Conference Paper

# Fünfliber-Drone: A Modular Open-Platform 18-grams Autonomous Nano-Drone

**Author(s):**
Müller, Hanna; Palossi, Daniele; Mach, Stefan; Conti, Francesco; Benini, Luca

**Publication Date:**
2021-02

**Permanent Link:**

https://doi.org/10.3929/ethz-b-000470533 →

# Fünfliber-Drone: A Modular Open-Platform 18-grams Autonomous Nano-Drone

Hanna Müller*, Daniele Palossi*†, Stefan Mach*, Francesco Conti‡, and Luca Benini*‡
*Integrated Systems Laboratory - ETH Zürich, Switzerland
† Dalle Molle Institute for Artificial Intelligence - University of Lugano and SUPSI, Switzerland
‡ Department of Electrical, Electronic and Information Engineering - University of Bologna, Italy
Email: hanmuell, dpalossi, smach, fconti, lbenini@ethz.ch

***Abstract*—Miniaturizing an autonomous robot is a challenging task – not only the mechanical but also the electrical components have to operate within limited space, payload, and power. Furthermore, the algorithms for autonomous navigation, such as state-of-the-art (SoA) visual navigation deep neural networks (DNNs), are becoming increasingly complex, striving for more flexibility and agility. In this work, we present a sensor-rich, modular, nano-sized Unmanned Aerial Vehicle (UAV), almost as small as a five Swiss Franc coin – called Fünfliber – with a total weight of 18g and 7.2cm in diameter. We conceived our UAV as an open-source hardware robotic platform, controlled by a parallel ultra-low power (PULP) system-on-chip (SoC) with a wide set of onboard sensors, including three cameras (i.e., infrared, optical flow, and standard QVGA), multiple Time-of-Flight (ToF) sensors, a barometer, and an inertial measurement unit. Our system runs the tasks necessary for a flight controller (sensor acquisition, state estimation, and low-level control), requiring only 10% of the computational resources available aboard, consuming only 9mW – 13x less than an equivalent Cortex M4-based system. Pushing our system at its limit, we can use the remaining onboard computational power for sophisticated autonomous navigation workloads, as we showcase with an SoA DNN running at up to 18Hz, with a total electronics' power consumption of 271mW.**

***Index Terms*—Autonomous UAV, CNNs, nano-UAV, ultra-low-power.**

## I. Introduction

In the last decade, autonomous Unmanned Aerial Vehicles (UAVs) have reached enormous popularity and diffusion due to their wide range of application, including aerial inspection of industrial and hazardous areas [1], precise agriculture [2], and entertainment [3], just to name a few. One of the major trends in the evolution of UAVs is their miniaturization, with commercially available pocket-size drones [4] and futuristic insect-scale robots actively studied by pioneering research groups [5], [6]. The vision of small-form-factor intelligent drones can be a game-changing aspect in many real-life use cases. Such robotic platforms would offer several advantages, as increased flexibility for reaching otherwise inaccessible places, reduced production cost, and enhanced safety, enabling novel human-machine interactions usually infeasible with big-size drones.

However, the miniaturization of these helpful and versatile robots comes at the price of reduced onboard intelligence and lack of autonomous navigation capabilities due to their limited power envelope. To date, the smallest class of UAVs available on the marketplace, namely nano-sized drones, is characterized by a few centimeters in diameter, a few tens of grams in weight, and a resource-constrained single-core microcontroller unit (MCU) onboard. This class of devices cannot afford complex state of the art (SoA) vision-based workloads that would enable sophisticated autonomous navigation capabilities, as found in their standard-size counterpart UAVs [3], [7].

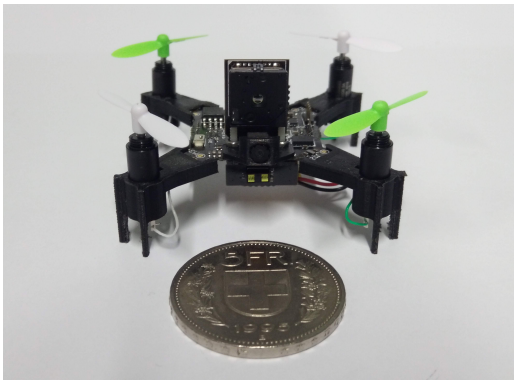


Fig. 1: Our sensor-rich nano-sized drone prototype, compared in size with a five Swiss Franc coin, called Fünfliber.

The vast majority of commercial and academic SoA nano-sized UAVs are severely constrained in their onboard computational power, including limited availability of floating-point units (FPUs), memory capacity and latency, and sensing/perception support. A common strategy to bypass these limitations is to take advantage of several off-board aids, from ad-hoc expensive localization infrastructure to offloaded computation to powerful remote base-stations or servers [7]–[9]. However, streaming of high-bandwidth data for vision-based navigation introduces additional drawbacks: additional radio power consumption, limited range of operation, unpredictable latency, cyber-security vulnerabilities, and many more. Reliance on closed-source hardware and software frameworks exacerbates these issues, making research on alternative solutions extremely challenging.

To the best of our knowledge, there are only a few examples that diverge from this scenario. The commercial off-the-shelf (COTS) Crazyflie 2.0 nano-quadrotor[1] represents a valuable open-source platform, but still offering limited onboard processing power, i.e., a single-core Cortex-M4 MCU, and limited sensing capabilities, e.g., no availability of cameras. A second

This work has been partially funded by the Swiss National Science Foundation (SNSF) Spark grant (190880) and the SNSF BRIDGE Discovery grant (187087).

[1]https://www.bitcraze.io

TABLE I: Rotorcraft UAVs taxonomy by vehicle class-size.

| Vehicle Class | Ø: Weight $[cm{:}Kg]$ | Power $[W]$ | Onboard Device |
|---|---|---|---|
| *std-size* [7] | $\sim 50 : \geq 1$ | $\geq 100$ | Desktop |
| *micro-size* [15] | $\sim 25 : \sim 0.5$ | $\sim 50$ | Embedded |
| *nano-size* [16] | $\sim 10 : \sim 0.01$ | $\sim 5$ | MCU |
| *pico-size* [13] | $\sim 2 : \leq 0.001$ | $\sim 0.1$ | ULP |

example is given by the open-source work introduced in [10], [11], where a Crazyflie 2.0 has been extended with a companion electronic board featuring an additional parallel ultra-low-power (ULP) MCU, a low-resolution camera, and off-chip RAM, enabling the real-time execution of a complex deep learning algorithm for autonomous driving.

In this work, we introduce the smallest (to the best of our knowledge) open-software and open-hardware nano-sized UAV with onboard real-time capability for computationally intensive SoA autonomous navigation algorithms, in a form factor $\sim 2\times$ smaller than a Crazyflie 2.0 and featuring a unique set of integrated sensors. Our nano-drone, comparable in size to a five Swiss Franc coin (called Fünfliber), is conceived to be a modular and flexible platform with a plethora of pluggable sensors for an 18-gram nano-sized UAV. Its rich set of sensors consists of three different cameras, i.e., a QVGA CMOS, an infrared (IR), and an optical flow (OF) camera, six time-of-flight (ToF) sensors (one per each direction), a barometer, and an inertial measurement unit (IMU), significantly improving the sensory capabilities over the SoA for this class of robots. The brain of our platform is embodied by a second-generation parallel ultra-low power system on a chip (SoC) called *Mr.Wolf* [12], and by $8\,\mathrm{MB}$ of off-chip RAM. The Mr.Wolf SoC is a powerful (up to 8 GOPS) octa-core general-purpose digital processing architecture with floating-point hardware support.

Our results show that the proposed robotic system can scale its operation from a basic Flight Controller (FL-Ctrl), with essential inertial sensors, consuming $\sim 30\,\mathrm{mW}$ for all electronics up to computationally autonomous, deep-learning driven visual navigation running at $18\,\mathrm{Hz}$ and consuming $271\,\mathrm{mW}$ for all electronics, less than 10% of the overall UAV power. Since $18\,\mathrm{Hz}$ is more than $2\times$ the real-time constraint of the visual navigation pipeline [10], this result leaves space for even more complex tasks to be offloaded to the robot, paving the way for multi-task autonomous navigation on sub-20 grams nano-drones.

## II. Related Work

To enable autonomous navigation capabilities aboard nano-UAVs, two major constraints must be addressed: *i*) sub-Watt computational power envelope and *ii*) limited payload, i.e., a few grams. Wood et al. [13] estimate that only up to $5\%$ of the total power consumption of a UAV is available for onboard computation, while the allotted payload for electronics is $< 25\%$ of the total mass. Table I shows an overview of weight, power envelope, and affordable onboard devices for four main class-size of vehicles, as introduced in [14]. Achieving a level of autonomy comparable to desktop CPUs/GPUs is challenging for nano/pico-UAVs, relying on MCU-class computing.

Focusing on nano-size vehicles, one group of solutions bypasses the onboard limitations using off-board computation. Offloading computation to remote base-stations can enable complex algorithms fed with abundant streams of sensory data, such as in vision-based workloads [8], [9]. In [8], visual-inertial simultaneous localization and mapping (SLAM) and external pose-estimation are computed on a remote base-station, demonstrating precise autonomous flight. The authors of [9] focus on DNN-based collision avoidance, demonstrating it on a Crazyflie with off-board inference on a computer connected wirelessly. The need for high-frequency data streaming limits these systems their operating range and control latency, causing low reliability. Pushing off-board computation to the extreme, some UAVs rely on external computation for sensing and low-level control as well [4], [6], limiting their applicability to research labs with special infrastructure.

An alternative trend to overcome the limited computational power of MCU-based nano-UAVs is given by the advent of onboard visual navigation accelerators [17], [18]. Dedicated ASICs are used to accelerate computationally intensive tasks, such as visual odometry [17] or SLAM [18], within the stringent power envelope of a nano-UAV. However, this approach does not simplify the platform's design as it still needs a MCU-like processor for interfacing with sensors and low-level control tasks. By contrast, our modular platform features an onboard SoC that unifies these needs, offering enough computational power and sensor interfaces to run both low-level flight controllers and complex SoA visual navigation workloads in real-time.

A last group of solutions tries to squeeze as many operations as possible in the available MCU aboard nano-UAVs [6], [16]. These approaches pay the price of simplified algorithms and limited functionality. The system proposed in [6] requires only an 8-bit PIC MCU, but its application is limited to indoor environments where all walls show specific visual patterns. A 40-grams pocket drone introduced in [16] performs obstacle avoidance and velocity control but is limited to low speed, i.e., $0.3\,\mathrm{m/s}$. A similar approach is proposed in [10], where the Cortex-M4 MCU of a Crazyflie nano-drone was extended with a custom companion board featuring a ULP camera and a commercial ULP parallel SoC called GAP-8 [19]. Such a system was able to run a complex DNN for autonomous navigation, called *PULP-DroNet*, in real-time, without any external aids. However, the MCU was still in charge of all low-level tasks (e.g., FL-Ctrl, state estimation, etc.), and the DNN was optimized to comply with fixed-point arithmetic, due to the lack of FPUs on the SoC. In this work, we improve upon the SoA of [10] in terms of both integration and sensing/compute capabilities. The open-source robotic platform we propose is $2\times$ smaller than a Crazyflie, features significantly improved sensing capabilities to a Crazyflie, and uses a single SoC, called *Mr.Wolf* [12]. Mr.Wolf delivers enough performance to run low-level control tasks and high-level intelligence, such as the PULP-DroNet DNN, on the same SoC, at $2.4\times$ higher energy efficiency than GAP-8 [19] and with support for floating-point operations.

## III. Fünfliber-drone Design

Any autonomous robot needs to follow a life cycle based on three main functional blocks, as shown in Figure 2. The *state estimation* computes the system's current state $A$, then the

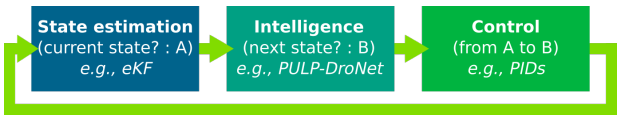



Fig. 2: Main functional blocks of any autonomous robot.

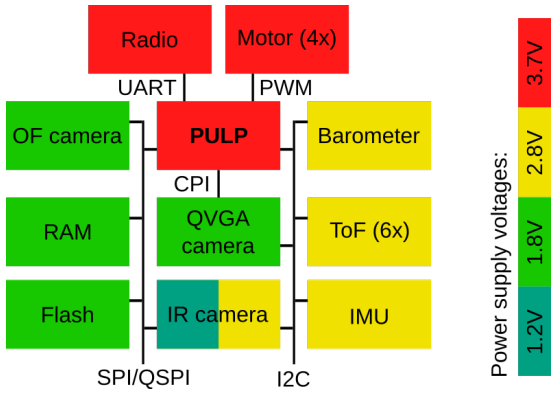
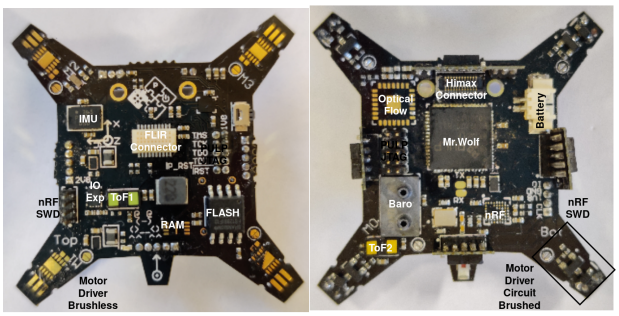



Fig. 3: Hardware overview of the Fünfliber-drone.

*intelligence* task determines in which state *B* the system should go to fulfil its mission, and finally the *control* block derives a control law that brings the system from state *A* to *B*. Among these three blocks, the one making a robot autonomous is the onboard intelligence (e.g., PULP-DroNet), as the state estimation (e.g., an extended Kalman filter – eKF) and control (e.g., PID controller) are required also by non-autonomous robots. This section first explains the proposed system's hardware design and the required software optimizations to fit all three main functional blocks in a small, lightweight, and energy-efficient nano-UAV. In the second part, we describe our state estimator, control tasks, and all the mechanisms required to orchestrate and synchronize the execution of such a complex multi-task system. Finally, we show the possibilities of our open-source robotic platform with a sample SoA DNN application.

### A. Hardware architecture

The heart of our system is Mr.Wolf [12], a second-generation parallel ultra low-power processor (PULP) SoC featuring various interfaces and nine cores. Mr.Wolf is based on the PULP open-source architecture and the RISC-V open ISA. The Mr.Wolf SoC includes two power domains - the *Fabric Controller* (FC) and the *cluster* (CL) domain. The FC domain consists of a low-power single-core MCU featuring a wide range of peripherals such as Quad SPI, I2C, a parallel camera interface (CPI), UART, PWM, GPIOs, and JTAG for debugging. The FC is built around a tiny zero-RISCY RV32IMC processor, designed for lightweight integer-only control purposes [20]. The CL domain, on the other hand, is designed for heavy-duty integer and floating-point calculations. It contains eight RI5CY cores with RV32IMFCXpulp ISA. The xPULP extensions include common signal processing instructions such as hardware loops, automatic pointer increment, and SIMD arithmetics on low-bitwidth integer data. The 8 RI5CY cores share one FPU and two fused multiply-add (FMA) units, fostering floating-point signal processing algorithms.




(a) Top view (b) Bottom view

Fig. 4: PCB layout of the proposed PULP-based nano-UAV

Around this highly capable computational unit, we build our sensor-rich system, whose key building blocks are shown in Fig. 3. 16 MB of quadSPI Flash (at up to 80 MB/s) and 8 MB of DRAM (at up to 40 MB/s) are available to the SoC. A BMI088 IMU (16-bit, triaxial accelerometer and gyroscope, up to 2 kHz), a BMP380 barometer (±0.50 hPa absolute accuracy, up to 200 Hz), as well as six VL53L1X ToF sensors facing all directions (up to 4 m distance measurement at 50 Hz) provide sensing of the UAV's environment. A downward-facing PMW3901 OF sensor (minimum distance 80 mm), a front-facing HM01B0 camera (QVGA, greyscale, up to 60 $fps$), and a front-facing IR camera provide the system with vision. Finally, an nRF51822 radio IC enables off-board communication.

Fig. 4 shows the PCB layout of our PULP-based UAV, highlighting its main components. Fig. 4a shows the top view, featuring mainly sensors, memory, and a plug for the IR camera. Fig. 4b shows the bottom view, featuring the remaining sensors, the Mr.Wolf SoC, and the radio IC. The four remaining ToF sensors on the sides and the IR camera are on separate PCBs to be assembled orthogonally to the body frame.

### B. Software architecture

Our Flight Controller (FL-Ctrl) is inspired by Crazyflie's software architecture and features sensor acquisition, state estimation, control loops, and motor conversion tasks. In Table II, we provide a hardware/software comparison between our Fünfliber-drone and the Crazyflie platform. The first key difference is represented by the SoC's computational power, hosting nine general-purpose cores running at higher frequencies compared to the single-core STM32F405 MCU. Also, the memory availability marks a considerable difference increasing both the on-chip low-latency memories and the off-chip ones. Both MCUs have hardware FPUs, but it is accessible only from the cluster domain on Mr. Wolf. From an operating system (OS) point of view, the main difference is the absence of any preemption mechanism on the Mr.Wolf SoC OS, called pulpOS. This limitation forces the programmer to optimize the tasks' scheduling carefully and set a lower-bound on the minimum SoC's frequency for which the system can operate without mapping every task in a dedicated interrupt service routine.

In Fig. 5, we show an overview of the software pipeline, where we highlight the tasks running on both FC and CL domains. The first stage is the IMU sensor data acquisition at 1 kHz, averaged

TABLE II: Comparison between STM32F405 and Mr.Wolf SoC.

| | STM32F405 | Mr.Wolf |
|---|---|---|
| Cores | 1 ARM Cortex-M4F | 1x RV32IMC + 8x RV32IMFCXpulp |
| Flash | 1 MB | (16 MB ext.) |
| RAM | 192 kB | L1: 64 kB, L2: 512 kB, (L3: 8 MB ext.) |
| Freq. | 168 MHz | up to FC@450 MHz/CL@350 MHz |
| FPU | Yes | FC: No; CL:Yes |
| OS | FreeRTOS | pulpOS (no preemption) |

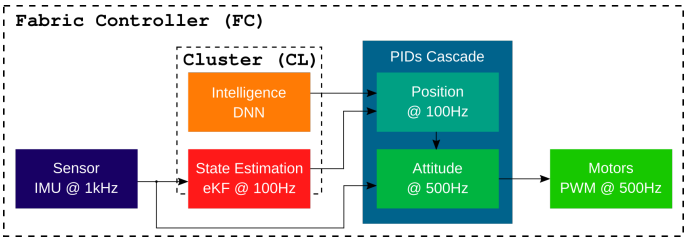



Fig. 5: Diagram of the software pipeline (FL-Ctrl + intelligence).

at 100 Hz, and fed to the state estimation task: an extended Kalman filter (eKF) [21]. The eKF runs partially on the FC, but mostly on the cluster at 100 Hz enabling efficient use of both FPUs and all eight cores. The control part is implemented as a soft-float PID cascade running on the FC to leave sufficient "computational space" on the CL for additional heavy workloads (e.g., intelligence in the form of a deep neural network (DNN)). The PID cascade is implemented with two stages running at different frequencies: the position PID runs at 100 Hz, while the attitude PID runs at 500 Hz, stabilizing the drone.

The scheduling scheme for the FL-Ctrl tasks is reported in Fig. 6, where the main loop runs as fast as the sensor acquisition task (IMU), i.e., 1 kHz. All other tasks are periodically scheduled according to their predefined frequencies, leaving sufficient execution time, on both the FC and the CL, to schedule additional tasks, such as DNN-based autonomous navigation.

### C. Floating-Point Support

Many algorithms running on a drone require or benefit from floating-point operations. For example, the eKF used for state estimation is significantly more robust when executed natively in floating-point instead of using fixed-point computation with a predetermined dynamic range. Floating-point code is also easier to write, maintain, and port to different architectures. To take advantage of Mr.Wolf's architecture, we choose to deploy low compute intensity floating-point code (such as the update of a PID controller) on the single-core FC, using a soft-float implementation. This avoids cluttering the cluster with

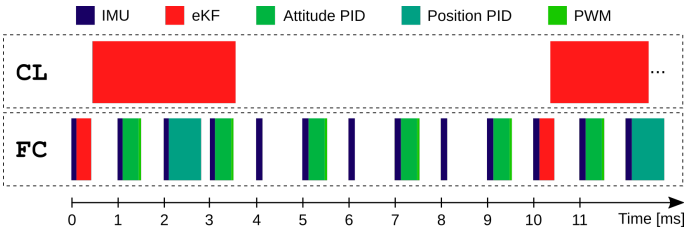



Fig. 6: Scheduling scheme of the basic tasks composing our FL-Ctrl – both FC and CL @100 MHz.

operations not requiring high performance, keeping it free for heavier calculations such as those required by the eKF for state estimation. Taking advantage of the cluster's multi-core architecture, heavy-duty workloads can be executed at high speed enabling fast and adaptive control of the nano-drone.

### D. Deep Neural Network on PULP

The chips' architecture with one FC and an eight-core cluster allows the execution of DNNs in a small power envelope. Palossi et al. [10] introduced a visual navigation engine called *PULP-DroNet* for a commercial PULP SoC called GAP8. DroNet [7] originally learned how to fly from an autonomous driving dataset, predicting steering angles for following roads (or what it generalizes as roads, e.g. a corridor) and a collision probability to scale the speed and avoid obstacles. As Mr.Wolf features the same eight cluster cores as GAP8 and achieves 2× higher performance at 2.4× higher energy-efficiency, we can expect even better results for running DroNet on Mr.Wolf.

## IV. Results

In this section, we evaluate our Fünfliber-drone experimentally, focusing on *i*) its physical properties and power consumption; *ii*) a functional evaluation comparing the FL-Ctrl accuracy with the reference Crazyflie implementation.

### A. System Evaluation

*1) System Weight:* In its default configuration, our drone weighs 18 g; the breakdown is shown in Fig. 8-A. Motors, battery, and the drone frame make up 78% of the system's weight, with electronics only contributing 22%. Thanks to the modularity of our design, many components can be replaced with alternatives: e.g., a battery pack with a different capacity, or any PWM-driven motors producing sufficient thrust. If unused, the IR camera can be unplugged to save weight.

*2) Flight Controller Performance:* The FC of our SoC is capable of running up to 450 MHz, while the multi-core cluster reaches 350 MHz. The FL-Ctrl software runs on the FC, offloading only the intensive floating-point part of the eKF task to the cluster. The software pipeline on the FC consists of data acquisition, part of the eKF state estimation, control loops (i.e., position/attitude PIDs), and the motor distribution tasks. Fig. 8-B shows the computational load in active cycles per second of each task running on the FC, requiring a minimum clock speed of 46 MHz to complete all periodic tasks with the desired frequency. Even if the sensor acquisition runs at a higher frequency (1 kHz), the attitude PID task constitutes the most significant part of active cycles per second (17.5MCycle/s). By increasing the FC clock frequency, we can gain additional computation resources on the FC while paying for it in increased power consumption. Up to 404 Mcycle/s, corresponding to 89.8% of the available FC load at maximum frequency are free for additional tasks for under 40 mW total power consumption of the FC. On the cluster, execution of the eKF requires 31 kcycle/s, representing a cluster load of merely 9% at its maximum frequency of 350 MHz. The remaining available 91% of compute resources can be leveraged for intelligent applications such as DNN-based visual navigation. Thanks to our higher clock speeds, we are

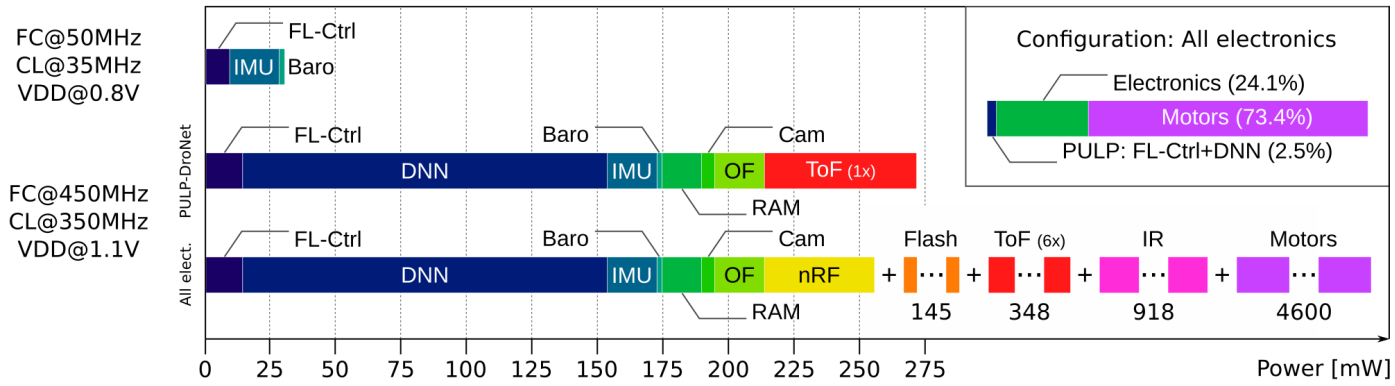



Fig. 7: Power breakdown of the Fünfliber-drone in three different configurations. From top to bottom, (*i*) only the FL-Ctrl and its sensors, (*ii*) the PULP-DroNet sample application, and (*iii*) all electronics in their active state (worst-case).

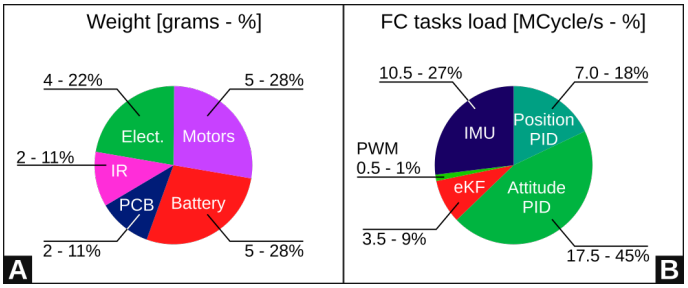



Fig. 8: Weight (A) and FC's active cycles (B) breakdown of the Fünfliber-drone.

able to run the SoA DNN used in [10] alongside the flight controller workload, achieving the same 18 Hz throughput.

*3) Power Consumption:* We evaluate our system's power consumption in three different configurations, as shown in Fig. 7. In addition to processor measurements, we estimate the power consumption of additional components from their respective data sheets and assume a conservative voltage converter efficiency of 87%. The top row of Fig. 7 shows only the flight controller running on PULP at 50MHz (FC) and 35MHz (cluster), with only the IMU and barometer active, consuming a total of 30 mW. In the middle row, the system executes PULP-DroNet [10] application, a visual-based navigation engine at 18MHz. PULP is operating at its maximum speed and the IMU, barometer, camera, optical flow sensor, and one ToF sensor are active, requiring only 271 mW. Finally, the bottom row shows PULP running at its maximum speed and all sensors on and measuring, which consumes 1666 mW of system power. This is a theoretical worst-case scenario, as continuous measurements using all sensors concurrently is neither supported nor required for operation. The upper right corner of Fig. 7 highlights the dominating power consumption of the motors (measured at 50% duty-cycle), requiring 73% of system power while the processor under full load only accounts for 2.5%.

## B. Functional Evaluation

To confirm our system's proper functionality, we recorded sensor data from a Crazyflie drone, replayed them on our drone's flight controller, and compare eKF and PID outputs.

*1) eKF:* For the eKF test, we recorded sensor data and filter outputs of a short flight on the CF at 100Hz. We

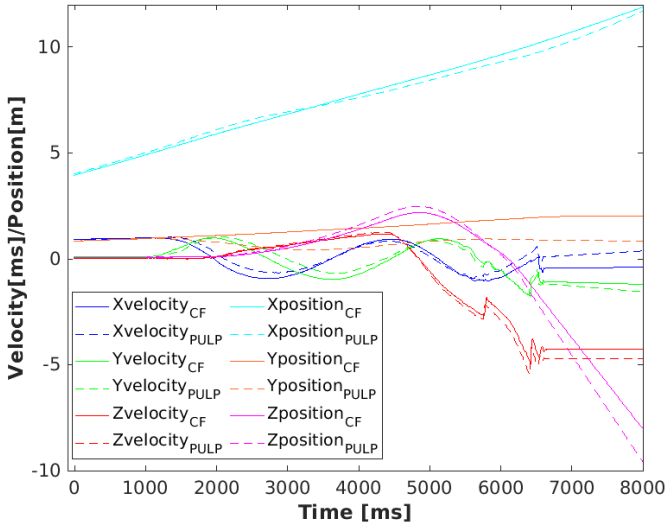



Fig. 9: IMU input data and eKF output comparison between Crazyflie (CF) and PULP-drone.

show the IMU (3-axis accelerometer and 3-axis gyroscope) measurements, originally sampled at 1kHz but averaged at 100Hz, and the comparison between the output, position, and velocity estimations, in Fig. 9. The values drift slightly apart over time due to the different logging mechanisms used: On the Crazyflie, the already implemented SD-card logging which captures the current values of the tracked variables at 100Hz was used, while on the PULP-drone, we log the results after each execution of the eKF, leading to accumulating small time differences. After 8 seconds, the position and velocity states are still inside 1.2m and 0.75m/s of the baseline, respectively.

*2) PIDs:* For the PIDs test, we recorded state estimation and gyroscope values as well as PID output values with a constant setpoint (turning on the yaw-axis with 0.1Hz at 1m over the ground) on the Crazyflie at 1kHz. We show the comparison between the rate PID outputs, split in proportional (P), integral (I), and derivative (D) parts, in Fig. 10. Due to the high logging frequency on the Crazyflie (1kHz), we miss some data points, leading to jumps in the gyroscope values and with this to spikes on the derivative parts of the rate PIDs. The mean average error of the PID outputs, in % of the maximum amplitude of the baseline (Crazyflie) values, is shown in Table III.

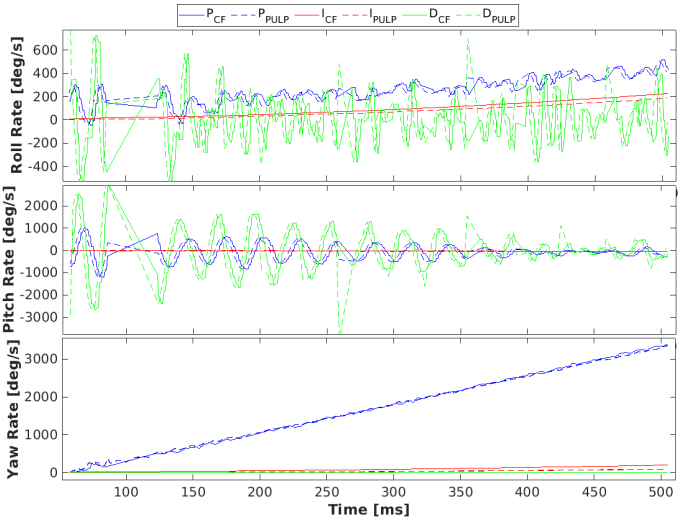



Fig. 10: PID controller output comparison between Crazyflie (CF) and PULP-drone.

TABLE III: Mean average error of PID outputs, in % of maximum amplitude of the baseline (Crazyflie) values.

| | Attitude | | | Rate | | |
|---|---|---|---|---|---|---|
| | **Roll** | **Pitch** | **Yaw** | **Roll** | **Pitch** | **Yaw** |
| **P** | 5.07e-1% | 7.59e-5% | 2.15e-1% | 1.76% | 1.56% | 0.76% |
| **I** | 7.29% | 6.87% | 10.35% | 6.82% | 4.03% | 9.45% |
| **D** | - | - | 1.39% | 6.88% | 3.39% | - |

## V. Conclusion

We present a sensor-rich, modular, nano-size UAV with a total weight of 18g and 7.2cm in diameter – 2× smaller than the SoA, but enriched with significantly improved sensing and onboard computing capabilities. We conceived our UAV as a fully open-source hardware robotic platform, controlled by a PULP SoC with a wide set of onboard sensors, including three cameras (i.e., infrared, optical flow, and standard QVGA), multiple time-of-flight (ToF) sensors, a barometer, and an inertial measurement unit. For basic flight control (sensor acquisition, state estimation, and low-level control), the proposed robotic system needs only 10% of the computational resources available in its lowest power operating point, consuming just 9 mW – 13× less than an equivalent Cortex M4-based system. Pushing our system to its limits, we show that we can run a sophisticated autonomous navigation pipeline, the SoA DroNet DNN, on top of the flight controller. It can run at up to 18 Hz, with a total electronics' power consumption of 271mW - below 6% of the total power consumed by the drone. This initial result paves the way for the deployment of advanced navigation and environment sensing tasks on top of sub-20g nano-drones; to foster academic research in this field, we release our system's design as open-source at *https://github.com/pulp-platform/fuenfliber*.

## Acknowledgments

The authors thank N. Brun for his support in making the motor-holders and V. J. K. Morinigo for his assistance in taking the prototype's pictures.

## References

[1] S. Jordan, J. Moore, S. Hovet, J. Box, J. Perry, K. Kirsche, D. Lewis, and Z. T. H. Tse, "State-of-the-art technologies for uav inspections," *IET Radar, Sonar & Navigation*, vol. 12, no. 2, 2018.

[2] U. R. Mogili and B. Deepak, "Review on application of drone systems in precision agriculture," *Procedia computer science*, vol. 133, 2018.

[3] E. Kaufmann, A. Loquercio, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza, "Deep drone acrobatics," 2020.

[4] M. Piccoli and M. Yim, "Piccolissimo: The smallest micro aerial vehicle," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

[5] R. J. Wood, "The first takeoff of a biologically inspired at-scale robotic insect," *IEEE Transactions on Robotics*, vol. 24, no. 2, 2008.

[6] J. Zufferey, A. Klaptocz, A. Beyeler, J. Nicoud, and D. Floreano, "A 10-gram microflyer for vision-based indoor navigation," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[7] A. Loquercio, A. I. Maqueda, C. R. del-Blanco, and D. Scaramuzza, "Dronet: Learning to fly by driving," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, 2018.

[8] O. Dunkley, J. Engel, J. Sturm, and D. Cremers, "Visual-inertial navigation for a camera-equipped 25g nano-quadrotor," in *IROS2014 aerial open source robotics workshop*, 2014.

[9] K. Kang, S. Belkhale, G. Kahn, P. Abbeel, and S. Levine, "Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

[10] D. Palossi, F. Conti, and L. Benini, "An open source and open hardware deep learning-powered visual navigation engine for autonomous nano-uavs," in *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2019.

[11] D. Palossi, A. Loquercio, F. Conti, E. Flamand, D. Scaramuzza, and L. Benini, "A 64-mw dnn-based visual navigation engine for autonomous nano-drones," *IEEE Internet of Things Journal*, vol. 6, no. 5, 2019.

[12] A. Pullini, D. Rossi, I. Loi, G. Tagliavini, and L. Benini, "Mr.wolf: An energy-precision scalable parallel ultra low power soc for iot edge processing," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 7, 2019.

[13] R. J. Wood, B. Finio, M. Karpelson, K. Ma, N. O. Pérez-Arancibia, P. S. Sreetharan, H. Tanaka, and J. P. Whitney, *Progress on "Pico" Air Vehicles*. Cham: Springer International Publishing, 2017. [Online]. Available: https://doi.org/10.1007/978-3-319-29363-9_1

[14] D. Palossi, A. Marongiu, and L. Benini, "Ultra low-power visual odometry for nano-scale unmanned aerial vehicles," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, 2017.

[15] J. Conroy, G. Gremillion, B. Ranganathan, and J. S. Humbert, "Implementation of wide-field integration of optic flow for autonomous quadrotor navigation," *Autonomous robots*, vol. 27, no. 3, 2009.

[16] K. McGuire, G. de Croon, C. De Wagter, K. Tuyls, and H. Kappen, "Efficient optical flow and stereo vision for velocity estimation and obstacle avoidance on an autonomous pocket drone," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, Apr 2017. [Online]. Available: http://dx.doi.org/10.1109/LRA.2017.2658940

[17] A. Suleiman, Z. Zhang, L. Carlone, S. Karaman, and V. Sze, "Navion: A 2-mw fully integrated real-time visual-inertial odometry accelerator for autonomous navigation of nano drones," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 4, 2019.

[18] Z. Li, Y. Chen, L. Gong, L. Liu, D. Sylvester, D. Blaauw, and H. Kim, "An 879gops 243mw 80fps vga fully visual cnn-slam processor for wide-range autonomous exploration," in *2019 IEEE International Solid- State Circuits Conference - (ISSCC)*, 2019.

[19] E. Flamand, D. Rossi, F. Conti, I. Loi, A. Pullini, F. Rotenberg, and L. Benini, "Gap-8: A risc-v soc for ai at the edge of the iot," in *2018 IEEE 29th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, 2018.

[20] P. Schiavone, F. Conti, D. Rossi, M. Gautschi, A. Pullini, E. Flamand, and L. Benini, "Slow and steady wins the race? a comparison of ultra-low-power risc-v cores for internet-of-things applications," 09 2017.

[21] M. W. Mueller, M. Hamer, and R. D'Andrea, "Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015.