Towards Anti-Collision Coordination for UAVs with Serverless Edge Computing

Tobias Pfandzelter*, David Bermbach*, Robert Vilter†, Ingo Friese‡,
Sergiy Melnyk§, Qiuheng Zhou§, Hans D. Schotten§¶
*Scalable Software Systems Group, Technische Universität Berlin & Einstein Center Digital Future, Berlin, Germany
†Aviation Engineering Group, Technical University of Applied Sciences Wildau, Wildau, Germany
‡Deutsche Telekom AG, Berlin, Germany
§Intelligent Networks, German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany
¶Institute for Wireless Communication and Navigation, University of Kaiserslautern (RPTU), Kaiserslautern, Germany

***Abstract*—Autonomously flying unmanned aerial vehicles in logistics, agriculture, and other domains will take up considerable airspace in the future. In areas with a high amount of air traffic, ground coordination to detect and mitigate encounters with collision risks is required. We propose a serverless approach to build and deploy such a system using edge computing resources. Serverless abstractions allow domain experts to focus on implementing the application logic while the underlying platform manages constrained resources and low-latency service access.**

***Index Terms*—serverless, edge computing, UAVs**

I. Introduction

The deployment of unmanned aerial vehicles (UAV) for logistics, agriculture, natural disaster control, and other use cases will transform how we use airspace, with a projected 400,000 commercial UAVs operating by 2050 in Europe alone [1]. Autonomous and remote operation of such UAVs in the limited airspace above urban areas or highly frequented airports and vertiports will require live coordination to lower collision risk to a minimum to prevent expensive and dangerous accidents.

In sixth-generation (6G) networks, ground-based coordination services can collect detailed UAV trajectory information, assess collision risks, and provide instructions for evasion maneuvers. These services can run on edge compute resources throughout the communication network as provided by the concept of split computing. Clients, such as UAVs, access these resources through reliable 6G wireless links [2], [3].

A key challenge in realizing such an edge anti-collision system is developing the right software abstractions that enable domain experts to implement the required logic in a flexible manner. Crucially, the low latency requirements of the application and the need to provide this service across geo-distributed edge nodes present unique software architecture challenges.

II. Serverless Edge Anti-Collision Coordination

We propose leveraging serverless abstractions to build a reliable, low-latency anti-collision coordination system for UAVs on edge computing infrastructure. Serverless computing allows us to maximize the efficiency of the constrained edge infrastructure while providing a high level of abstraction for the domain experts implementing the application itself.

Funded by the Bundesministerium für Bildung und Forschung (BMBF, German Federal Ministry of Education and Research) – 16KISK174K, 16KISK176, 16KISK177, 16KISK183.

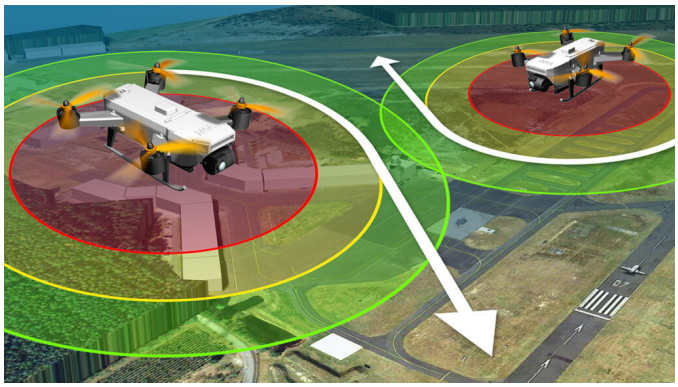

Fig. 1. If autonomously flying UAVs over a crowded airfield come too close, a coordinating entity must provide new trajectories to mitigate collision risks.

Architecture: We show the overall system architecture in Fig. 2. Within a critical airspace, e. g., above an airport or urban area, UAVs publish their current trajectories and flight plans using their 6G radio network access. These messages are collected at an edge node within the area, triggering a serverless workflow that computes collision risks, possibly leveraging AI models. If a collision risk is detected, the workflow calculates new trajectories that are sent to the UAVs. Crucially, given the high speed of these UAVs, it is paramount that this process can be executed within fractions of a second. Using edge computing ensures that this time can be spent on the application itself rather than on messaging delays [3], [4].

Serverless Compute: At the core of the system is a serverless compute platform based on the Function-as-a-Service (FaaS) paradigm, where applications are broken down into small, stateless *functions* executed in an event-based manner. FaaS is a natural fit for this application, as the anti-collision detection itself is event-based, i. e., it should be executed as soon as any new trajectory information is received. Using FaaS also allows the system to elastically scale with demand, e. g., dynamically (de)allocating constrained edge resources with the

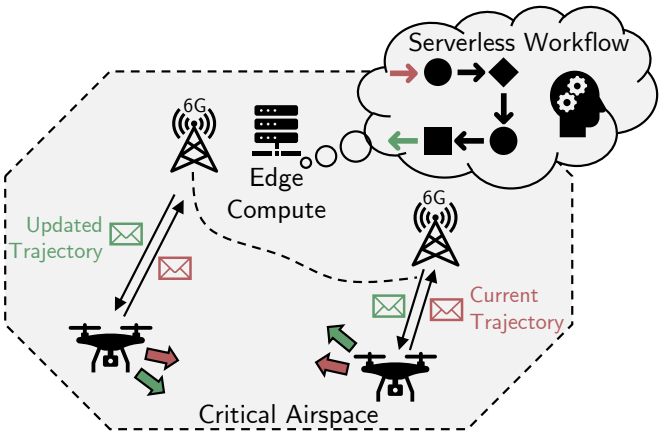

Fig. 2. The serverless anti-collision coordination system on an edge compute node collects published, geotagged trajectory messages from UAVs within the critical airspace and executes a serverless workflow in response. If a collision risk is detected, updated trajectories can be sent back to the UAVs.

amount of UAVs in the airspace an edge node is responsible for. Existing research has shown that FaaS platforms for the edge can be designed with minimal overhead for the application, although we will need to further assess this for applications as latency-critical as the anti-collision system [5].

Low-Latency, Reliable FaaS Workflows: FaaS functions can be composed to build a larger workflow. Our workflow comprises, e. g., traffic situation assessment, risk modeling, evaluation of UAV maneuvering capabilities, and trajectory re-calculation. Building this workflow on top of FaaS makes its execution flexible: Where possible, the FaaS platform can parallelize workflow steps, overlay computation and communication, and functions that are not critical for the workflow can be offloaded or executed with a delay, all without intervention from the application developer. FaaS workflows also support polyglot applications combining, e. g., machine learning in Python and simulation on C++, without the need to coordinate an interface between them. Finally, a FaaS platform can ensure reliable execution without requiring adaptations to the application itself. Nevertheless, existing FaaS platforms often impose considerable performance overheads for workflows, which must be eliminated when facing strict latency requirements [6].

Augmenting State: The main drawback of current FaaS platforms is their requirement for functions to be stateless, i. e., not to share data between invocations. In the cloud, this can be mitigated by using a database, yet this incurs overheads for database round-trip times, which is infeasible for a latency-critical edge workflow. In-memory caches can solve this to some extent but are mostly *best-effort* and thus also unfit. For the few stateful steps in the anti-collision coordination workflow, our edge serverless platform will need extensions to provide reads and writes with near-native speeds [7].

Geo-Distributed Messaging: Simply running our FaaS workflow on a single edge node is of course insufficient to provide anti-collision services in all critical areas. UAVs travelling across different regions must be able to transparently access a local service instance when in a critical airspace. Given the latency constraints we outlined, distributing messages through a central broker, e. g., in the cloud, is not viable. Instead, geo-distributed messaging with native support for geographical context is required. Running local edge brokers in each critical region that subscribe only to messages within that region (e. g., using geotags) ensures that the application receives all relevant messages with the shortest delay possible. Relying on geographical regions also allows for decoupling the radio network topology from the application logic, as a critical airspace region may have multiple radio access points. UAVs (equipped with hardware to calculate their geographical position) simply publish messages with their geotag into the network, without having to know a) if they are in a critical airspace or not, b) whether there is a broker in that airspace, or c) the network address of that broker. This allows dynamically changing airspace allocations without updating the UAVs, while providing a simple interface to develop against [8], [9].

III. Conclusion & Future Work

In the future, UAVs will require a considerable portion of the available airspace, which makes anti-collision coordination in highly frequented areas paramount. We propose building such an anti-collision system based on serverless edge computing, which abstracts from the complexity of handling geo-distributed networked computing systems and lets domain experts focus on implementing their application logic. In this paper, we have laid out the architecture for such a system based on serverless edge computing and have given an overview of the platform components required to implement it. Yet there remain open research questions in serverless edge computing that we plan to address in future work: The performance overhead of edge FaaS platforms must be reduced to support latency-critical edge applications. Especially (polyglot) function workflows at the edge must be executed with minimal overhead, which the current state-of-the-art does not support. Finally, we currently lack efficient support for stateful functions at the edge.

References

[1] SESAR, "European drones outlook study: Unlocking the value for europe," European Union, Tech. Rep. MG-01-17-281-EN-C, 2017.
[2] S. Melnyk *et al.*, "6g next — joint communication and compute mobile network: Use cases and architecture," in *Proc. KommA*, Nov. 2023, pp. 62–71.
[3] ——, "6g next — towards 6g split computing network applications: Use cases and architecture," in *Proc. MKT*, May 2023, pp. 126–131.
[4] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, 2016.
[5] T. Pfandzelter and D. Bermbach, "tinyfaas: A lightweight faas platform for edge environments," in *Proc. ICFC*, Apr. 2020, pp. 17–24.
[6] T. Schirmer *et al.*, "Fusionize: Improving serverless application performance through feedback-driven function fusion," in *Proc. IC2E*, Sep. 2022, pp. 85–95.
[7] T. Pfandzelter and D. Bermbach, "Enoki: Stateful distributed faas from edge to cloud," in *Proc. MiddleWEdge*, Dec. 2023, pp. 19–24.
[8] J. Hasenburg and D. Bermbach, "Geobroker: Leveraging geo-context for iot data distribution," *Elsevier Computer Communications*, vol. 151, pp. 473–484, 2020.
[9] ——, "Disgb: Using geo-context information for efficient routing in geo-distributed pub/sub systems," in *Proc. UCC*, Dec. 2020, pp. 67–78.