tracking that is straightforward for swarm robotics researchers to use on our *S-drone* or implement on another UAV platform. We have therefore included in our *S-drone* platform an open-source approach for local inter-UAV tracking in laboratory environments that does not require specialist sensor technology nor expertise. The onboard capability requirements for our approach are: downward-facing visual camera(s), a single-board computer, and the ability to send messages between specific UAVs when in close physical proximity.

The *S-drone* is equipped for autonomous vision-based waypoint navigation, using AprilTags [19] in the environment (see Sec. IV for demonstrations). The AprilTag detection used for individual waypoint navigation is also used for inter-UAV tracking. When two *S-drone*s have the same AprilTag in their field of view, they can detect each other indirectly, using feature-level cooperative sensor fusion between UAVs (see Fig. 1). Effectively, each UAV knows its own ID and can detect the IDs and relative poses of any AprilTags in its onboard cameras' fields of view. Then, each UAV broadcasts the IDs of the AprilTags it currently detects, along with its own ID. If a UAV receives a message that contains an AprilTag ID matching one of those it currently detects, it unicasts the feature-level information it has about the respective AprilTag to the ID of the respective message sender. Thus, when two UAVs detect the same AprilTag, they both receive the pose information of that AprilTag, relative to the other UAV's body frame. Using this information, each UAV can calculate the pose of the other UAV, relative to its own body frame. The *S-drone* platform is equipped with this simple cooperative sensor fusion and includes all software components needed for operation (including camera calibration and control scripts for example experiments, see Secs. II-C and III). We demonstrate inter-UAV tracking with the *S-drone* in scenarios of multi-UAV formations and cooperative object transport (see Sec. IV for demonstrations).

B. RELATED UAV RESEARCH PLATFORMS

There are several existing UAV research platforms that are similar to the *S-drone*, described in detail below. A summary table of the characteristics of these platforms is also provided (see Table 1).

The FLA platform [20], designed for obstacle detection and environmental mapping, is based on the DJI F450 with a 450 mm wheelbase (motor to motor distance). It incorporates a Hokuyo 2D LiDAR (light detection and ranging), a downward-facing Garmin LiDAR-Lite rangefinder, multiple cameras, a high-performance VectorNav VN-100 inertial measurement unit (IMU), and is powered by an Intel NUC i7 onboard computer. It weighs 3 kg, has a flight duration of

TABLE 1. Characteristics of existing UAV research platforms (i.e., non-commercial) that are similar to the *S-drone*.

	S-drone	FLA platform [20]	ASL-Flight [21]	Agilicious [22]	MRS platform [2]	Palm-Sized Drone [3]
Target application	Swarm robotics	Obstacle detection, environment mapping	Visual inertial odometry	Autonomous agile flight	Underground tunnel exploration	Multi-UAV outdoor navigation
Based on	DJI F450	DJI F450	DJI Matrice 100	Armattan Chameleon	Holybro X500	Not included
Wheelbase (motor-to-motor distance)	450 mm	450 mm	650 mm	264 mm	500 mm	114 mm
Distance/ depth sensing	Downward-facing cameras; downward-facing Benewake TFmini Lidar	Hokuyo 2D Lidar; downward-facing Garmin rangefinder	Intel RealSense ZR300 depth imaging sensor	Not included	Ouster OS1 Lidar; Intel RealSense D435i depth camera	Intel RealSense D430 depth camera
Camera	multiple	multiple	Not included	Intel RealSense T265 tracking camera	x2 Basler RGB cameras	grayscale camera
Internal measure-ment unit	Not included	VectorNav VN-100	Not included	Not included	Not included	Not included
Onboard computer	UP Core 01/16	Intel NUC i7 mini-PC	Intel NUC i7 mini-PC	NVIDIA Jetson TX22 embedded AI computing device	Intel NUC i7 mini-PC	NVIDIA Xavier NX system-on-module
Weight	2.5 kg	3 kg	2.4 kg	750 grams	4.8 kg	300 grams
Flight duration	15 minutes	5 minutes	16 minutes	10 minutes	25 minutes	11 minutes
Cost (approx.)	2 250 EUR	5 500 EUR	4 700 EUR	1 600 EUR	12 000 EUR	1 450 EUR
Software/ hardware design	Open-source	Undisclosed	Hardware proprietary to DJI; Software undisclosed	Available on request (some exceptions, e.g., based on requester nationality)	Hardware undisclosed; Software open-source	Open-source
Local pose tracking of nearby UAVs	Yes	No	No	No	With extension, in development	No

5 minutes, and is estimated to cost approximately 5 500 EUR. However, its software and details of its hardware design are undisclosed.

The ASL-Flight platform [21], designed for verification of custom visual inertial odometry framework, is based on the proprietary DJI Matrice 100 with a 650 mm wheelbase. It is equipped with an Intel RealSense ZR300 depth imaging sensor and an Intel NUC i7 onboard computer. Weighing 2.4 kg, it has a flight duration of 16 minutes and is estimated to cost approximately 4 700 EUR. The UAV's hardware is proprietary to DJI and the details of its design are undisclosed. Its VIO (visual inertial odometry) framework is also not open-source.

The Agilicious platform [22], designed for autonomous agile flight, utilizes an Armattan Chameleon frame with a 264 mm wheelbase. It incorporates an Intel RealSense T265 tracking camera and an NVIDIA Jetson TX22 onboard computer. It has a flight duration of 10 minutes, weighs 750 grams and has an estimated cost of approximately 1 600 EUR. Even though it is open-source, there are some access restrictions based on criteria such as the requester's nationality.

The MRS platform [2], designed for underground tunnel exploration, utilizes a Holybro X500 frame which has a 500 mm wheelbase. It is equipped with an Ouster OS1 LiDAR, one Intel RealSense D435i depth camera, two Basler RGB cameras and an Intel NUC i7 onboard computer. Weighing 4.8 kg, it has a flight duration of 25 minutes and is estimated to cost 12 000 EUR. The software is open-source and system-level design is partially open-source. However, the details of its hardware design are undisclosed.

The Palm-Sized Drone [3], designed for multi-UAV navigation in cluttered outdoor environments, has a 114 mm wheelbase. It carries an Intel RealSense D430 depth camera and a grayscale camera, and it is powered by an NVIDIA Xavier NX onboard computer. It has a flight duration of 11 minutes, weighs 300 grams and is estimated to cost approximately 1 450 EUR. Both its software and hardware designs are open-source.

Besides these research platforms that are closely related to the *S-drone*, another very well-known platform is the Crazyflie [23], which has an 80 mm wheelbase, weighs 27 grams and is estimated to cost 220 EUR. However, due to its minimal hardware configuration, it is primarily used for basic algorithm verification tasks, as it does not support onboard sensing or computation capabilities, and therefore we do not report it in Table 1.

II. THE S-DRONE PLATFORM

Our *S-drone* for swarm robotics research (see Fig. 2, Table 2) is designed with the objectives to: support both piloted and autonomous flight, support the development of control algorithms for deployments ranging from a simple single UAV to complex missions with UAV swarms, and operate without an external positioning system or support from a ground station. Based on these objectives, we prioritized the following features when designing the *S-drone* platform:

- **Robot-to-robot coordination and vision-based navigation.** The *S-drone* has onboard localization, navigation, and coordination capabilities for multi-robot operations. Also, its quad-camera configuration gives it a wide enough field of view at low altitudes to support vision-based navigation indoors.
- **Decentralization.** It can operate without any external centralized infrastructure such as global positioning systems, motion capture systems, off-board sensing, off-board processing, or ground control stations.
- **Simulator.** The *S-drone* comes with an open-source plug-in for simulation in ARGoS [24], a multi-physics engine simulator for robot swarms.
- **Transferability from simulation to real hardware.** The *S-drone* plug-in for ARGoS provides a high-level control interface, enabling users to implement the same control software in both simulation and on real hardware.
- **Extensible control.** By using ARGoS and the PX4 flight controller, it is possible to develop and implement a wide range of new swarm algorithms and control laws.
- **Open-source.** All hardware, software, and mechanical design files, as well as assembly instructions, modeling and flight control design, and operation instructions, are hosted on Open Science Framework.[2]
- **Accessible and extensible hardware.** Most of the components are commercially available, and those that are not can be manufactured (either in-house or by a third-party service provider) using the provided design files and widely available machinery (e.g., 3D printer). The *S-drone* is also customizable. It uses well-modularized components when possible and it is straightforward to integrate a variety of off-the-shelf sensors (e.g., a localization module for GPS-guided outdoor navigation or a 360° LiDAR module for mapping). Deeper customization can also be made using the design files provided (e.g., mechanical mounts can be customized to change the sensor types or fields of view).
- **Software support for managing experiments.** The *S-drone* is supported by an experiment *Supervisor* program, used for starting, monitoring, and shutting down real UAV experiments.

[2] https://osf.io/hp6rf/

TABLE 2. Open-source hardware specifications table.

Hardware name	*S-drone* (Swarm-drone)
Open-source licenses	Hardware and software: MIT License
Cost of hardware	2 250 EUR
Project repository	https://osf.io/hp6rf/

Based on these design priorities, we equipped the *S-drone* with a combination of standard UAV components, other off-the-shelf components, and a few custom components.

First, the standard UAV components include many that are essential for the autonomous or piloted flight of UAV platforms, including electronic speed control (ESC) modules, brushless motors, propellers, a single-point LiDAR, an optical flow module, a battery, and a flight controller. These standard components are off-the-shelf from the hobbyist market and therefore are widely available, affordable, well-modularized, and well-interfaced.

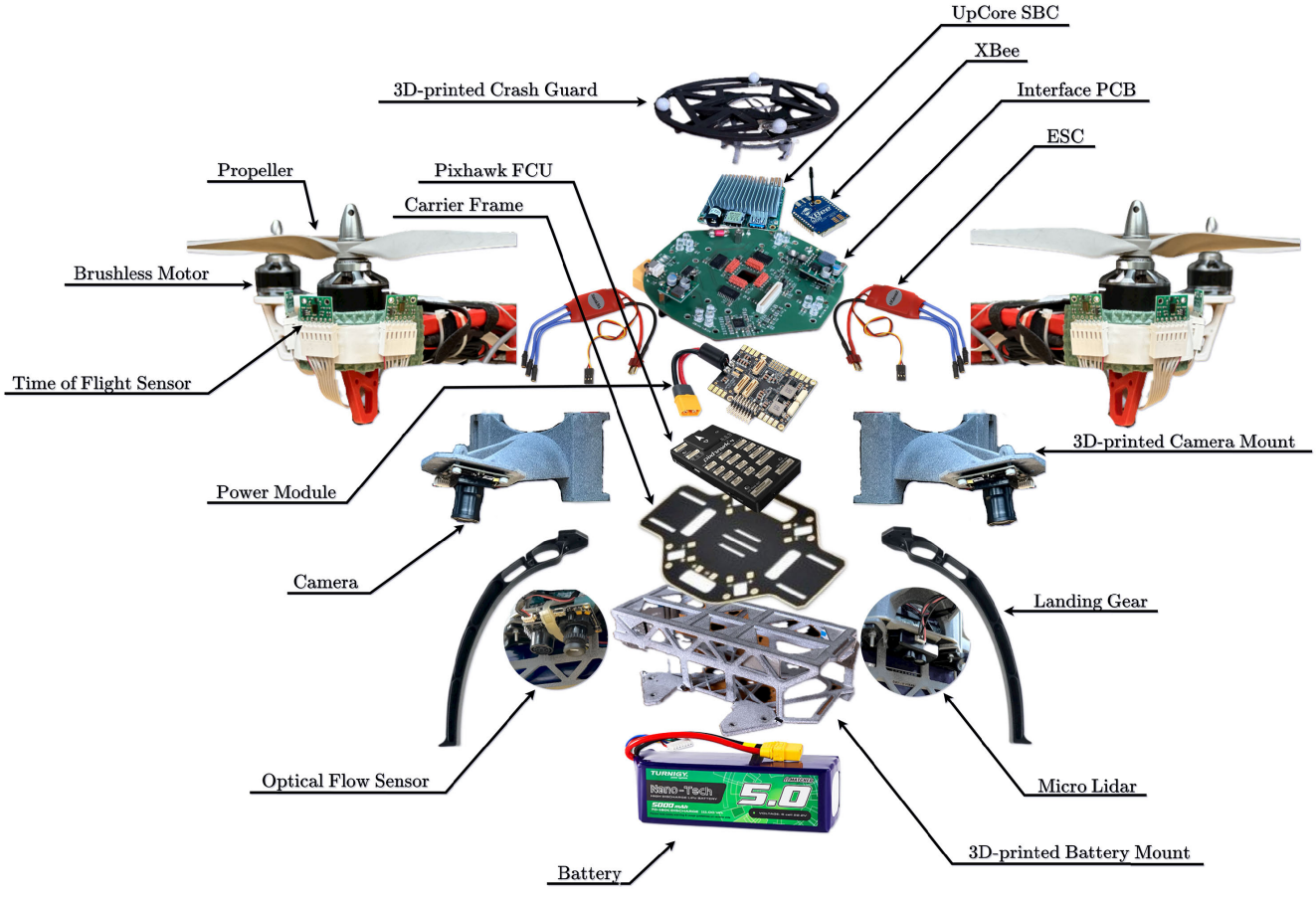
Second, the other off-the-shelf components are those that have not been produced specifically for UAV platforms. The *S-drone* uses these components for autonomous flight and decentralized coordination. They are: the cameras, the single-board computer, and the optional attachment for collision avoidance using time-of-flight (ToF) distance sensors. The *S-drone* is also equipped with two separate WiFi communication links: one for low-level interaction with the UAV (e.g., controlling and monitoring the Pixhawk4 flight controller) and one for communication with the single-board computer (e.g., interaction with Linux or ARGoS).

Third, some custom components are necessary. Many versions of the off-the-shelf components exist, but not all combinations are compatible, and because many of them are not modularized for UAV platforms, they need to be interfaced. Also, the components overall are interconnected and constrain each other in a complex way. For these reasons, we designed a custom printed circuit board to interface the components. The *S-drone* interface printed circuit board (PCB) includes power modules and connectors for peripherals, powers the flight controller and other components, and provides reliable data communication between the onboard computer and the flight controller and other components. The *S-drone* also includes custom cables and some custom mechanical components.

Built with these components, the *S-drone* has the following basic operational characteristics:

- a maximum flight time of approximately 15 minutes in hover (as built, without extensions),
- a maximum velocity of 25 m/s,

(a)



(b)

FIGURE 2. **(a) A set of four *S-drones*. (b) Exploded view of the *S-drone*.**

- an operational angular velocity of 220°/s on roll and pitch axes and 15°/s on the yaw axis,
- a maximum altitude while flying autonomously without external infrastructure (i.e., by relying on its downward-facing single-point LiDAR and optical flow sensor) of approximately 12 m,
- can detect tags at a rate of five frames per second (FPS) with an input resolution of 700 × 700 pixels when flying at an altitude of 1 m and using the default AprilTag and camera configuration.

In the remainder of this section, we describe the details of the *S-drone*'s electronics, mechanical components, and software.

A. ELECTRONICS

The electronic components of the *S-drone* can be categorized in four groups: 1) the flight control electronics, 2) the interface PCB, 3) the propulsion electronics, and 4) the autonomy electronics. The connectivity diagram of the components can be seen in Fig. 3 and full details are given in the bill of materials in the repository (see Table 3).

1) FLIGHT CONTROL ELECTRONICS

The *S-drone*'s flight control electronics are responsible for controlling the UAV's attitude, position, and trajectory. These components are all commercially available for modularized UAV platforms.

In the *S-drone*, we use a Pixhawk4 flight controller from the manufacturer Holybro. The Pixhawk4 is based on an open-source project [16] and is optimized to run the open-source autopilot software called PX4. Its primary purpose is to fly and control the UAV without constant 'hands-on' remote control by a human operator being required. It has a central STM32F765 processor, an I/O STM32F100 processor, an accelerometer, a gyroscope, a magnetometer, barometric air pressure sensors, five serial ports, three I2C ports, and four SPI buses. In the *S-drone*, the Pixhawk4 is connected to the interface PCB via a serial port. Through the interface PCB, it communicates with the Up Core at 921 600 baud rate and 80 000 bytes/second data rate. The Pixhawk4 is operated at 4.9-5.5 V voltage and is powered from an external PM07 power module, which is manufactured specifically for the Pixhawk and also manages power distribution to the ESC modules for the brushless motors. The PM07 power module also sends information to the Pixhawk4 about the battery voltage and current supplied to the motors.

For relative position estimation, the *S-drone* includes a downward-facing PX4Flow optical flow smart camera module from Holybro. To help in precision hovering, terrain following, and precision landing, the *S-drone* also includes a downward-facing TFMini single-point LiDAR from Benawake. The optical flow sensor captures 752 × 480 pixels images and estimates the UAV's velocity at 400 Hz. The single-point LiDAR is low-cost and low-power, has a range of 12 m, and provides measurement data at 100 Hz at a baud rate of 115 200. These sensors work indoors and outdoors without the need for an external supplement, e.g., external illumination for the flow sensor.

2) INTERFACE PCB

The *S-drone* has a two-sided interface PCB that allows communication between electronic components and provides connectors for peripherals. Here we describe the main components of the PCB (see Fig. 3).

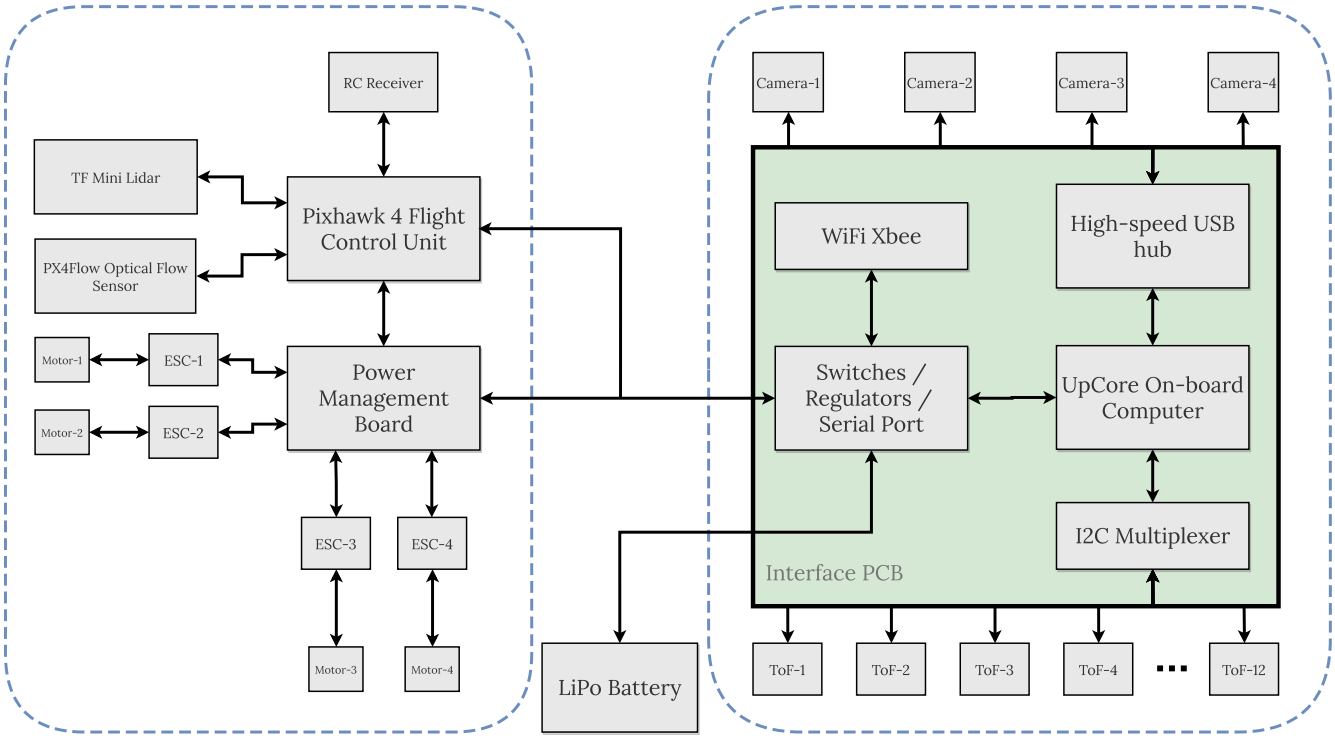
The *S-drone* also includes some removable components to optionally support manual piloting, which can be important, e.g., during development phases, even when working with swarm algorithms for autonomous drones. Using a DSMX remote receiver from Spektrum, an operator can remotely control the UAV's movement (e.g., speed, direction, throttle, yaw angle, etc.), switch between the flight modes, or engage a kill-switch in an emergency. Using SiK telemetry radios from Holybro, which provide a wireless data link between a ground control station and the Pixhawk4, an operator can adjust control gains and review telemetry data in real-time, which is useful for development phases.

3) PROPULSION ELECTRONICS

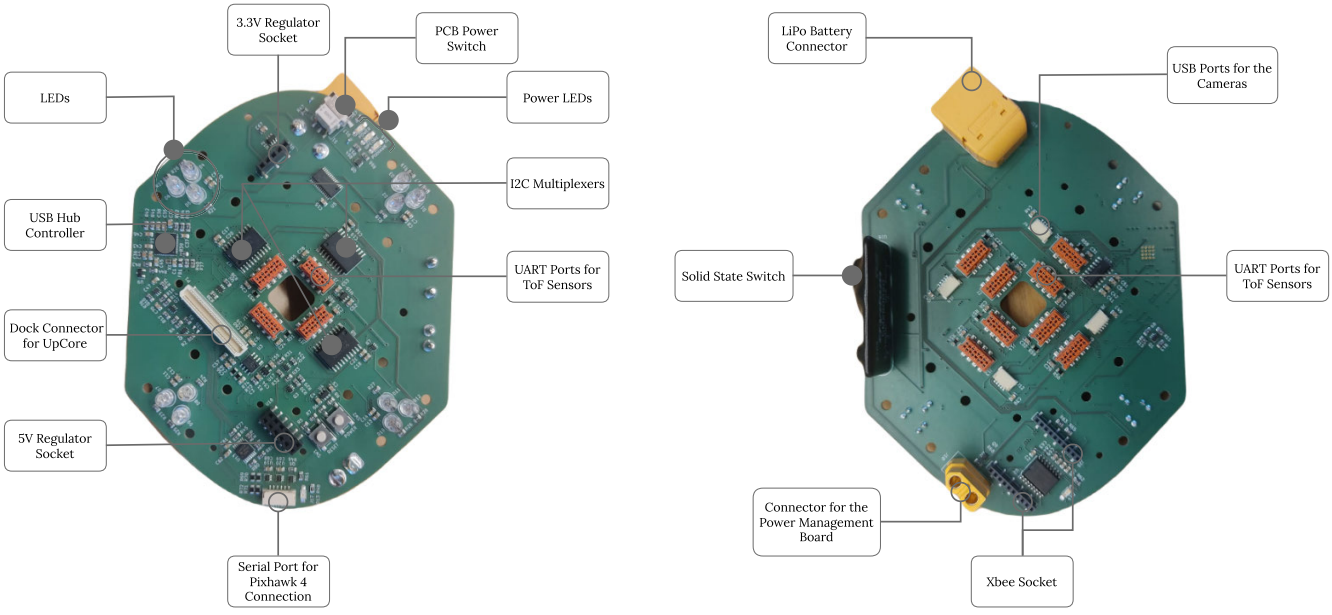
The propulsion system of a UAV is crucial: it determines key performance metrics such as flight time, payload capacity, flying speed, and range. The propulsion electronics of the *S-drone* comprise motors, ESC modules, and a battery. Note that, although there are many possible versions of these components, only a few combinations are compatible. Poor combinations would result in UAVs that either fail under system stress or cannot take off at all.

The *S-drone* has four MN-series brushless DC motors from T-Motor. The current fed to the motors is controlled by four 30 A ESC modules with SimonK firmware. The ESC modules control the speed of the motors based on the PWM signal from the Pixhawk4. Finally, the *S-drone* is powered by a lithium-ion polymer (LiPo) 5000 mAh four-cell battery.

On the top face of the PCB, located next to the LiPo battery connector, there is a power switch to turn the PCB power on and off. The top face also has two step-down voltage regulators (a 3.3 V regulator for the control circuitry and a 5 V regulator for the Up Core) and a docking connector for the Up Core. On its bottom face, the PCB includes a standard docking socket for an XBee WiFi module, which communicates with a router in a 2.4 GHz frequency band. For experiment management (e.g., triggering the start of an experiment or engaging a kill-switch for safety), this module provides a two-way communication link between the experiment *Supervisor* software and the UAV. Both sides of the PCB also include connectors for autonomy electronics (refer to Sec. II-A.4): USB ports with custom JST connectors for cameras and UART ports for ToF distance sensors with 27-degree field-of-view flash LiDAR. To orchestrate the data coming from many ToF sensors, the PCB includes three low-voltage four-channel I2C multiplexers, which are each responsible for sending the distance information from up to



(a)



(b)

FIGURE 3. **(a) Connectivity diagram for the electronics of the *S-drone*. (b) The interface PCB.**

four ToF sensors to the Up Core via I2C (i.e., supporting up to twelve ToF sensors in total).

On the top face of the PCB there are also 12 through-hole round RGB LEDs and four surface mounted diode (SMD)

LEDs. The through-hole RGB LEDs are driven by a 16-bit LED driver from NXP, used to set the brightness of the RGB channels and for color mixing. The colors of the through-hole RGB LEDs can provide visual state indicators for a developed swarm algorithm and the colors of the surface-mounted LEDs provide visual feedback about the on/off status of the PCB, Pixhawk4, Up Core, XBee WiFi module, and *S-drone* autonomous mode.

4) AUTONOMY ELECTRONICS

The autonomy electronics of the *S-drone* comprise an Up Core single-board computer, four downward-facing cameras, and an optional attachment for collision avoidance using twelve ToF distance sensors with 27-degree field-of-view flash LiDAR. Note that the attachment for collision avoidance is considered optional (i.e., not required for flight) because the *S-drone* can operate without it, and the attachment is designed to be easily mechanically interchangeable with other custom sensor attachments. Substitutions can be made by customizing the provided open-access design files.

The Up Core single-board computer is responsible for processing sensing and control information, as needed for autonomy and swarm algorithms. It communicates with the flight controller, the cameras, and extensions for further autonomy such as the ToF distance sensors. The Up Core has an Intel Atom X5 Z8350 processor, an Intel Gen. 8 HD Graphics 400 GPU, 4 GB DDR3L RAM, 64 GB eMMC, two USB 2.0 ports, one UART port, and an embedded WiFi module.

The cameras of the *S-drone* are used for autonomous vision-based navigation and detection of other robots during decentralized coordination. The camera modules are manufactured by Leopard Imaging, are based on a 5 MP OV5650 image sensor from OmniVision, and have a 2.8 mm manual focus lens. The pixel data from the image sensors are routed from these connectors to the Microchip USB4715 low-power USB 2.0 hub controllers. Then, the USB hub controller routes the pixel data to the Up Core's USB 2.0 data port.

The optional ToF distance sensors are intended to increase the situational awareness capabilities of the *S-drone* for purposes such as collision avoidance. The attachment includes twelve VL53LOX flash LiDAR ToF distance sensors from Pololu. They have a composite field of view that covers most directions around the UAV and can measure the distance of an object at up to 2 meters.

B. MECHANICAL COMPONENTS

The *S-drone* includes several 3D printed components, which have been designed according to mechanical stiffness, anti-vibration, and manufacturing constraints. The components have been designed to reduce volume as much as possible (and therefore reduce both weight and manufacturing costs) while maintaining appropriate mechanical properties.

It is important to note that the camera mounts need to be carefully designed to meet anti-vibration requirements. The UAV is relatively heavy and therefore the motors are quite powerful and produce strong vibrations. Also, the cameras need to be mounted sufficiently away from the body of the UAV in order to have an unobstructed view, effectively creating cantilever beams, which can easily deflect with significant magnitude on the unsupported end (i.e., the end with the camera). Any customization of the *S-drone* will need to take this consideration into account.

C. SOFTWARE

The software components for the *S-drone* allow for transferring code from simulation to real hardware and support simulations using the accompanying simulator ARGoS. The seamless transition from simulation to real hardware enables swarm algorithms to be developed in the simulation environment before deploying them on real UAVs, reducing the amount of testing on real hardware and the potential for crashes during the development phase. The full firmware and support software are provided in the Open Science Framework repository.

The *S-drone* uses the Yocto Project[3] to automatically build a custom embedded Linux distribution at runtime. In the Yocto build system, layers define a specific version and configuration of the Linux kernel alongside userspace configuration and software. We use a Docker image to partially automate the configuration of the build system and to provide a clean and consistent environment for the build process.

The custom layer `meta-drone` for the Yocto build system includes a Linux kernel based on Linux Yocto 5.2 with several patches.[4] The layer `meta-drone` also contains configuration segments that are combined to form the `defconfig` for the kernel, which results in the necessary kernel modules being compiled and installed, and the ACPI configuration to communicate the location and configuration of all the devices to the Linux kernel drivers.

The *S-drone* preinstalled software includes LibIIO (a userspace library for the kernel industrial input-output API), Intel's threading building blocks (TBB) library, the AprilTag library, MJPEG streamer, the PX4 flight system library, and Python 3 (with packages for libjpeg-turbo, pymavlink, pySerial, and V4L2). We have also included a custom JSON-based RPC service called Fernbedienung,[5] which allows us to run programs remotely and interact with them over the standard input, output, and error streams. This service is written in Python and is started automatically on boot using `Systemd`.

For running experiments, we use an executable control interface of the UAV defined in the ARGoS simulator.

[3] https://www.yoctoproject.org

[4] Patches are an upstream version of PCA954x I2C multiplexer driver for compatibility with ACPI (Advanced Configuration and Power Interface), a workaround for USB video class (UVC) bandwidth calculation (so that multiple cameras can operate on the same bus), and an update to the VL53l0X ToF driver to incorporate sampling based on the industrial input-output (IIO) kernel API.

[5] https://github.com/iridia-ulb/fernbedienung-python

TABLE 3. Supplementary documentation.

File	Format	OSF Repository[2] location
Demonstration video for the *S-drone*	MP4	Demonstration / demo.mp4
Assembly video for the *S-drone*	MP4	Assembly / AssemblyVideo.mp4
Operation manual for the *S-drone*	PDF	Operation / OperationManual.pdf
S-drone modeling and flight control	PDF	SupplementaryDocuments / ModelingAndFlightControl.pdf
UAV BOM	XLS	BillofMaterials / BOM.xls (Sheet:Parts_S-drone)
Interface PCB BOM	XLS	BillofMaterials / BOM.xls (Sheet:Parts_InterfaceBoard)

TABLE 4. Design files.

File	Format	OSF Repository[2] location	GitHub Repository URL
Interface PCB	EDA software project	DesignFiles / Electronics / ...	https://github.com/iridia-ulb/drone-pcb
Camera mount	STEP, STL, OBJ	DesignFiles / Mechanics / CameraMount / ...	-
Upper plate crash guard	STEP, STL, OBJ	DesignFiles / Mechanics / CrashGuard / ...	-
Battery mount	STEP, STL, OBJ	DesignFiles / Mechanics / BatteryMount / ...	-
Sensor attachment	STEP, STL, OBJ	DesignFiles / Mechanics / SensorAttachment / ...	-
Custom cable harnesses	PDF	DesignFiles / CableHarnesses / ...	-
Calibration boards	SVG, PDF	DesignFiles / Calibration / ...	-

TABLE 5. Repositories for software components.

Software component	OSF Repository[2] location	GitHub Repository URL
Flight controller PX4 firmware and parameter list	Software / FlightControl / ...	-
Yocto system image layer for the Up Core	Software / SystemImage / ...	https://github.com/iridia-ulb/meta-drone
Camera calibration software	Software / Calibration / ...	-
Control scripts and ARGoS plugin for the *S-drone*	Software / SdroneControl / ...	https://github.com/iridia-ulb/argos3-drone

By using this executable and the ARGoS libraries, we are able to use the same control software in simulation as we do on real UAVs. The executable can be passed configuration settings (e.g., identifiers, message router configuration settings) as arguments and provides an implementation of all of the sensors and actuators defined in the control interface for the *S-drone*. To give a few examples, the flight system is implemented using the PX4 library; the camera system is implemented using the V4L2, Intel TBB, and AprilTag libraries; the ToF distance sensors are implemented using LibIIO; and the LEDs are implemented on top of the sysfs virtual filesystem.

We also use a custom *Supervisor* program for starting, monitoring, and shutting down multi-robot experiments. The program provides a web-based user interface in which the user can record data from an Optitrack tracking system, log messages sent between robots, and capture ARGoS's standard output and standard error from each UAV during an experiment. The *Supervisor* program has been written in Rust. The back-end is built on top of the Tokio asynchronous framework/runtime and the front-end (compiled to WebAssembly for the browser) is built on top of the Yew web framework. The front-end and back-end communicate with each other over a WebSocket.

For *S-drone* controllers, and for the development of swarm algorithms or other types of control, the *S-drone* uses Lua control software, including Lua scripts which run on top of the Lua bindings for the sensors and actuators defined in the control interface. By writing our controllers in Lua, there is no need for recompilation each time a change is made to our control software. Also, because Lua has a small footprint and is fast and portable, it can be easily embedded, avoiding potential problems with, e.g., incompatible instruction sets or linking libraries. Experiments with the *S-drone* use an XML configuration file to define the sensors and actuators in use and the update rate of the controller. More details about using *S-drone* control software and running experiments are given in the section for assembly and operation instructions (see Sec. III-C) and the accompanying online repository (see Table 3).

III. OPEN-SOURCE REPOSITORY

In open-source online repositories, we provide: the bills of materials and assembly and operation instructions (including a demo video and the modeling and flight control information) for the *S-drone* (see Table 3), all the design files for its custom components (see Table 4), and all the software components needed for its development and operation (see Table 5). The software components have been described in the previous section. The remainder of this section gives the details of the repositories for design files, bills of materials, and assembly and operation instructions.

A. BILLS OF MATERIALS

Two bills of materials are located in the repository (see Table 3): one for the components of the UAV and another

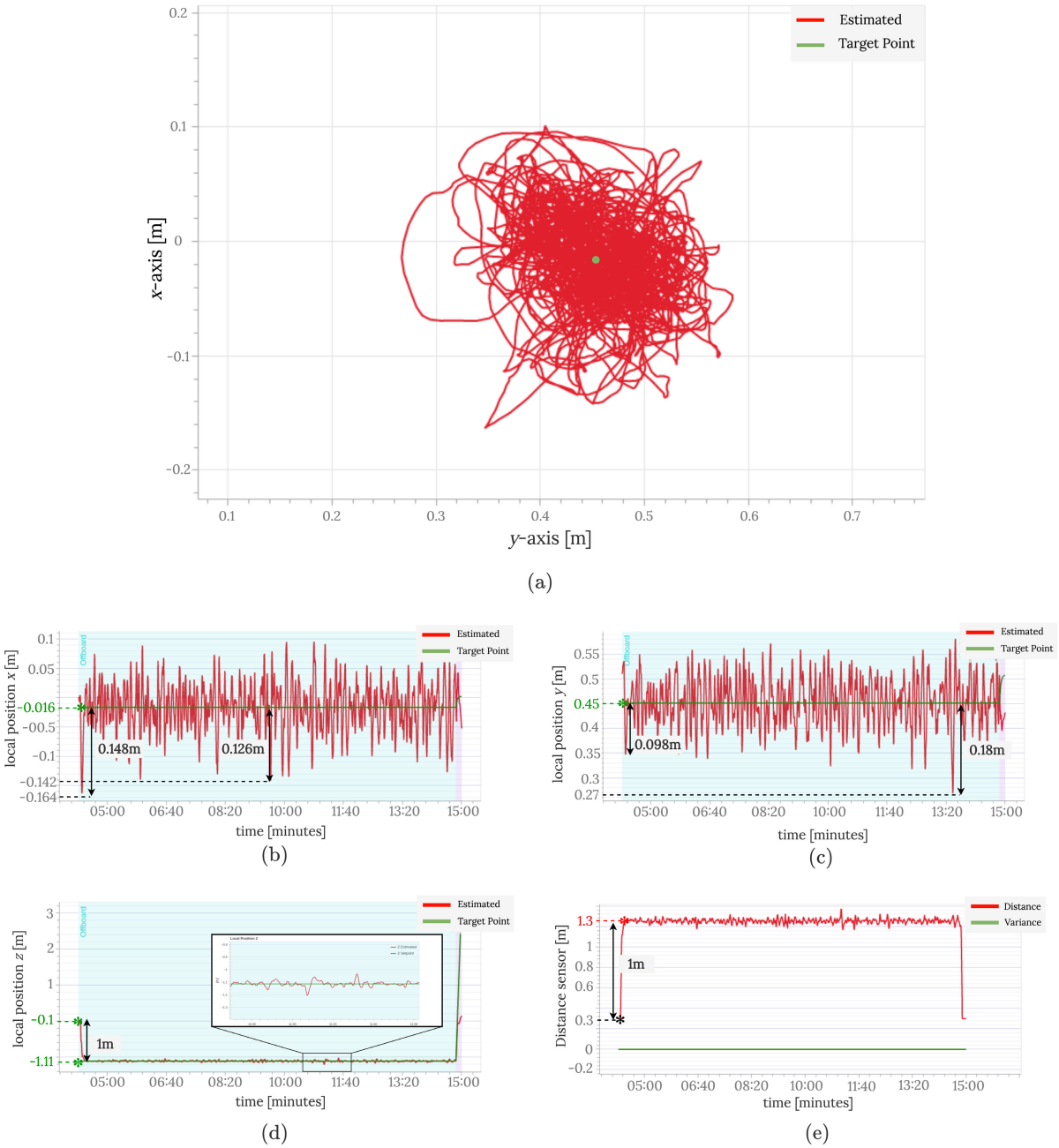


FIGURE 4. Autonomous indoor hover performance during a 10-minute flight. (a) Flight shown in the *xy*-plane. The target point for the drone, communicated from the onboard computer to the Pixhawk for maintaining hover flight, is a tuple of *x* and *y* values (shown in green). Specifically, the target point tuple consists of: (b) *x*-coordinate values set at -0.016 m for the *x*-axis, indicated by the green star and line, and (c) *y*-coordinate values set at 0.45 meters for the *y*-axis, also marked by the green star and line. (d) Relative *z*-coordinate values. Note that the PX4 uses a North-East-Down coordinate frame, so the real positive *z*-axis is negative in this plot. The take-off altitude is -0.1 m, and the target altitude is -1.11 m (indicated by the green stars and line). The maximum deviation is 0.1 m, and the UAV reaches an altitude of -1.21 m at that time (indicated by a blue star). (e) Relative *z*-coordinate values read by the downward-facing distance sensor.

for the components of the interface PCB (excluding cost). The cost of the interface PCB, including manufacturing the board, ordering the components, and having the board assembled, is listed on the main bill of materials. The bill of materials for the interface PCB is intended to be used as a reference, as fabricators and board assemblers might use their own spreadsheets styles. To manufacture the interface PCB, a bill of materials in a customized style can be automatically populated using an EDA software such as KiCad and the provided design files for the interface PCB (see Table 4).

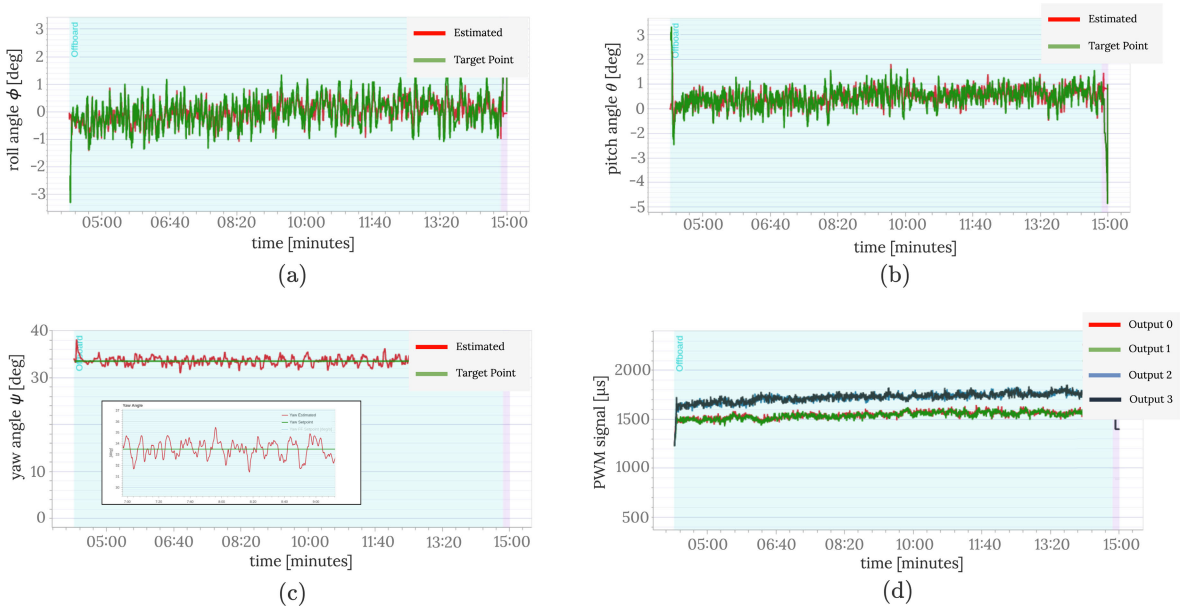


FIGURE 5. Autonomous indoor hover performance during a 10-minute flight. (a) Roll angle. (b) Pitch angle. (c) Yaw angle. (d) Control signals that are sent to the individual motors.

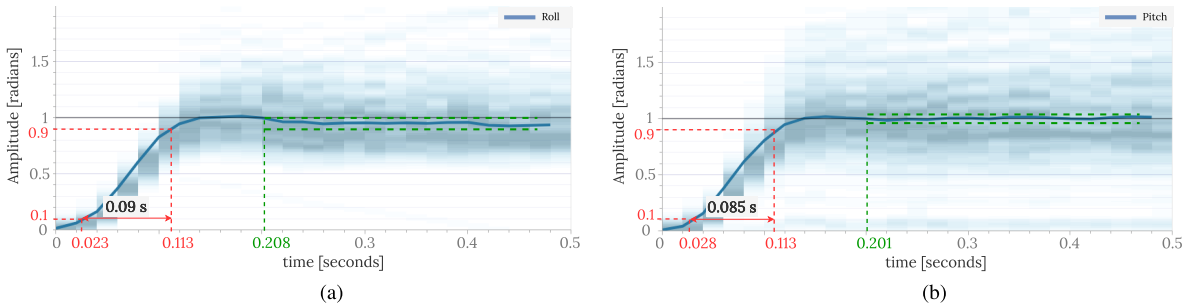


FIGURE 6. Unit step input responses of the *S-drone* controllers. The red arrows indicate the time interval required for the response to rise from 10% to 90% of its final value. The vertical green lines show the settling time, which is the time it takes for the response to be stably within a certain range of the final value (horizontal green lines). (a) The step response of the roll rate controller. (b) The step response of the pitch rate controller.

B. DESIGN FILES

The design files required to manufacture the *S-drone* are summarized in Table 4. The 3D printed parts are provided in several file formats, including formats compatible with open-source software (e.g., FreeCAD 0.17-7) and the circuit board design files are provided in a format compatible with KiCad, an open-source EDA. For the circuit board, we provide an archive that contains the complete project in addition to a quick reference PDF for checking the functionalities and layout of the PCB. For each of the 3D printed parts, we provide an archive of multiple file formats for editing purposes and for submission to a rapid prototyping facility (e.g., STL, OBJ). For the custom cables, we provide an archive of the production-ready shop drawings in PDF, for submission to a cable manufacturer. For the calibration boards needed for the operation instructions, we provide print-ready files.

C. ASSEMBLY AND OPERATION INSTRUCTIONS

We provide a detailed video presenting the assembly process of the *S-drone*. The video is located in the repository (see Table 3).

An operation guide is provided in an online repository (see Table 3) and includes: operation instructions for the *S-drone*; operation instructions for the Pixhawk; procedures for downloading, installing, and configuring the PX4 firmware; procedures to prepare a system image for the Up Core; procedures for calibrating the cameras for accurate detection and vision-based navigation, and instructions for executing the example routines.

IV. DEMONSTRATIONS OF THE S-DRONE

In this section, we demonstrate the *S-drone* in basic flight operations and cooperative behaviors: single-UAV take-off and hover flight, single-UAV vision-based navigation, and

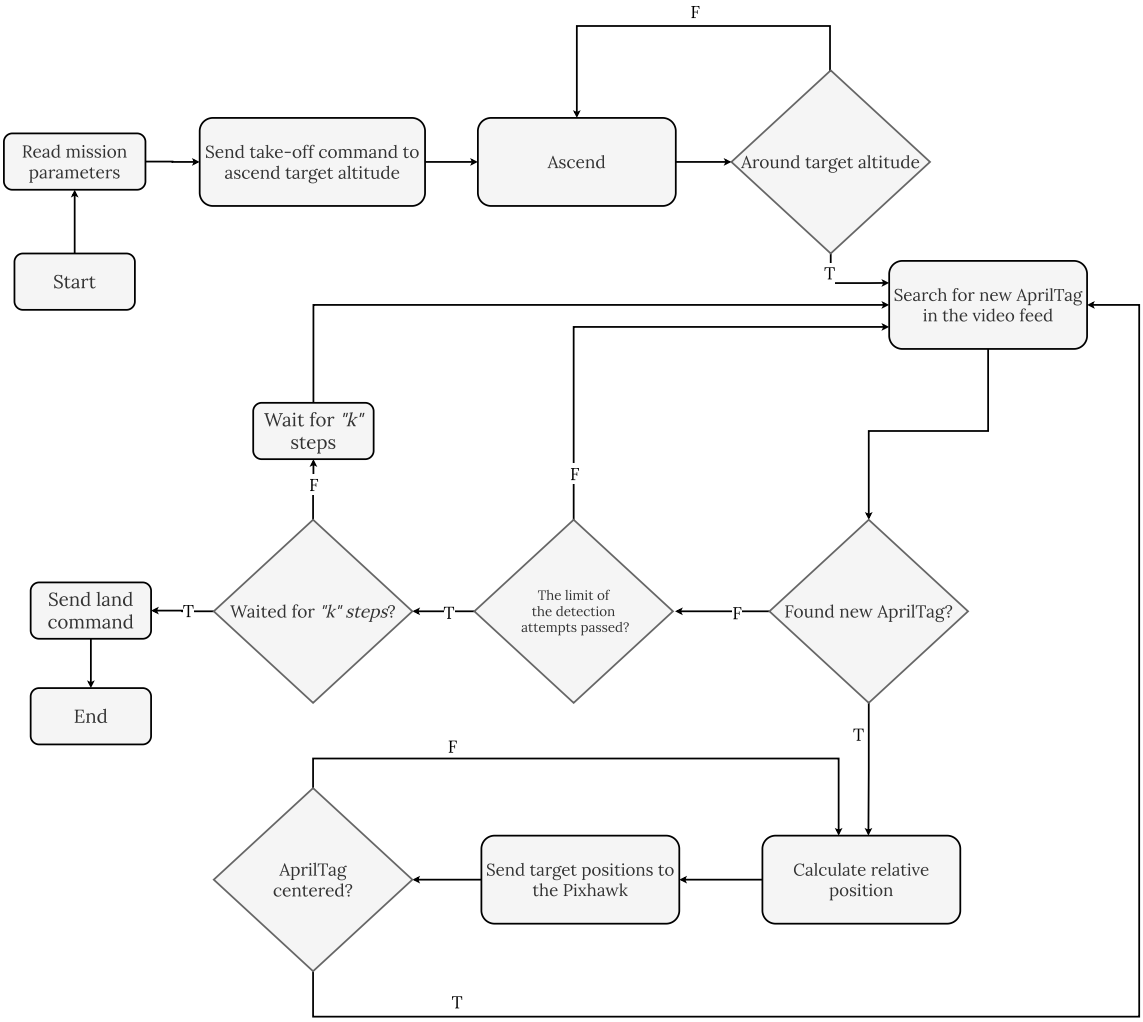


FIGURE 7. Control algorithm flow chart for vision-based navigation with the *S-drone*.

two swarm demonstrations (one multi-UAV flight in a dynamic formation, and one multi-UAV object transport).

A. SINGLE-UAV TAKE-OFF AND HOVER FLIGHT

This experiment validates stable hover flight of the *S-drone* and reports deviation from the target position. In this setup, the *S-drone* tracks a static target position and hovers for 10 minutes after autonomous take-off. The take-off position is set as the target position in the *xy*-plane, and the altitude target position is set to 1 m. Using the cascaded PID controller of the *S-drone*, the target position is fed to the position controller, which triggers the roll and pitch as control inputs.

Fig. 4(a) shows the *xy*-position of the *S-drone* during flight. The *S-Drone* stays inside a 0.2 m radius from the target *xy*-position, which is comparable to the performance of a human operator using a joystick to actuate pitch and roll. This deviation is reasonable for a UAV that uses onboard sensors for localization and does not use an external positioning system.

The *x*- and *y*-coordinates relative to the target position during flight are given in Figs. 4(b-c). For the *x* values, the maximum deviation from the target position (green line) during hover is 0.126 m and during take-off is 0.148 m. The mean deviation for the whole flight is approx. 13 cm. For the *y*-coordinate values, the maximum deviation from the target position during hover is 0.18 m and during take-off is 0.098 m. The mean deviation is approx. 16 cm. The slightly larger deviation on one axis compared to the other is because, in practice, the thrust of the rotors will never be perfectly balanced, and therefore some adjustment is required as the UAV gains altitude (i.e., it might pitch forward or backward).

The altitude (*z*-coordinate) values are given in Figs. 4(d-e). Fig. 4(d) shows the take-off altitude and the relative *z*-coordinate values during flight. Due to barometer and IMU sensor noise, the initial relative *z*-coordinate value during the experiment is 0.1 m. Therefore, the target position of 1 m above the initial take-off altitude is set at 1.1 m. The relative *z*-coordinate values of the UAV are around 1.1 m during the flight, with a maximum deviation of approx. 0.1 m and an average deviation of approx. 0.05 m. Fig. 4(e) shows the altitude values measured by the downward-facing single-point LiDAR. The take-off altitude read from the LiDAR is

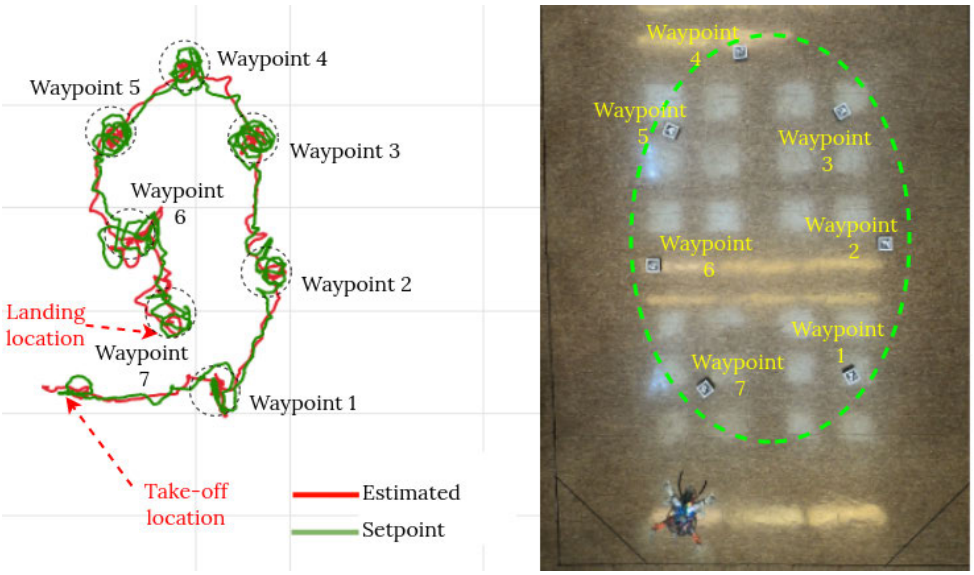


FIGURE 8. Autonomous vision-based waypoint navigation.

0.3 m (when the *S-Drone* is sitting on the ground). Therefore, a LiDAR reading of 1.3 m during flight indicates a true relative altitude of 1 m. The LiDAR altitude readings during hover are recorded around 1.3 m, again with a maximum deviation of approx. 0.1 m.

Another important metric for UAVs is attitude control performance during hover. The attitude controllers must keep the attitude angles (i.e., roll, pitch, and yaw angles) in certain intervals during hover. In Fig. 5(a-c), it can be seen that the recorded roll, pitch, and yaw values of the *S-drone* show limited oscillation, with an amplitude of approx 1.5 degrees. This amplitude of oscillation is small enough that it is not visually observable during flight.

The attitude control performance of a UAV is directly linked to the motor control signals. Fig. 5(e) reports the signals sent to the individual motors of the *S-drone* during flight. These values should usually be between the minimum and maximum configured PWM values (in this case, from 1 000 to 2 000). It can be seen from Fig. 5(e) that all signals are well within range and not too noisy.

Note that Fig. 5(e) is also an indication of the mass distribution of the specific *S-drone* that was built and used in this experiment (i.e., a second built *S-drone* will show slight differences). Fig. 5(e) shows that two motors (indicated as Output2 and Output3) operate with higher thrust on average than the other two, because of a slight imbalance in the UAV mass that can happen during assembly. This mass imbalance might reduce the maximum achievable thrust and will put more strain on some of the motors; however, the onboard flight controller automatically compensates for this so it does not impact flight stability.

Finally, another metric for UAVs is the unit step input response of the controllers. The step response is important because large and rapid deviations from the long-term steady state can potentially have extreme effects on a UAV. Formally, the step response of a controller provides information about the overall stability of the vehicle and its ability to reach a steady state. From a practical point of view, it is important to know how a UAV responds to sudden input changes. Fig. 6 shows the step response of the roll and pitch rate controllers of the *S-drone* with rising and settling times. The rise time of both the roll and pitch rate responses is approx. 0.09 s and the settling time is approx. 0.2 s without overshoot. In this demonstration, the responses of the *S-drone* were fast enough to maintain stability and the steady state errors remained relatively small.

B. SINGLE-UAV VISION-BASED NAVIGATION

This experiment validates autonomous flight and vision-based navigation with the *S-drone* in a GPS-denied environment. In this setup, the *S-drone* autonomously takes-off, hovers, and flies with a velocity of 0.2 m/s while performing waypoint navigation by guiding itself with fiducial markers (tags).

The flow chart of the control algorithm used for vision-based navigation is given in Fig. 7. After it receives a command to start and take-off, the *S-drone* flies upward until it reaches its target altitude and then begins searching for tags in order to start navigating the environment. When it finds a tag, the *S-drone* processes the frame, retrieves the tag information, estimates the position and orientation of the tag relative to itself, and prepares to fly towards the tag. It calculates a two-dimensional vector for its new target position (i.e., the tag position) relative to its current position. The target position is then provided to the Pixhawk4 flight controller and the *S-drone* moves towards the detected tag and hovers above its center (within a specified offset). After the first waypoint is reached, the *S-drone* looks for new tags. If the *S-drone* detects a new tag, it repeats the process

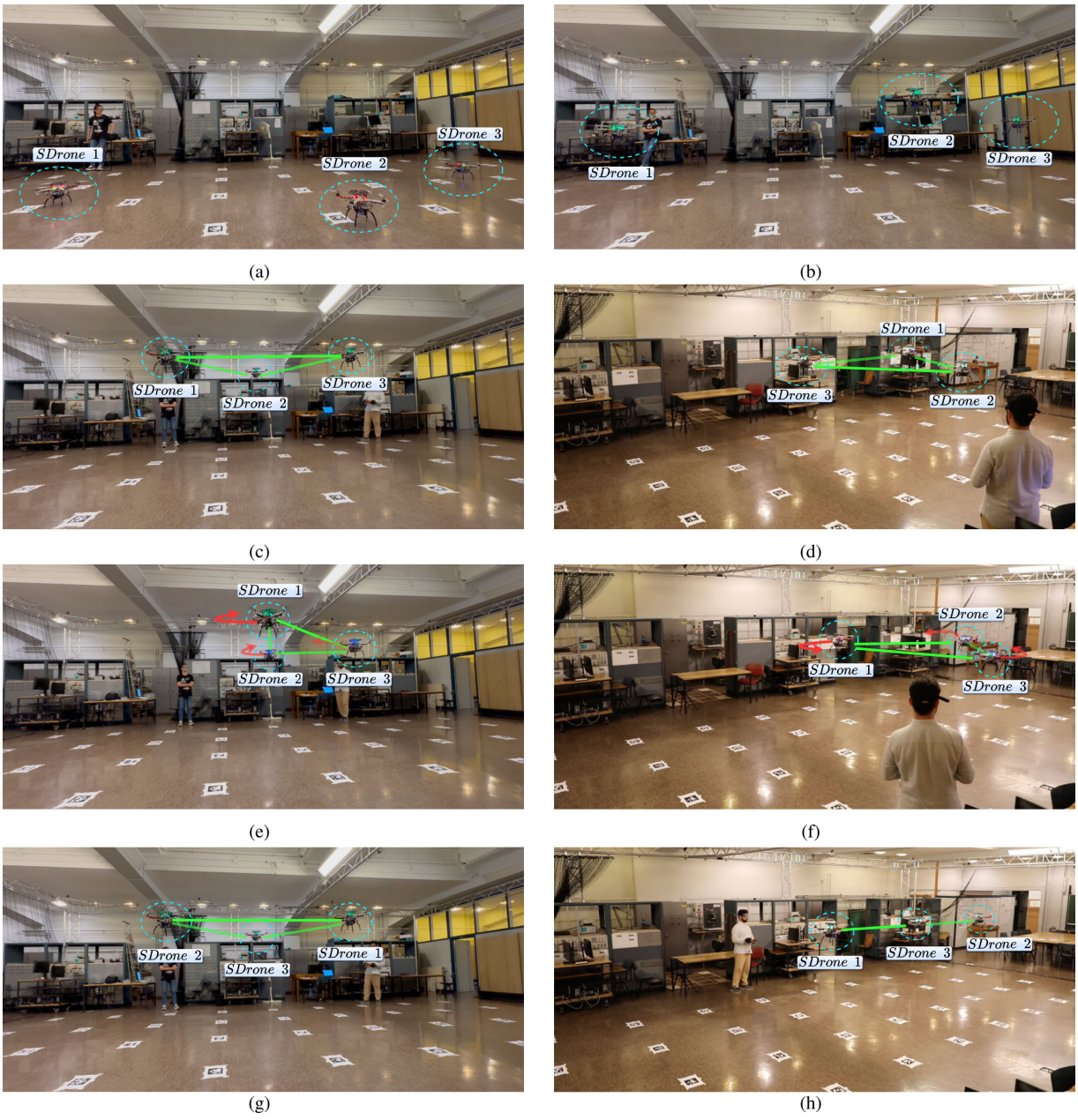


FIGURE 9. Demonstration of multi-UAV flight in a dynamic formation: (a) Initial random placement of the three UAVs. (b) Take-off procedure. (c) Establishment of a triangle formation. (d) Alternative perspective of the established triangle formation. (e) Start of formation rotation. (f) UAVs in rotation. (g) Completion of the rotation. (h) Final reconfiguration transitioning from triangular to linear formation.

described above. If no tag is detected at first, the *S-drone* waits for a certain amount of time and makes another detection attempt while hovering and holding its current position. After a certain number k of unsuccessful detection attempts, it decides to land.

In the experiment shown in Fig. 8, the *S-drone* uses this control algorithm to follow an ellipse-like path composed of seven waypoints. Fig. 8 shows the setup of this vision-based navigation experiment and reports the xy positions of the *S-drone* during flight.

C. SWARM ROBOTICS DEMONSTRATIONS

We conduct two multi-UAV demonstrations. The first is **Flying in a formation**, in which three *S-drones* fly together in a formation, while updating and rotating the shape, forming a series of target formations during flight. To establish, maintain, and reconfigure these formations, each *S-drone* needs to determine the current differences between (a) the real relative distances and relative orientations of neighboring *S-drones* and (b) the current targets. This inter-robot tracking is accomplished using our method for cooperative feature-level

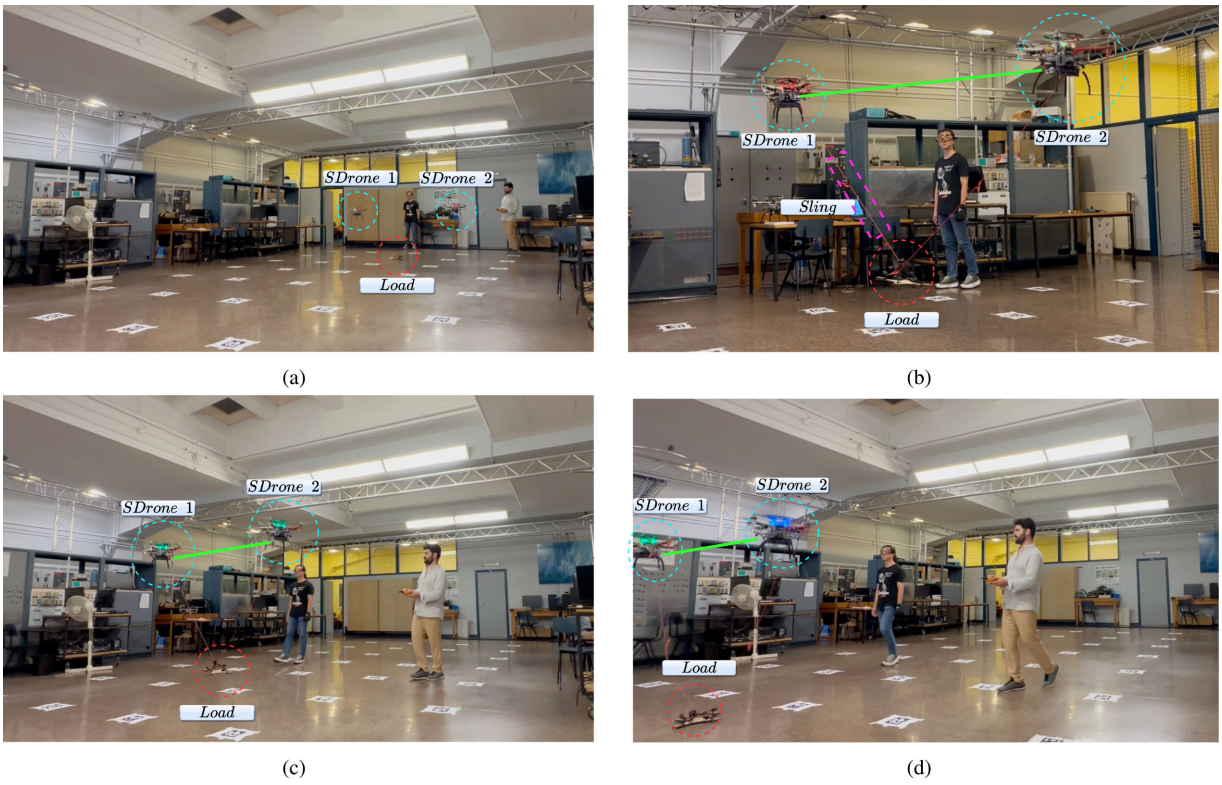


FIGURE 10. Demonstration of multi-UAV object transport: (a) Coordinated take-off. (b) UAVs navigating the environment. (c) Arrival at the target destination. (d) Coordinated descent ensuring safe payload placement.

sensor fusion between robots. Using this inter-robot tracking, each *S-drone* determines its relative spatial relationship with neighboring *S-drones* and attempts to reach its targets. This first demonstration (see Fig. 9 and the online movie[6]) shows the following procedure: **(1) Take-off:** Each of the three UAVs takes off, stabilizing near a target altitude of 1.5 m. **(2) Triangle Formation:** After take-off, the UAVs establish a triangular formation. **(3) Formation Rotation:** With the triangular formation shape intact, the UAVs undertake a synchronized turn. **(4) Line Formation:** The UAVs then reconfigure into a linear arrangement according to new targets for their relative positions and orientations.

The second demonstration is **Cooperative object transport**, in which two *S-drones* collaborate to transport a hung payload (i.e., an object hanging from two attachments, one to each *S-drone*). Here, the *S-drones* again use inter-robot tracking via cooperative feature-level sensor fusion to maintain their target relative spatial relationship, in this case while collaboratively lifting and transporting a hung payload. This demonstration (see Fig. 10) shows the following procedure: **(1) Take-off:** With the payload securely attached, the UAVs execute a coordinated take off, keeping the payload supported on both sides, and stabilizing near a target altitude of 1.5 m. **(2) Object Transport and Formation Maintenance:** After take-off, the UAVs move towards a target landmark while maintaining a side-by-side formation with fixed target UAV–UAV x, y distances. The UAVs need to continuously adjust to each other's movements in order to maintain this formation despite disturbances introduced because of the hung payload. **(3) Endpoint Descent:** When reaching the target landmark, the UAVs execute a coordinated landing to achieve gentle placement of the payload.

[6]https://osf.io/h9azb

V. CONCLUSION

We have presented the *S-drone*, a completely open-source UAV platform for swarm robotics research. The hardware platform supports swarm robotics research with UAVs using only onboard sensors and positioning. Future work could extend the support software for external disturbances (e.g., wind) to reduce the potential customization and tuning work for future researchers working in outdoor conditions.

ACKNOWLEDGMENT

(Mary Katherine Heinrich and Michael Allwright contributed equally to this work.)

REFERENCES

[1] M. Dorigo et al., "Swarmanoid: A novel concept for the study of heterogeneous robotic swarms," *IEEE Robot. Autom. Mag.*, vol. 20, no. 4, pp. 60–71, Dec. 2013.

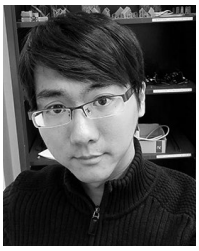
[2] T. Baca, M. Petrlik, M. Vrba, V. Spurny, R. Penicka, D. Hert, and M. Saska, "The MRS UAV system: Pushing the frontiers of reproducible research, real-world deployment, and education with autonomous unmanned aerial vehicles," *J. Intell. Robotic Syst.*, vol. 102, no. 1, pp. 1–28, May 2021.

[3] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu, and F. Gao, "Swarm of micro flying robots in the wild," *Sci. Robot.*, vol. 7, no. 66, May 2022, Art. no. eabm5954.
[4] M. Coppola, K. N. McGuire, C. De Wagter, and G. C. H. E. de Croon, "A survey on swarming with micro air vehicles: Fundamental challenges and constraints," *Frontiers Robot. AI*, vol. 7, p. 18, Feb. 2020.
[5] M. Schranz, M. Umlauft, M. Sende, and W. Elmenreich, "Swarm robotic behaviors and current applications," *Frontiers Robot. AI*, vol. 7, p. 36, Apr. 2020.
[6] M. Abdelkader, S. Güler, H. Jaleel, and J. S. Shamma, "Aerial swarms: Recent applications and challenges," *Current Robot. Rep.*, vol. 2, no. 3, pp. 309–320, Sep. 2021.
[7] H. Xu, Y. Zhang, B. Zhou, L. Wang, X. Yao, G. Meng, and S. Shen, "Omni-swarm: A decentralized omnidirectional visual-inertial-UWB state estimation system for aerial swarms," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3374–3394, Dec. 2022.
[8] A. Fishberg and J. P. How, "Multi-agent relative pose estimation with UWB and constrained communications," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 778–785.
[9] P. T. Morón, S. Salimi, J. P. Queralta, and T. Westerlund, "UWB role allocation with distributed ledger technologies for scalable relative localization in multi-robot systems," in *Proc. IEEE Int. Symp. Robotic Sensors Environments (ROSE)*, Nov. 2022, pp. 1–8.
[10] A. Valada, N. Radwan, and W. Burgard, "Deep auxiliary learning for visual localization and odometry," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 6939–6946.
[11] V. Walter, M. Vrba, and M. Saska, "On training datasets for machine learning-based visual relative localization of micro-scale UAVs," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 10674–10680.
[12] C. Chen, B. Wang, C. X. Lu, N. Trigoni, and A. Markham, "Deep learning for visual localization and mapping: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 2024, doi: 10.1109/TNNLS.2023.3309809.
[13] V. Walter, N. Staub, A. Franchi, and M. Saska, "UVDAR system for visual relative localization with application to leader-follower formations of multirotor UAVs," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2637–2644, Jul. 2019.
[14] A. Ahmad, V. Walter, P. Petrácek, M. Petrlík, T. Báca, D. Žaitlík, and M. Saska, "Autonomous aerial swarming in GNSS-denied environments with high obstacle density," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 570–576.
[15] D. Hert et al., "MRS drone: A modular platform for real-world deployment of aerial multi-robot systems," 2023, *arXiv:2306.07229*.
[16] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A system for autonomous flight using onboard computer vision," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 2992–2997.
[17] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision," *Auto. Robots*, vol. 33, nos. 1–2, pp. 21–39, Aug. 2012.
[18] L. Meier, D. Honegger, and M. Pollefeys, "PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 6235–6240.
[19] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 3400–3407.
[20] K. Mohta, M. Watterson, Y. Mulgaonkar, S. Liu, C. Qu, A. Makineni, K. Saulnier, K. Sun, A. Zhu, J. Delmerico, K. Karydis, N. Atanasov, G. Loianno, D. Scaramuzza, K. Daniilidis, C. J. Taylor, and V. Kumar, "Fast, autonomous flight in GPS-denied and cluttered environments," *J. Field Robot.*, vol. 35, no. 1, pp. 101–120, 2018.
[21] I. Sa, M. Kamel, M. Burri, M. Bloesch, R. Khanna, M. Popovic, J. Nieto, and R. Siegwart, "Build your own visual-inertial drone: A cost-effective and open-source autonomous drone," *IEEE Robot. Autom. Mag.*, vol. 25, no. 1, pp. 89–103, Mar. 2018.
[22] P. Foehn, E. Kaufmann, A. Romero, R. Penicka, S. Sun, L. Bauersfeld, T. Laengle, G. Cioffi, Y. Song, A. Loquercio, and D. Scaramuzza, "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *Sci. Robot.*, vol. 7, no. 67, Jun. 2022, Art. no. eabl6259.
[23] W. Giernacki, M. Skwierczynski, W. Witwicki, P. Wronski, and P. Kozierski, "Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering," in *Proc. 22nd Int. Conf. Methods Models Autom. Robot. (MMAR)*, Aug. 2017, pp. 37–42.
[24] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, M. Birattari, L. M. Gambardella, and M. Dorigo, "ARGoS: A modular, parallel, multi-engine simulator for multi-robot systems," *Swarm Intell.*, vol. 6, no. 4, pp. 271–295, Dec. 2012.

SINAN OĞUZ received the bachelor's degree in electrical and electronics engineering from Ankara University, in 2017. He is currently pursuing the joint Ph.D. degree with IRIDIA, the Artificial Intelligence Laboratory, and SAAS, the Department of Control Design and Systems Analysis, Université Libre de Bruxelles, Brussels, Belgium. Between September 2015 and April 2017, he worked on passive radar systems and radar tracking algorithms with the Estimation and Fusion Laboratory, Ankara University, through the Scientific and Technological Research Council of Turkey (T°TAK) undergraduate research fellowship. From April 2017 to February 2018, he was a full-time avionics Software Engineer with ASELSAN Inc., where he designed and programmed sensor fusion and state estimation algorithms for aircraft navigation systems. From February 2018 to October 2019, he was a full-time Research Engineer with STM Defense Inc., on autonomous UAVs, swarm intelligence, and GPS-less navigation. His primary research interests include swarm robotics, networked control systems, self-organizing networks, GPS-less navigation, and UAVs.

MARY KATHERINE HEINRICH received the B.Sc. degree from the University of Cincinnati, OH, USA, in 2013, the M.A.I. degree from IAAC, Universitat Politécnica de Catalunya, Barcelona, Spain, in 2014, and the Ph.D. degree from the Centre for Information Technology and Architecture (CITA), Royal Danish Academy, Copenhagen, Denmark, in 2019. From 2016 to 2018, she was a recurring Visiting Ph.D. Researcher with the New England Complex Systems Institute (NECSI), Cambridge, MA, USA; and from 2018 to 2019, she was a Research Associate with the Service Robotics Group, Institute of Computer Engineering, University of Lübeck, Luebeck, Germany. Since 2019, she has been a Postdoctoral Researcher with IRIDIA, the Artificial Intelligence Laboratory, Université Libre de Bruxelles, Brussels, Belgium. Her research interests include swarm intelligence, swarm robotics, and construction automation.

MICHAEL ALLWRIGHT received the Bachelor of Engineering degree in mechatronics and robotics from the Swinburne University of Technology, in 2011, and the Ph.D. degree in the field of swarm robotics from Paderborn University, in 2017. From 2017 to 2022, he continued his work in the field of swarm robotics as a Postdoctoral Researcher with Université Libre de Bruxelles. His research interests include swarm robotics, distributed systems, embedded systems, and software engineering. In 2022, he created a startup called Learn Robotics which aims to provide an easy-to-use platform for instructors and lecturers of robotics courses.



WEIXU ZHU received the bachelor's and master's degrees from the Department of Computer Science, Wuhan University, Wuhan, China, in 2014 and 2017, respectively. He is currently pursuing the Ph.D. degree with IRIDIA, the Artificial Intelligence Laboratory, Université Libre de Bruxelles, Brussels, Belgium. His research interests include swarm intelligence and swarm robotics.

MOSTAFA WAHBY received the Ph.D. degree in engineering from the University of Lübeck (UzL), in 2019. During his doctoral studies, he developed a collective robot-plant bio-hybrid system for plant shaping within the H2020 EU Project Flora robotica. From 2019 to 2020, he was a Postdoctoral Researcher in swarm robotics with Université Libre de Bruxelles. In 2020, he became a Postdoctoral Researcher at UzL, where he investigated training prediction models for dangerous road user behaviors that would serve as an early warning system for autonomous vehicles. His research interests include machine learning, evolutionary algorithms, and swarm robotics.

EMANUELE GARONE (Member, IEEE) received the master's degree (Laurea) and the Ph.D. degree in systems engineering from the University of Calabria, Rende, Italy, in 2005 and 2009, respectively. Since 2010, he has been an Associate Professor with the Control Design and System Analysis Department, Université Libre de Bruxelles, Brussels, Belgium. His research interests include constrained control, reference governors, networked control systems, and aerial robotics. In 2014, he received an honorable mention for the Young Author Prize of the 19th IFAC World Congress. He has authored more than 160 contributions in peer-reviewed conferences and journals.

MARCO DORIGO (Fellow, IEEE) received the Laurea, Master of Technology degree in industrial technologies engineering, and the Ph.D. degree in electronic engineering from the Politecnico di Milano, Milan, Italy, in 1986 and 1992, respectively, and the title of Agrégé de l'Enseignement Supérieur from Université Libre de Bruxelles (ULB), Brussels, Belgium, in 1995.

From 1992 to 1993, he was a Research Fellow with the International Computer Science Institute, Berkeley, CA, USA. In 1993, he was a NATO-CNR Fellow, and from 1994 to 1996, he was a Marie Curie Fellow. Since 1996, he has been a tenured Researcher of the FNRS, the Belgian National Funds for Scientific Research, and the Co-Director of IRIDIA, the Artificial Intelligence Laboratory, ULB. He is an inventor of the Ant Colony Optimization metaheuristic. His current research interests include swarm intelligence, swarm robotics, and metaheuristics for discrete optimization.

Prof. Dorigo was a recipient of the Italian Prize for Artificial Intelligence, in 1996, the Marie Curie Excellence Award, in 2003, the Dr. A. De Leeuw-Damry-Bourlart Award in Applied Sciences, in 2005, the Cajastur International Prize for Soft Computing, in 2007, an ERC Advanced Grant, in 2010, the IEEE Frank Rosenblatt Award, in 2015, and the IEEE Evolutionary Computation Pioneer Award, in 2017. He is a fellow of AAAI and EurAI. He is the Founding Editor of *Swarm Intelligence* and an associate editor or a member of the editorial boards of several journals on computational intelligence and adaptive systems.