IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), Galway, Ireland, 2025.

Remote Autonomy for Multiple Small Lowcost UAVs in GNSS-denied Search and Rescue Operations

Daniel Schleich[a], Jan Quenzel[a,b], and Sven Behnke[a,b]

***Abstract*— In recent years, consumer-grade UAVs have been widely adopted by first responders. In general, they are operated manually, which requires trained pilots, especially in unknown GNSS-denied environments and in the vicinity of structures. Autonomous flight can facilitate the application of UAVs and reduce operator strain. However, autonomous systems usually require special programming interfaces, custom sensor setups, and strong onboard computers, which limits a broader deployment.**

We present a system for autonomous flight using lightweight consumer-grade DJI drones. They are controlled by an Android app for state estimation and obstacle avoidance directly running on the UAV's remote control. Our ground control station enables a single operator to configure and supervise multiple heterogeneous UAVs at once. Furthermore, it combines the observations of all UAVs into a joint 3D environment model for improved situational awareness.

I. Introduction

In disaster response scenarios, the ability to quickly obtain an overview of the situation is essential for first responders. Moreover, the situational picture must be continuously updated throughout the rescue operation. Unmanned aerial vehicles (UAVs) are increasingly deployed for this task as UAVs cover large inaccessible areas quickly, independent of the terrain. In most disaster-response operations, trained human pilots directly control the UAVs. Autonomous flights are commonly restricted to high flight altitudes where the UAV follows preplanned GNSS-waypoint missions without risking collisions [1]. Autonomous UAVs that don't need GNSS-based localization [2]–[4] rely on custom sensor setups and onboard compute to fly in the vicinity of obstacles. Thus, larger UAVs with special communication interfaces and sufficient payload are required. This restricts large-scale deployment and increases risks due to high impact weight and dangerous propellers.

In this work, we propose remote autonomy for GNSS-denied flight of affordable lightweight consumer-grade UAVs, like the DJI Mini 3 Pro. Our solution does not require custom sensors or any hardware modifications, which makes it cost-efficient and allows for broad deployment in real-world operations and for using multiple UAV simultaneously to quickly obtain situation-awareness.

In particular, our remote autonomy system includes:

- an Android app based on the DJI Mobile SDK (MSDK) to control a UAV in GNSS-denied environments, including obstacle avoidance,

[a]Autonomous Intelligent Systems, University of Bonn, Germany; [b]Center for Robotics and Lamarr Institute for Machine Learning and Artificial Intelligence, University of Bonn, Germany

Fig. 1. Multiple consumer-grade UAVs supervised by a single operator facilitate Search & Rescue in an industrial hall. Left: External view of the UAVs; Right: Created 3D RGB map with UAV trajectories (colored arrows).

- an environment mapping module that combines the observations of all UAVs into a global model, and
- an intuitive user interface and fleet management system for configuration and supervision of multiple UAVs.

II. Related Work

UAVs are increasingly employed by public safety authorities [5], e.g., for disaster response [6], [7]. This often includes custom hardware designs for the challenges of specific rescue operation, e.g., lifebuoy delivery under harsh offshore conditions [8] or avalanche victim search [9]. For the task of 3D environment mapping, Lauterbach et al. [10] equip a large professional-grade UAV with a custom-build sensor unit including LiDAR. These systems are capable of autonomous flight but rely on GNSS for navigation. Furthermore, due to the necessary payload, they are usually based on large UAV platforms.

For indoor missions, small UAVs are necessary, which cannot rely on GNSS localization. One example is the low-cost platform of Tavasoli et al. [11] for damage assessment of structural columns. The UAV is manually controlled but the movements for data collection can be automated. Similarly, the solution of Pliakos et al. [12] is capable of semi-autonomous GNSS-denied indoor missions, including waypoint navigation, obstacle avoidance and 3D reconstruction of the UAV surroundings.

In contrast, Beul et al. [13] developed a 3D LiDAR-based UAV for SLAM and fully autonomous navigation in warehouses. Quenzel et al. [14] navigate autonomously inside industrial chimneys based on 3D LiDAR and a camera. Reyes-Munoz et al. [15] demonstrate a system with RGB-D and stereo cameras for fully autonomous flight in GNSS-denied environments. All of these systems extend commercial air frames with sensors and onboard computers, which increases complexity, weight, and costs; and which decreases their flight time.

For a broader applicability, cost-efficient consumer-grade UAVs are favorable, which do not need hardware modifi-

cations. Following this idea, Khosravi et al. [16] use the DJI Go app and SDK to execute exploration trajectories for autonomous person search on a DJI drone. However, they consider outdoor environments where the UAV can operate at obstacle-free altitudes using GNSS.

Most similar to our approach is the work of Surmann et al. [17], who use small consumer-grade UAVs to explore indoor environments. Their system offers local 3D modeling and autonomous corridor following, while otherwise still relying on manual control. In contrast, our remote autonomy solution offers fully autonomous flight and globally consistent environment mapping which can incorporate the perception of multiple UAVs.

III. Remote Autonomy System Setup

Our remote autonomy system works with a variety of unmodified DJI drones, e.g., Mini 3 Pro, Mini 4 Pro and Mavic 3T, using the DJI MSDK[1] (v5.8+). Figure 2 gives an overview of our approach.

Each UAV connects to its remote controller (RC) via a proprietary wireless link. The RC connects to an integrated Android device, e.g. in the DJI Pro RC, or an external one, e.g. a smart phone, where the MSDK runs. To reduce latency as far as possible, all safety-relevant software modules, like visual-inertial odometry (VIO) and obstacle avoidance, run within a single Android app on the RCU. The app receives new tasks from our ground control station (GCS) and sends images and odometry to the GCS to enable there the fusion of environment measurements from multiple UAVs to a 3D map. GCS and the RCUs communicate over WiFi using a custom server with Protobuf[2] for efficient serialization of transmitted messages.

The UAV's h265-encoded video feed is decoded on the RCU for processing within the VIO module. For efficiency, DJI submits the crucial I-frames exceptionally infrequent. Missing one I-frame renders the stream useless for one minute or more. To address this issue, we re-encode the image stream using the RCU's integrated hardware acceleration to ensure correct transmission and conserve bandwidth when forwarding the h265 packages through our Protobuf server.

Our fleet management also has an interface to connect UAVs from other manufacturers over MAVROS[3]. For these, we assume that the main functionality of our Android app (e.g., odometry and obstacle avoidance) is provided by the manufacturer's flight controller.

A. Ground Control Station

The GCS consists of a powerful PC for executing all resource-intensive and non-time-critical tasks. It computes a joint environment model (Sec. III-A.1) from the local perception of each UAV. Additionally, the GCS visualizes all UAV states in real time and provides an intuitive control interface, allowing a single operator to configure and supervise multiple UAVs simultaneously (Sec. III-A.2). All of these modules are implemented using ROS.

[1] https://github.com/dji-sdk/Mobile-SDK-Android-V5
[2] https://github.com/protocolbuffers/protobuf
[3] https://github.com/mavlink/mavros

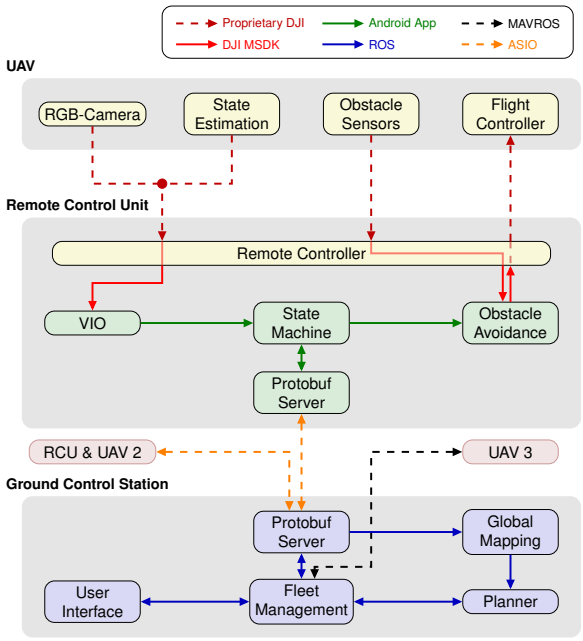

Fig. 2. Remote autonomy system architecture. Software modules are divided into two layers: For each UAV, the safety-relevant modules are executed on the UAV's remote control unit (RCU). It consists of an Android device and a remote controller (RC), connected via the DJI MSDK. A WiFi Protobuf server connects each RCU to the ground control station (GCS), where a single operator configures and supervises multiple UAVs and where globally consistent 3D environment models are created and visualized. UAVs from other manufactures can be connected over a MAVROS interface.

1) Globally Consistent 3D Environment Mapping: To generate a global map, we select keyframes from multiple UAVs based on a frustum overlap. The selection uses our VIO pose estimate as a prior and chooses adjacent keyframe candidates for matching. We extract XFeat [18] from all keyframes and perform matching with LighterGlue between candidates. Afterwards, we employ sparse incremental reconstruction with the optimization backend of GLOMAP [19].

Assuming that multiple UAVs start in the same GNSS-denied area, our mapping initially reconstruct observations from a single UAV. Another UAV is added once it has enough keyframe matches (e.g., 5) with the current reconstruction. Starting from there, all observations of the UAV will be included.

The optimization backend provides a reconstruction in an arbitrary frame with scale ambiguity. Our application requires a common metric reference frame, though. To address this issue, a RANSAC-based pose alignment facilitates the scale transformation if GNSS data is available. In GNSS-denied situations, we estimate the scale from the first 10 poses, and align the origin with the first UAV's starting position and orientation.

The resulting map is quite sparse and often difficult to understand quickly, even by trained professionals. Hence, we establish dense pairwise correspondences using MAST3R [20] between our previously matched adjacent keyframes. Due to time and compute limitations, MAST3R is initially only applied to the new keyframe with two previous

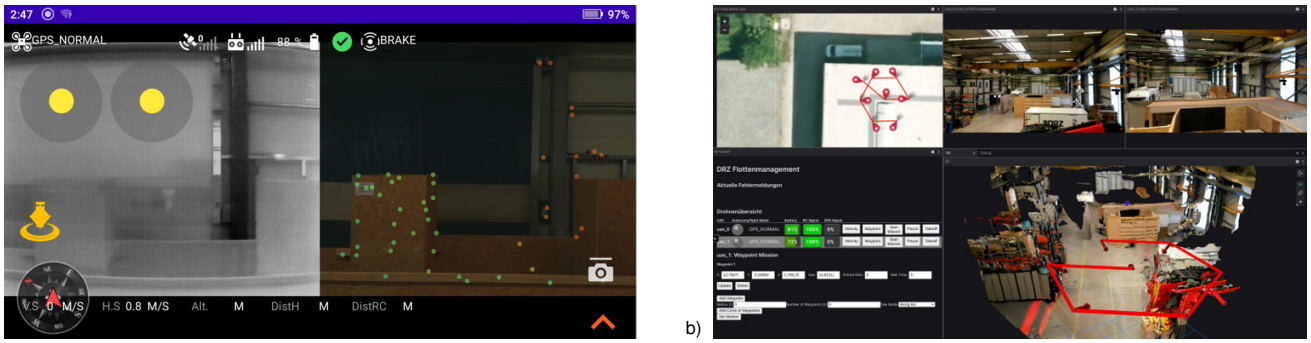

Fig. 3. User interfaces on our remote controller (RCU, a) and at the ground control station (GCS, b). RCU: Various status indicators are overlayed on the video feed for the safety pilot including UAV heading and virtual stick commands. The example shows the thermal image and tracked VIO features on the color image of a DJI M3T in side-by-side mode. GCS: The fleet management shows the current UAVs' states including control options (b, bottom left). The current waypoint mission is shown in the joint 3D environment model (b, bottom right) and overlayed on a satellite image (b, top left). Live-streams (b, top right) allow direct supervision and targeted inspection.

ones. More previously unprocessed pairs are matched by MAST3R in a lazy-evaluation scheme, if no new keyframes are established within some seconds, e.g., when all UAVs are hovering. We merge shared observations from pairwise correspondences between multiple keyframes prior to triangulation. The triangulation uses the dense matches from MAST3R with our reconstructed metric poses. Figure 1 shows estimated camera poses and colored 3D points of autonomous flights of two DJI Mini 3 Pro within the industrial hall of the DRZ Living Lab.

2) User Interface & Fleet Management: For intuitive supervision and control of our UAVs, we implemented a custom extension to Foxglove Studio[4] (see Fig. 3b). It continuously monitors the state of all registered UAVs and informs the operator about possible errors. During missions, new UAVs are added anytime just by connecting the Android app on the corresponding RCU. The app automatically initiates the connection with the GCS and registers the UAV into the fleet management module. Here, existing UAVs are recognized on re-establishing the connection after temporary interruption.

In addition to supervision, our UI supports switching between individual UAVs in two control modes. In *Velocity Control*, the operator steers the UAV using a gamepad similar to the original RCU. However, directly commanding flight velocities is not feasible due to the latency between control input and video feedback. Instead, we implemented a *carrot mode* where the operator moves a virtual target pose in the local 3D environment. This target is visualized without latency and facilitates control significantly.

In *Waypoint Control*, the operator inputs a sequence of target poses as a list of coordinates via keyboard, by moving a marker within the 3D environment model using the gamepad, or by marking positions on satellite images or maps. Additional parameters like individual wait time or gimbal orientation provide flexibility for every mission. Flight patterns like circles are inserted automatically, without manually defining each waypoint. To ensure collision-freeness, a dynamic trajectory planner [21] interpolates between waypoints using the global 3D model. The fleet management then transforms the resulting flight plan into the local UAV frame and keeps sending the next planned waypoint to the RCU for execution, while monitoring the progress.

[4]https://foxglove.dev/studio

B. Remote Controller with Android App

The RCU runs our Android app (Fig. 3a) as a configuration and supervision interface for the safety pilot. It visualizes the current UAV state, camera live-stream, control mode, and the executed VirtualStick commands. To enable the autonomy functions, an explicit clearance from the safety pilot is necessary. Moreover, clearance may be revoked at any time, e.g., by moving the control sticks of the RC.

The Android app steers the UAV towards the next target pose using DJI's VirtualStick commands. These emulate RC stick movements of an operator and represent 3D linear velocities and yaw. We obtain target velocities from the difference between current and target pose using either GNSS or our state estimation (Sec. III-B.1) as position feedback. These velocities are clipped at a configurable maximum flight speed, and adjusted by the reactive obstacle avoidance (Sec. III-B.2). Once the UAV is sufficiently close to the target, we switch to DJI's Position Hold mode. If the UAV drifts too far from the target pose, we reactivate our control.

In addition to the UAV pose, our app controls the gimbal either according to a manually defined orientation from the GCS or automatically during people tracking. For the latter, we run an EfficientDet Lite0 [22] at $\approx$ 20 Hz directly on the RCU using the TensorFlow Lite Task library[5]. Given the person detection, our app adjusts the gimbal to keep the bounding box centered.

1) State Estimation: Most autonomy functions rely on knowledge of our position and orientation relative to the environment, which is especially important in GNSS-denied environments. Outdoors, the DJI MSDK provides the UAV's GNSS position with up to $\pm 0.1\,\mathrm{m}$ resolution. In GNSS-denied environments no position information is given, though. The MSDK's reported velocity and orientation are

[5]https://ai.google.dev/edge/litert/libraries/task_library/overview

quantized to an accuracy of $\pm 0.1\,\mathrm{m/s}$, resp. $\pm 0.1°$. Moreover, current MSDKs (v5.8+) supply the current (quantized) state not on a regular basis, but only once something changes.

As simply integrating the velocity would be too inaccurate and without IMU measurements, we adapted OpenVINS [23] for the provided inputs: i) We merge and retain the current measurements to submit them to the VIO at a required continuous rate. ii) We replaced the IMU input during state propagation with a constant velocity (CV) motion model. iii) The truncated velocity and orientation are integrated in the update step considering an appropriate uncertainty. If available, the UAV's altitude is further included in the update. As a result, our modified VIO runs on the RCU on the monocular FPV camera stream with up to 60 Hz.

2) Obstacle Avoidance: In order to react quickly to dynamic or unseen obstacles, we reactively adjust the velocity commands similar to the Predictive Angular Potential Fields method [24]. The MSDK provides 360 horizontal obstacle measurements in a scan line where only a UAV-depending subset is valid. A Mini 3 Pro supplies valid distances in forward and backward direction, while a Mini 4 Pro covers the full 360°. The received distances are aggregated over a short window into a spherical range image. Before computing the global potential field $\mathcal{P}$ using a L_1 distance transform [25], we define per pixel the force as $\mathrm{atan2}(d_\mathrm{s}, r)$, with safety distance d_s and obstacle distance r. To determine the direction, i.e., yaw, of the adjusted velocity, we first project the target command into the potential field image. We then find the closest zero-force pixel within the FoV using gradient descent and reproject it into 3D. The magnitude of the velocity command is determined by the obstacle distance in the corresponding direction.

In general, we try to maximize the distance to obstacles. However, close distances cannot be avoided in certain situations, i.e., when passing through narrow corridors. We address this using two different safety distances d_s and d_min along with their corresponding potential fields $\mathcal{P}_\mathrm{s}$ and $\mathcal{P}_\mathrm{min}$. If no zero-force pixel is found for $\mathcal{P}_\mathrm{s}$, we choose a zero-force pixel from $\mathcal{P}_\mathrm{min}$ minimizing $\mathcal{P}_\mathrm{s}$. If such a pixel is not found either, we stop as there is no trajectory with sufficient obstacle clearance.

3) Door Traversal Mode: The measurement range of DJI obstacle sensors has a lower limit of 0.5 m. During experiments, we found that closer obstacles are still frequently detected but the distance measurements become unreliable, overestimating the actual values. Thus, we enforce an obstacle clearance of at least 0.5 m within our obstacle avoidance. However, when passing through doors, this clearance cannot be maintained. Thus, we integrated a specific control mode for autonomous door traversal.

After approaching the door using the control pipeline described above, the operator manually triggers the door traversal mode. In a first step, the door opening is detected within the horizontal obstacle scan by searching for jumps in adjacent distance measurements. Since we assume that the UAV is already roughly oriented towards the door, we restrict this search to $\pm 60°$ around the current heading. The detection

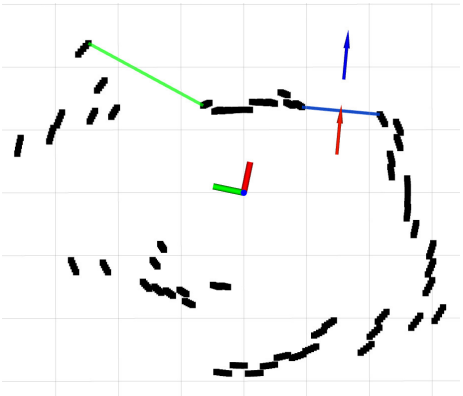

Fig. 4. Door detection within horizontal obstacle scan. The current UAV pose is marked by the coordinate axes. The green line indicates a door candidate that is filtered out since its normal deviates too much from the scan's radial direction. The blue line represents the selected door candidate. The red and blue arrows correspond to the planned pre- and post-traversal poses, respectively.

reliability is maximized when the door's surface normal aligns with the radial direction of the obstacle scan. Thus, we remove all detections exhibiting a large deviation from this alignment. Furthermore, this helps to remove false positives caused by jumps in the distance scan due to occlusions. Finally, all candidates are checked for sufficient width and clearance behind the opening. If multiple suitable detections remain, we choose the one whose normal aligns best with the current UAV heading. Figure 4 shows an example of the door detection.

In a next step, a pre-traversal pose is computed: We project the end points of the door opening into 3D and horizontally offset the midpoint along the outward normal. The target yaw is chosen to align with the direction of the inward normal. To compensate for drift or inaccuracies in the obstacle measurements, we only steer the UAV a small distance towards the pre-traversal pose and update the target on receiving the next obstacle scan.

When the UAV reaches a stable pre-traversal pose, we fix the estimated doorway position and steer the UAV along the doorway normal toward the post-traversal pose placed behind the door opening. Lateral obstacle avoidance is disabled while traversing the door.

IV. Evaluation

1) Control Latency: In a first experiment, we evaluate the latency of our pipeline. We measure the time from sending a motion command at the GCS until the movement is visualized to the operator. Additionally, we record the delay until the UAV starts moving using a motion capture system. On average, it took 496 ms until a motion started and another 344 ms for the updated VIO estimate to be visualized at the GCS. The round-trip time of the WiFi connection between GCS and Android RCU was only 4.5 ms. The total latency of 839 ms emphasizes the necessity for our *carrot mode* as manual control with raw velocity commands becomes infeasible at this latency.

2) Flight Towards a Waypoint: DJI lets us control the UAV via velocity commands with a resolution of 0.1 m/s at

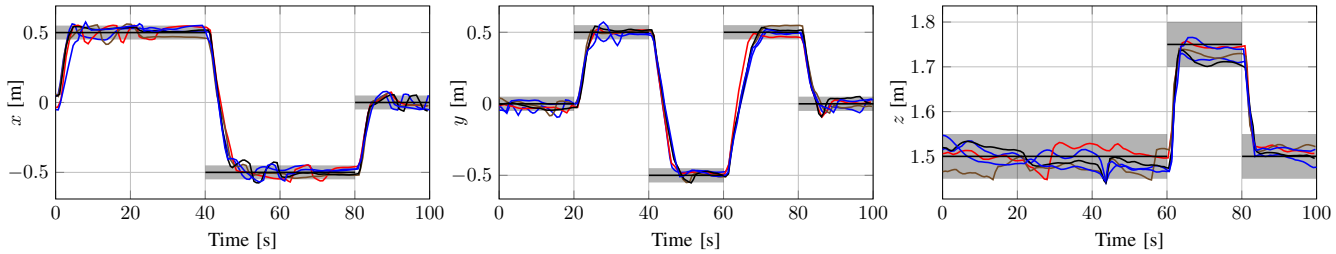

Fig. 5. Evaluating position control accuracy. Horizontal black lines depict five target positions with the gray-shaded area indicating to the target threshold. The colored lines show the trajectories of five different runs as estimated by the VIO.

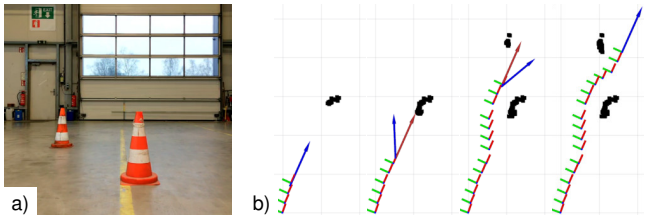

Fig. 6. Obstacle avoidance experiment. a) The UAV is commanded to fly straight towards two obstacles. b) The flight trajectory is shown as sequence of coordinate axes. Detected obstacles are black. The arrows depict the target velocity (red) and the executed velocity (blue) after adjustment by the obstacle avoidance module.

a frequency of 10 Hz. To analyze how accurately we reach a given target, we autonomously fly towards five different poses using a target threshold of 5 cm per axis. The resulting trajectories (estimated from VIO) for five repeated flights are shown in Fig. 5. We successfully reached all targets in all tries. Occasionally, the UAV drifted out of the target zone but our control directly steered it back. We tracked the distance between the VIO and target poses, from first entering the zone until the next target was sent. On average, we achieve an accuracy of 4.7 cm with a std. dev. of 1.2 cm.

3) Obstacle Avoidance: Next, we evaluate our obstacle avoidance. The UAV was commanded to fly straight forward towards traffic cones (Fig. 6 a). As shown in Fig. 6 b, our method successfully adjusted the velocity commands and steered the UAV around the obstacles.

4) Door Traversal: Additionally, we tested the *Door Traversal Mode* by commanding the UAV to fly through a door within our lab (Fig. 7). In the onboard camera, the door opening was only partially visible from the start position (Fig. 7 a) and not visible anymore after reaching the pre-traversal pose (Fig. 7 c). Nevertheless, the door was reliably detected within the obstacle scan during the whole flight. The estimated door width ranged from 0.78 m to 1.38 m, with an average of 1.10 m and a std. dev. of 0.12 m, thus slightly underestimating the actual width of 1.25 m most of the time. The detected door orientation ranged from 0.2° to 30°, with average 6.7° and std. dev. 6.4°. The inaccuracies in the estimation of the door pose were successfully handled by continuously updating the pre-traversal pose to the most recent detections. Figure 7 d shows that the UAV passed through the center of the door opening with sufficient safety clearance to the door frame.

5) 3D Mapping: Finally, we evaluate the global environment mapping. Figure 8 shows an example of a reconstructed 3D point cloud from three different UAVs (DJI Mini 3 Pro,

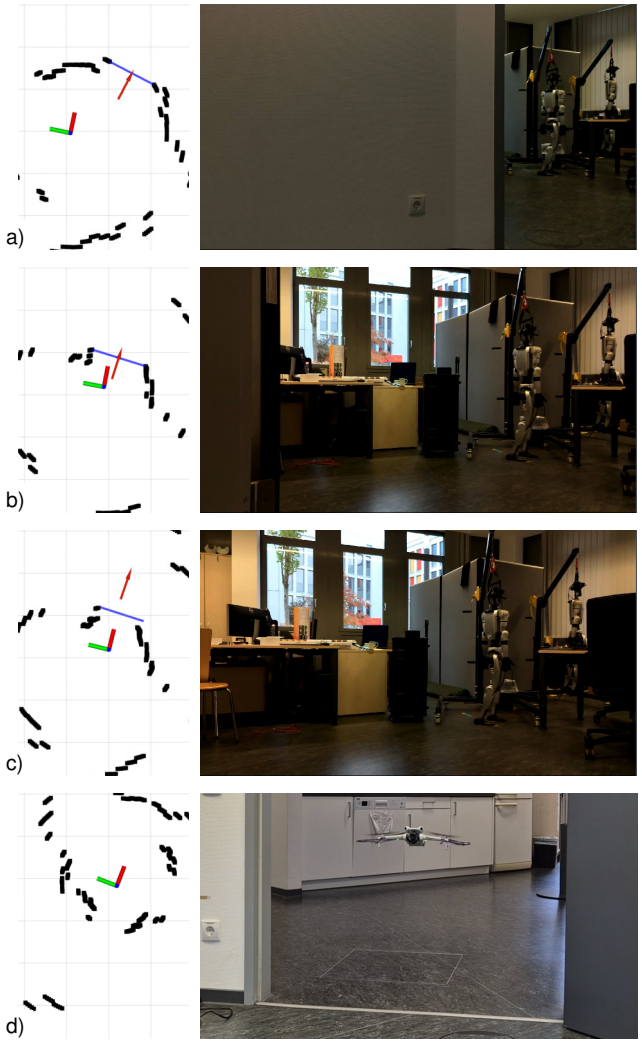

Fig. 7. Door traversal experiment. Obstacles scans including the current UAV pose (axes), door detections (blue line) and current target pose (red arrow) are depicted on the left. a)-c) show corresponding onboard footage, while d) shows an external view of the UAV passing the door.

Mini 4 Pro, and Mavic 3T) flying simultaneously in the DRZ Living Lab. Additionally, we tested our system in an outdoor scenario. A densely aggregated LiDAR point cloud [26] provides a reference for the reconstruction when either GNSS or odometry poses are used. For the comparison, we compute the point-to-point distance to the LiDAR map. To cope with unrepresented areas within the LiDAR map, we threshold distances above 0.5 m and 1 m. The mean distance is 0.081 m at threshold 0.5 m for both sets of poses. However, we obtain 0.135 m for the odometry and 0.150 m

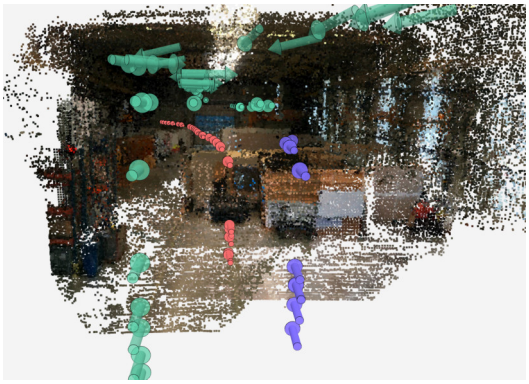

Fig. 8. Reconstructed colored 3D map from images of three UAVs. Colored arrows represent the UAV flight trajectories.

using GNSS at threshold 1 m. Moreover, the odometry poses are more consistent as emphasized by the 33% higher count of triangulated points whereas the percentage of points above the threshold is similar.

V. Conclusion & Future Work

We presented a remote autonomy pipeline for flight in GNSS-denied environments using small consumer-grade UAVs. We developed an Android app for the remote controller that steers the UAV towards target poses while successfully avoiding obstacles. The ground control station fuses the perception of multiple UAVs into a global 3D environment model, while offering an intuitive interface for configuration and supervision of multiple UAVs by a single operator. As a next step, we plan to conduct a user study in a simulated rescue operation to evaluate the usability of our approach under realistic conditions by real first responders.

Acknowledgement

We would like to thank Subham Agrawal, Nils Dengler and Maren Bennewitz from the Humanoid Robots Lab at the University of Bonn for their help with the Motion Capture System.

This work has been supported by the German Federal Ministry of Research, Technology and Space (BMFTR) in the project "Kompetenzzentrum: Etablierung des Deutschen Rettungsrobotik-Zentrums (E-DRZ)", grant 13N16477.

References

[1] H. Surmann, D. Slomma, R. Grafe, and S. Grobelny, "Deployment of aerial robots during the flood disaster in Erftstadt / Blessem in july 2021," in *8th International Conference on Automation, Robotics and Applications (ICARA)*, 2022.

[2] D. Schleich, M. Beul, J. Quenzel, and S. Behnke, "Autonomous flight in unknown GNSS-denied environments for disaster examination," in *Int. Conference on Unmanned Aircraft Systems (ICUAS)*, 2021.

[3] M. Petrlík, P. Petráček, V. Krátký, T. Musil, Y. Stasinchuk, M. Vrba, T. Báča, D. Heřt, M. Pecka, T. Svoboda, and M. Saska, "UAVs beneath the surface: Cooperative autonomy for subterranean search and rescue in DARPA SubT," *IEEE Transactions on Field Robotics (TRO)*, vol. 2, pp. 643–689, 2024.

[4] Y. Luo, Z. Zhuang, N. Pan, C. Feng, S. Shen, F. Gao, H. Cheng, and B. Zhou, "Star-Searcher: A complete and efficient aerial system for autonomous target search in complex unknown environments," *IEEE Robotics and Automation Letters (RA-L)*, vol. 9, no. 5, 2024.

[5] M. Stampa, A. Sutorma, U. Jahn, J. Thiem, C. Wolff, and C. Röhrig, "Maturity levels of public safety applications using unmanned aerial systems: a review," *Journal of Intelligent & Robotic Systems (JINT)*, vol. 103, pp. 1–15, 2021.

[6] A. Khan, S. Gupta, and S. K. Gupta, "Emerging UAV technology for disaster detection, mitigation, response, and preparedness," *Journal of Field Robotics (JFR)*, vol. 39, no. 6, pp. 905–955, 2022.

[7] I. Kruijff-Korbayová, R. Grafe, N. Heidemann, A. Berrang, C. Hussung, C. Willms, P. Fettke, M. Beul, J. Quenzel, D. Schleich, S. Behnke, *et al.*, "German Rescue Robotics Center (DRZ): A holistic approach for robotic systems assisting in emergency response," in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2021, pp. 138–145.

[8] S. E. Eid and S. S. Dol, "Design and development of lightweight-high endurance unmanned aerial vehicle for offshore search and rescue operation," in *Advances in Science and Engineering Technology International Conferences (ASET)*, 2019.

[9] M. Silvagni, A. Tonoli, E. Zenerino, and M. Chiaberge, "Multipurpose UAV for search and rescue operations in mountain avalanche events," *Geomatics, Natural Hazards and Risk*, vol. 8, no. 1, pp. 18–33, 2017.

[10] H. A. Lauterbach, C. B. Koch, R. Hess, D. Eck, K. Schilling, and A. Nüchter, "The Eins3D project—Instantaneous UAV-based 3D mapping for search and rescue applications," in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2019.

[11] S. Tavasoli, X. Pan, T. Yang, S. Gazi, and M. Azimi, "Autonomous damage assessment of structural columns using low-cost micro aerial vehicles and multi-view computer vision," in *Canadian Conference on Earthquake Engineering (CCEE)*, 2023.

[12] C Pliakos, S Vlachos, C Bliamis, and K Yakinthos, "Preliminary design of a multirotor UAV for indoor search and rescue applications," in *Journal of Physics: Conference Series*, 2024.

[13] M. Beul, D. Droeschel, M. Nieuwenhuisen, J. Quenzel, S. Houben, and S. Behnke, "Fast autonomous flight in warehouses for inventory applications," *IEEE Robotics and Automaton Letters (RA-L)*, vol. 3, no. 4, pp. 3121–3128, 2018.

[14] J. Quenzel, M. Nieuwenhuisen, D. Droeschel, M. Beul, S. Houben, and S. Behnke, "Autonomous MAV-based indoor chimney inspection with 3D laser localization and textured surface reconstruction," *Journal of Intelligent and Robotic Systems (JINT)*, vol. 93, no. 1-2, pp. 317–335, 2019.

[15] J. A. Reyes-Munoz and A. Flores-Abad, "A MAV platform for indoors and outdoors autonomous navigation in GPS-denied environments," in *IEEE International Conference on Automation Science and Engineering (CASE)*, 2021.

[16] M. Khosravi, R. Arora, S. Enayati, and H. Pishro-Nik, "A search and detection autonomous drone system: From design to implementation," *IEEE Tr. on Automation Science and Engineering*, 2024.

[17] H. Surmann, T. Kaiser, A. Leinweber, G. Senkowski, D. Slomma, and M. Thurow, "Small commercial UAVs for indoor search and rescue missions," in *International Conference on Automation, Robotics and Applications (ICARA)*, 2021.

[18] G. Potje, F. Cadar, A. Araujo, R. Martins, and E. R. Nascimento, "XFeat: Accelerated features for lightweight image matching," in *IEEE/CVF Conf. on Comp. Vision and Pattern Rec. (CVPR)*, 2024.

[19] L. Pan, D. Baráth, M. Pollefeys, and J. L. Schönberger, "Global structure-from-motion revisited," in *European Conference on Computer Vision (ECCV)*, 2024.

[20] V. Leroy, Y. Cabon, and J. Revaud, "Grounding image matching in 3D with MASt3R," in *Eur. Conf. on Computer Vision (ECCV)*, 2024.

[21] D. Schleich and S. Behnke, "Search-based planning of dynamic MAV trajectories using local multiresolution state lattices," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[22] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[23] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A research platform for visual-inertial estimation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[24] D. Schleich and S. Behnke, "Predictive angular potential field-based obstacle avoidance for dynamic UAV flights," in *IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS)*, 2022.

[25] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance transforms of sampled functions," *Theory of Computing*, vol. 8, no. 1, pp. 415–428, 2012.

[26] J. Quenzel and S. Behnke, "Real-time multi-adaptive-resolution-surfel 6D LiDAR odometry using continuous-time trajectory optimization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 5499–5506.