# Radar-Inertial Odometry for Closed-Loop Control of Resource-Constrained Aerial Platforms

Jan Michalczyk, Martin Scheiber, Roland Jung and Stephan Weiss[1]

***Abstract*— In this paper, we present a Radar-Inertial Odometry (RIO) framework capable of running on a portable resource-constrained embedded computer in real-time as a state estimator for closing the feedback control loop on an Unmanned Aerial Vehicle (UAV) platform. The presented framework efficiently implements a RIO approach relying on the multi-state tightly-coupled Extended Kalman Filter (EKF) fusing instantaneous velocities of and distances to 3D points delivered by a lightweight, low-cost, off-the-shelf Frequency Modulated Continuous Wave (FMCW) radar with Inertial Measurement Unit (IMU) readings. The usage, accuracy and consistency of the implemented framework are improved compared to state-of-the-art by the online calibration of the sensors' extrinsic parameters. Our method is particularly relevant for (but not limited to) UAVs, enabling them to navigate autonomously in Global Navigation Satellite System (GNSS)-denied environments using very affordable and accessible hardware. In addition, thanks to the properties of the radar sensor, we enable autonomous navigation in challenging conditions for robot perception due to external factors such as fog, darkness or strong illumination which might be encountered in disaster zones. We show in real-world closed-loop flight experiments the effectiveness and efficiency of our estimator. The beneficial impact of the online calibration on estimation accuracy and consistency is also shown. Moreover, we compare the presented approach to a state-of-the-art vision-based algorithm (Visual-Inertial Odometry (VIO)) in visually degraded conditions.**

## I. Introduction And Related Work

One of the expected advantages of UAVs is their ability to be deployed in complex and hostile environments such as disaster zones, areas with adverse weather conditions, or hazardous industrial zones to perform autonomous missions such as reconnaissance, inspection, search and rescue, and others. In order to operate autonomously in such environments, the localization system of the platform must make use of sensors robust towards phenomena such as fog, smoke or extreme illumination. Since recently, thanks to their physical properties, their miniaturization, and general advances in their technology, radar sensors are being increasingly investigated for their use in navigation of various types of autonomous robots requiring resilience towards unfavourable environment factors. Another key requirement of an onboard localization system on an autonomous mobile robot is the need to execute within the tight limits imposed by the real-time control loop and often on a resource-constrained computing platform.

A number of radar-based localization methods have been presented in the scientific community in the past few years.

[1] All authors are with the Control of Networked Systems Group, University of Klagenfurt, Austria {firstname.lastname}@ieee.org

This research received funding from the Austrian Ministry of Climate Action and Energy (BMK) under the grant agreement 880057 (CARNIVAL).

**Pre-print version, accepted October/2023, DOI follows ASAP ©IEEE.**


Fig. 1. Experimental platform used in this work with the FMCW radar and camera sensors placed in the take-off position in dense artificial fog used to simulate a disaster site conditions. Note that the camera has been used only for comparison with VIO in the experiments with the artificial fog (see the Subsection V-D).

Significant contributions made in the automotive domain cited in [1], [2], [3], [4], [5], [6] make use of bulky and expensive scanning radars inadequate for use on UAVs mostly due to their size and power consumption. In [7] the authors present an optimization-based continuous-time RIO method which is particularly well suited for multi-radar setups as the continuous representation of the vehicle trajectory allows sampling it at any given time, thus permitting efficient asynchronous fusion of multiple radars measurements with the IMU. The method in [7] is demonstrated on an automotive platform with high-performance cascaded radars, an order of magnitude costlier, and still of greater size than the single-chip radar used in our work. Indeed, we focus more on the application of low-cost and small single-chip FMCW radars on UAVs, where the payload is of crucial relevance. In this domain, in [8], [9], [10], [11] the authors present a real-time RIO method in which the platform velocity estimated from a single radar scan is fused in a loosely-coupled manner with IMU readings in an EKF framework. In [9] the authors extend the method presented in [8] with online calibration of radar extrinsic parameters which boosts the accuracy and simplifies the usage of the approach. Nonetheless, the barometer sensor is additionally used, which contrasts with our work where we aim at using radar and IMU only. In [10] the same EKF framework is further augmented with the yaw aiding using Manhattan assumptions on the environment thanks to which the method attains high accuracy with

minimal run-time footprint. In [11] still the same method is equipped with GNSS sensor providing a versatile and precise outdoor localization solution, nonetheless at the expense of introducing a dependency on GNSS reception. As opposed to [10], [11] our approach makes no assumptions on the environment and employs only radar and IMU. In [12] and [13] an optimization-based approach is shown where a sliding window of past radar and IMU measurements is used to estimate the UAV velocity. The approach in [12] attains comparable performance with the vision-based system and even outperforms it in visually degraded settings. Additionally, in [13] the authors add a comparison of their method with a lidar-based approach and propose a method to reduce the noise in the radar pointclouds based on Deep Learning. The authors in both [12] and [13] concentrate on estimating the ego-velocity instead of the full 6D pose. Also, both approaches have not yet been demonstrated in closed-loop flights.

In the present paper, we demonstrate the capabilities of the RIO method presented in [14] and [15] in closed-loop flights and show that it enables a UAV to reliably navigate autonomously in an unknown environment. The real-time state feedback control of a resource-constrained UAV is enabled by the efficient implementation of the aforementioned RIO approach. The accuracy of the implementation, as shown still boosted with respect to [15] thanks to the online extrinsic sensor parameters calibration, permits precise tracking of trajectories even when vision-based methods fail. Online calibration also obviates the manual efforts related to measuring spatial configuration of sensors and improves the estimator's consistency. Thanks to the above features, the presented framework allows for a reliable and seamless use within an autonomous UAV navigation stack as a localization component. We leverage this for integrating our framework into the *CNS Flight Stack* [16]. Integration and subsequent deployment into an autonomous mission require only minimal effort. To the best of our knowledge this is the first time that a RIO framework using *only* radar and IMU and making use of no assumptions on the environment has been used for closed-loop control of a UAV including sensor extrinsic calibration capabilities. Our main contributions are:

- Implementation of a RIO framework capable of executing in real-time on a portable resource-constrained credit-card sized onboard computer.
- Evaluation of the implemented framework in closed-loop control flights.
- Introduction of the online estimation of the extrinsic calibration parameters and thus simplifying the use of the framework as well as improving its accuracy and consistency.
- Comparison with a state-of-the-art vision-based method in challenging environmental conditions (artificial fog).

This paper is organized as follows. Section II introduces the notation used in this paper. In Section III, we broadly describe our RIO algorithm. In Section V, we outline the experiments and evaluation conducted in order to validate the proposed method. In Subsection V-A, we report the experimental setup used during the experiments, the Subsection V-B presents the results of the closed-loop flight evaluation, whereas the Subsection V-C shows the impact of the online calibration on the accuracy and the consistency of our estimator. In Subsection V-D, we assess the performance of our method when used in visually degraded conditions. Finally, we present conclusions in Section VI.

## II. Preliminaries

Below, we introduce the notation for the easier following of the paper. A normally distributed multivariate variable is defined as $\mathbf{X}_i \sim \mathcal{N}(\mathbf{x}_i, \mathbf{\Sigma}_{ii})$, with a mean $\mathbf{x}_i$ and covariance (uncertainty) $\mathbf{\Sigma}_{ii}$, which is called the *belief* of $i$. Names of reference frames are capitalized and calligraphic, e.g., $\{\mathcal{I}\}$ for IMU. A pose between the reference frames $\mathcal{A}$ and $\mathcal{B}$ is defined as ${}^{\mathcal{A}}\mathbf{T}_{\mathcal{B}} = \begin{bmatrix} {}^{\mathcal{A}}\mathbf{R}_{\mathcal{B}} & {}^{\mathcal{A}}\mathbf{p}_{\mathcal{B}} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix} \in \mathrm{SE}(3)$ , with $\mathbf{R} \in \mathrm{SO}(3)$ and $\mathbf{p} \in \mathbb{R}^3$. The transformation of a coordinate vector ${}^{\mathcal{C}}\mathbf{p}_{P_1}$ pointing from the origin of the reference frame $\mathcal{C}$ to a point $P_1$, expressed in $\mathcal{C}$, can be transformed into the frame $\mathcal{A}$ by $\begin{bmatrix} {}^{\mathcal{A}}\mathbf{p}_{P_1} \\ 1 \end{bmatrix} = {}^{\mathcal{A}}\mathbf{T}_{\mathcal{C}} \begin{bmatrix} {}^{\mathcal{C}}\mathbf{p}_{P_1} \\ 1 \end{bmatrix}$ (read as ${}^{from}_{in}\mathbf{x}_{to}$). Rotations are stored as unit quaternions $\bar{\mathbf{q}} \in \mathrm{S}(3)$ with $\|\bar{\mathbf{q}}\| = 1$ allowing a direct mapping between rotation matrices and unit Hamiltonian quaternions by ${}^{\mathcal{A}}\mathbf{R}_{\mathcal{B}} = \mathbf{R}\{{}^{\mathcal{A}}\bar{\mathbf{q}}_{\mathcal{B}}\} \in \mathrm{SO}^3$ and ${}^{\mathcal{A}}\bar{\mathbf{q}}_{\mathcal{B}} = \bar{\mathbf{q}}\{{}^{\mathcal{A}}\mathbf{R}_{\mathcal{B}}\}$ [17]. The *a priori* and *a posteriori* of a belief are indicated by a $\{\bullet\}^{(-)}$ and $\{\bullet\}^{(+)}$, respectively. $\{\bullet\}^{\#}$ specifies measured (perturbed) quantities.

## III. Multi-State Radar-Inertial State Estimation With Persistent Landmarks And Online Calibration

We use the state estimator formulated in [15] and extend it with the online calibration capabilities of sensor extrinsic parameters which now constitute additional state variables in the estimator formulation. We use the stochastic cloning formalism from [18] in order to process measurements relative to the robot poses at past time instances. Cloned poses corresponding to the instants at which radar measurements were taken are appended to the state in a FIFO fashion and later used for pointclouds alignment necessary for matching and updates. Continuously detected and matched 3D points are recorded in the measurement trails. As long as a trail remains active, that is, new detections are matched and appended to it, all its 3D points are used for update with Eq. 3. Trails detected continuously for $N$ number of instants are promoted to *persistent landmarks*, moved to the state vector and used during update according to Eq. 4. To make use of the radial velocity measurements of 3D points provided by FMCW radars thanks to Doppler shift, we construct an update equation (Eq. 7) relating the projection of the estimated ego-velocity onto the normalized direction vectors to the detected 3D points. The full state vector $\mathbf{x}$ of our filter

is defined as follows:

$$\begin{aligned}\mathbf{x} = [\mathbf{x}_I; \mathbf{x}_R; \mathbf{x}_C; \mathbf{x}_L] = \big[\, \big[{}^{\mathcal{G}}\mathbf{p}_{\mathcal{I}}; {}^{\mathcal{G}}\bar{\mathbf{q}}_{\mathcal{I}}; {}^{\mathcal{G}}\mathbf{v}_{\mathcal{I}}; \mathbf{b}_{\mathbf{a}}; \mathbf{b}_{\boldsymbol{\omega}}\big]; \\ \big[{}^{\mathcal{I}}\mathbf{p}_{\mathcal{R}}; {}^{\mathcal{I}}\bar{\mathbf{q}}_{\mathcal{R}}\big]; \big[{}^{\mathcal{G}}\mathbf{p}_{\mathcal{I}_1}; {}^{\mathcal{G}}\bar{\mathbf{q}}_{\mathcal{I}_1}; \ldots; {}^{\mathcal{G}}\mathbf{p}_{\mathcal{I}_N}; {}^{\mathcal{G}}\bar{\mathbf{q}}_{\mathcal{I}_N}\big]; \\ \big[{}^{\mathcal{G}}\mathbf{p}_{\mathcal{L}_1}; \ldots; {}^{\mathcal{G}}\mathbf{p}_{\mathcal{L}_M}\big]\,\big]\end{aligned} \tag{1}$$

with the IMU state $\mathbf{x}_I$, the extrinsic calibration parameters $\mathbf{x}_R$, the stochastically cloned states $\mathbf{x}_C$ of the IMU poses corresponding to the previous radar measurements and the set of persistent landmarks $\mathbf{x}_L$. The previous radar measurements (point cloud of reflecting objects and their Doppler velocities) are not part of the state vector. ${}^{\mathcal{G}}\mathbf{p}_{\mathcal{I}}$; ${}^{\mathcal{G}}\mathbf{v}_{\mathcal{I}}$, and ${}^{\mathcal{G}}\bar{\mathbf{q}}_{\mathcal{I}}$ are the position, velocity, and orientation of the IMU/body frame $\{\mathcal{I}\}$ with respect to the navigation frame $\{\mathcal{G}\}$, respectively. $\mathbf{b}_{\boldsymbol{\omega}}$ and $\mathbf{b}_{\mathbf{a}}$ are the measurement biases of the gyroscope and accelerometer, respectively. $\big[{}^{\mathcal{I}}\mathbf{p}_{\mathcal{R}}; {}^{\mathcal{I}}\bar{\mathbf{q}}_{\mathcal{R}}\big]$ describe the 3D pose of the radar sensor with respect to the IMU sensor. $\big[{}^{\mathcal{G}}\mathbf{p}_{\mathcal{I}_n}; {}^{\mathcal{G}}\bar{\mathbf{q}}_{\mathcal{I}_n}\big]$ with $n = 1, \ldots, N$ define a set of past IMU poses with respect to the navigation frame $\{\mathcal{G}\}$ at the moments of past radar measurements. ${}^{\mathcal{G}}\mathbf{p}_{\mathcal{L}_m}$ with $m = 1, \ldots, M$ define the position of persistent landmarks $\{\mathcal{L}\}$ with respect to the navigation frame $\{\mathcal{G}\}$. We use $\big[{}^{\mathcal{G}}\mathbf{p}_{\mathcal{I}_1}; {}^{\mathcal{G}}\bar{\mathbf{q}}_{\mathcal{I}_1}\big]$ (corresponding to the newest coordinates of the trails) for ad-hoc point correspondence generation as detailed in [14] thanks to which the measured 3D points need not be kept in the state vector in order to be used in distance based measurements). After the addition of the sensor extrinsic calibration parameters as variables to the state vector the stochastic cloning procedure remains analogous to the one reported in [15], with the only difference being, that the covariance matrix blocks containing the cross-covariance terms copied upon the cloning of a new IMU state are bigger in size. Namely, the increase in size corresponds to the two $3\times3$ blocks which correspond to the cross-covariance values between the state-to-be-cloned and the calibration error states - position and orientation.

The evolution of the state is expressed by the following differential equations:

$$\begin{aligned}{}^{\mathcal{G}}\dot{\mathbf{p}}_{\mathcal{I}} &= {}^{\mathcal{G}}\mathbf{v}_{\mathcal{I}}, \\ {}^{\mathcal{G}}\dot{\mathbf{v}}_{\mathcal{I}} &= {}^{\mathcal{G}}\mathbf{R}_{\mathcal{I}}\left({}_{\mathcal{I}}\mathbf{a}^{\#} - \mathbf{b}_{\mathbf{a}} - \mathbf{n}_{\mathbf{a}}\right) + {}_{\mathcal{G}}\mathbf{g}, \\ {}^{\mathcal{G}}\dot{\mathbf{R}}_{\mathcal{I}} &= {}^{\mathcal{G}}\mathbf{R}_{\mathcal{I}}\left[{}_{\mathcal{I}}\boldsymbol{\omega}^{\#} - \mathbf{b}_{\boldsymbol{\omega}} - \mathbf{n}_{\boldsymbol{\omega}}\right]_{\times}, \\ \dot{\mathbf{b}}_{\mathbf{a}} &= \mathbf{n}_{\mathbf{b}_{\mathbf{a}}}, \dot{\mathbf{b}}_{\boldsymbol{\omega}} = \mathbf{n}_{\mathbf{b}_{\boldsymbol{\omega}}}, \\ {}^{\mathcal{G}}\dot{\mathbf{p}}_{\mathcal{I}_n} &= \mathbf{0}, {}^{\mathcal{G}}\dot{\mathbf{R}}_{\mathcal{I}_n} = \mathbf{0}, \\ {}^{\mathcal{I}}\dot{\mathbf{p}}_{\mathcal{R}} &= \mathbf{0}, {}^{\mathcal{I}}\dot{\mathbf{R}}_{\mathcal{R}} = \mathbf{0}, \\ {}^{\mathcal{G}}\dot{\mathbf{p}}_{\mathcal{L}_m} &= \mathbf{0}\end{aligned} \tag{2}$$

where $n = 1, \ldots, N$ refers to the most recent past IMU poses which are not changing in time, $m = 1, \ldots, M$ refer to $M$ most recent estimated positions of landmarks, ${}_{\mathcal{I}}\mathbf{a}^{\#}$ and ${}_{\mathcal{I}}\boldsymbol{\omega}^{\#}$ are the accelerometer and gyroscope measurements of the IMU with a white measurement noise $\mathbf{n}_{\mathbf{a}}$ and $\mathbf{n}_{\boldsymbol{\omega}}$. $\mathbf{n}_{\mathbf{b}_{\mathbf{a}}}$ and $\mathbf{n}_{\mathbf{b}_{\boldsymbol{\omega}}}$ are assumed to be white Gaussian noise to model the bias change over time as a random process. The gravity vector is assumed to be aligned with the z-axis of the navigation frame ${}_{\mathcal{G}}\mathbf{g} = [0, 0, 9.81]^{\mathsf{T}}$.

We use the error-state EKF formulation [19] and the below update equations to correct the nominal state from Eq.1. For each update component, we apply an outlier rejection using the chi-squared test. The first two components are distance measurements to points in the trail $d_{\mathcal{P}_j}$ and to the persistent landmarks $d_{\mathbf{l}_m}$:

$$d_{\mathcal{P}_j} = \left\| {}^{\mathcal{R}}\mathbf{p}'^{t_p}_{\mathcal{P}_j} \right\| \tag{3}$$

and

$$d_{\mathbf{l}_m} = \left\| \mathbf{l}'_m \right\| \tag{4}$$

with

$$\begin{aligned}{}^{\mathcal{R}}\mathbf{p}'^{t_p}_{\mathcal{P}_j} = {}^{\mathcal{I}}\mathbf{R}^{\mathsf{T}}_{\mathcal{R}}\big(-{}^{\mathcal{I}}_{\mathcal{I}}\mathbf{p}_{\mathcal{R}} + ({}^{\mathcal{G}}\mathbf{R}^{t_c}_{\mathcal{I}})^{\mathsf{T}}\big(-{}^{\mathcal{G}}\mathbf{p}^{t_c}_{\mathcal{I}} + \\ {}^{\mathcal{G}}\mathbf{p}^{t_p}_{\mathcal{I}} + {}^{\mathcal{G}}\mathbf{R}^{t_p}_{\mathcal{I}}\left({}^{\mathcal{I}}_{\mathcal{I}}\mathbf{p}_{\mathcal{R}} + {}^{\mathcal{I}}\mathbf{R}_{\mathcal{R}}\,{}^{\mathcal{R}}\mathbf{p}^{t_p}_{\mathcal{P}_j}\right)\big)\big)\end{aligned} \tag{5}$$

and

$$\mathbf{l}'_m = {}^{\mathcal{R}}_{\mathcal{R}}\mathbf{p}_{\mathcal{L}_m} = {}^{\mathcal{I}}\mathbf{R}^{\mathsf{T}}_{\mathcal{R}}\big({}^{\mathcal{G}}\mathbf{R}^{\mathsf{T}}_{\mathcal{I}}\big(\mathbf{l}_m - {}^{\mathcal{G}}_{\mathcal{G}}\mathbf{p}_{\mathcal{I}}\big) - {}^{\mathcal{I}}_{\mathcal{I}}\mathbf{p}_{\mathcal{R}}\big). \tag{6}$$

Where $t_p$ is the time index of a trail history element and $t_c$ is the current time index, $\mathcal{P}_j$ is a single 3D measurement point and ${}^{\mathcal{R}}_{\mathcal{R}}\mathbf{p}_{\mathcal{L}_m}$ is the radar observation of the 3D trail point in the current radar reference frame. $\mathbf{l}_m = {}^{\mathcal{G}}_{\mathcal{G}}\mathbf{p}_{\mathcal{L}_m}$ is the estimate of the $m$-th landmark.

The third component consists of the velocities of the points in the current pointcloud:

$$\begin{aligned}{}^{\mathcal{R}}\mathbf{v}_{\mathcal{P}_i} = \frac{\mathbf{r}^{\mathsf{T}}}{\|\mathbf{r}\|}\big({}^{\mathcal{I}}\mathbf{R}^{\mathsf{T}}_{\mathcal{R}}\,{}^{\mathcal{G}}\mathbf{R}^{\mathsf{T}}_{\mathcal{I}}\,{}^{\mathcal{G}}\mathbf{v}_{\mathcal{I}} + \\ {}^{\mathcal{I}}\mathbf{R}^{\mathsf{T}}_{\mathcal{R}}\left({}_{\mathcal{I}}\boldsymbol{\omega} \times {}^{\mathcal{I}}_{\mathcal{I}}\mathbf{p}_{\mathcal{R}}\right)\big)\end{aligned} \tag{7}$$

where $\mathbf{r} = {}^{\mathcal{R}}\mathbf{p}_{\mathcal{P}_i}$ is the 3D point detected in the current scan. The actual set of velocities used for the update is found using 3-point RANSAC as in [8].

## IV. Real-Time control of the UAV using the CNS Flight Stack

To perform a closed-loop flight evaluation, we leverage our previous work for a fully autonomous and customizable flight stack, the *CNS Flight Stack* [16]. Relying on SkiffOS [20], this flight stack provides a computationally lightweight, fully autonomous flight framework specifically designed for closed-loop estimator evaluation on MAVROS-controlled UAVs [21].

Within the *CNS Flight Stack*, the estimation module is replaced by the above-described RIO and its estimates are directly forwarded to a PX4-based controller [22] using the provided estimator bridge. Thus the RIO framework is the only observer in this control system.

Further, we configure the flight stack to fly a predefined trajectory (seen from the top-view in green in the Fig. 2) with varying waypoints in position and yaw. The maximum distance between two consecutive waypoints is set to 0.45 m horizontally and 1 m vertically.

The onboard processing times outlined in the Tab. II for the used resource-modest platform described in Subsection V-A show that our method is very lightweight with respect to the computation time and leaves a comfortable margin for the closed-loop control.

## V. Experiments And Evaluation

Below we describe the setup we used and the experiments we carried out to validate our framework in closed-loop flights as well as the results of the evaluation. We also show the results of the analysis of the impact of the online calibration on the accuracy and consistency using the manually flown trajectories from the dataset used in [15]. We choose to measure the accuracy of our algorithm by computing Mean Absolute Error (MAE) according to:

$$MAE = \frac{\sum_{i=1}^{N} |x_i - y_i|}{N}. \tag{8}$$

For consistency evaluation, we use the estimation error with its uncertainty and the Normalized Estimation Error Squared (NEES) (Bar-Shalom et al. [23]) according to:

$$NEES = \tilde{\mathbf{x}}^{\mathsf{T}} \mathbf{P}^{-1} \tilde{\mathbf{x}} \tag{9}$$

with $\tilde{\mathbf{x}}$ and $\mathbf{P}$ being the error between the true and estimated state variables and the error-state covariance matrix, respectively.

### A. Experimental Setup

We employ in our experiments a lightweight, inexpensive, single-chip FMCW radar from Texas Instruments integrated on an evaluation board AWR1843BOOST, shown affixed to the UAV in Fig. 1, equipped with a USB interface and powered with $2.5\,\mathrm{V}$. The frequency spectrum of chirps emitted by the radar lies between $f_l = 77\,\mathrm{GHz}$ and $f_u = 81\,\mathrm{GHz}$. The Field of View (FoV) is $120\,^\circ$ in azimuth and $30\,^\circ$ in elevation. We set the frequency of measurements acquisition to $f_m = 15\,\mathrm{Hz}$. The radar is attached to one extremity of the experimental platform facing forward by a tilt of about $45\,^\circ$ with respect to the horizontal plane as shown in Fig. 1. This improves the velocity readings compared to nadir view while keeping point measurements on the ground and thus at a reasonable distance. For inertial measurements, we use the IMU of the Pixhawk 4 flight controller unit (FCU) with the sampling frequency of $f_{si} = 200\,\mathrm{Hz}$. Transformation between the radar and IMU sensors is estimated online during the filter operation. For the dataset processing in Subsection V-C the initial navigation states of the filter are set to the ground truth values. For the closed-loop flight in Subsection V-B we set the initial navigation states to $\mathbf{p}_i = [0, 0, 0]^T$ and $\bar{\mathbf{q}}_i = [1, 0, 0, 0]^T$. Number of cloned states $N$ was set to 7. Some arbitrary reflective objects were placed in the scene since the test environment was otherwise a clutter-less clean lab space. No position information from the added objects of any sort was measured or used in our approach other than what the onboard radar sensor perceived by itself. We use a motion capture system to record the ground truth trajectories. During acquisition of the manually flown dataset used in Subsection V-C, we recorded sensor readings from the IMU and radar together with the poses of the UAV streamed by the motion capture system as ground truth. Our EKF-based RIO was in this case executed offline on the recorded sensor data on an Intel Core i7-10850H vPRO laptop with $16\,\mathrm{GB}$ RAM. In the case of the closed-loop flight in Subsection V-B, we recorded the pose estimates calculated (and fed to the controller) onboard in real-time by our RIO. Our RIO was executed on a raspberry pi 4 with $4\,\mathrm{GB}$ RAM mounted on the platform. Presented RIO is a custom C++ framework and in both cases we compiled it with gcc at -O3 optimization level. Execution timings for the closed-loop flight are shown in Tab. II.

### B. Closed-Loop Flights Evaluation

For evaluation of the presented RIO approach, we design a 3D trajectory such that the UAV can execute it within the motion capture covered area. The top view of the trajectory can be seen in green in the Fig. 2. The UAV then executes this trajectory twice within a single mission using the state estimates computed by our RIO implementation onboard in real-time. We executed three missions and computed the mean values of the final drift and the norm of MAE for each of them. Our estimator exhibited stable behaviour in all performed flights. Computed mean values are $2.84\,\%$ for the final drift and $0.58\,\mathrm{m}$ for $\|MAE\|$ (see Tab. I). The mean traveled distance across trajectories is $28.62\,\mathrm{m}$. As can be seen in the Fig. 4, the accuracy (measured by $\|MAE\|$) obtained in closed-loop flights, marked with a brown star, lies slightly above the mean calculated over the manually flown and subsequently offline processed dataset of seven trajectories collected in [15]. We attribute this slim discrepancy to the online closed-loop execution of the estimator and controller on a resource-constrained hardware during which some of the measurements are dropped due to CPU load spikes. We note that these are precisely the challenges when demonstrating estimation approaches in closed-loop autonomous flights and that these are important factors our RIO approach is able to mitigate well. In the Fig. 3 we plot the 2D coordinates of the estimated and true position. In Tab. II we show the average values of the time it takes to execute the entire IMU and radar processing functions on the onboard computer mentioned in Subsection V-A.

TABLE I
Metrics computed across closed-loop flights

| Trajectory | Distance [m] | Norm of MAE at full distance [m] | Final drift [%] |
|---|---|---|---|
| 1 | 28.43 | 0.64 | 3.41 |
| 2 | 28.88 | 0.37 | 2.05 |
| 3 | 28.55 | 0.73 | 3.07 |
| Average | **28.62** | **0.58** | **2.84** |

TABLE II
Onboard execution timings

| Average time [ms] | |
|---|---|
| IMU processing | Radar processing |
| 1.04 | 16.97 |

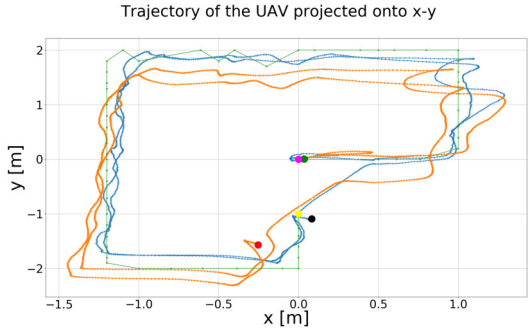


Fig. 2. Top view of one of the executed closed-loop flights. Marked in blue and orange are the estimated and true trajectories respectively. Plotted in green is the trajectory input to the controller. Coloured dots are the starting and end points. The magenta-coloured dot marks the starting point of the input trajectory, whereas the dots marking the starting points of the ground truth and estimate coincide.

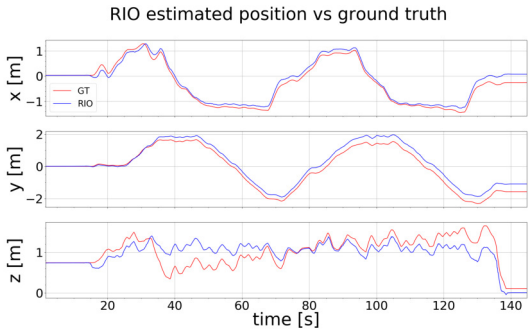


Fig. 3. Position coordinates of the true and estimated trajectories plotted against time for one of the closed-loop flights. The take-off point was on the table seen in the Fig. 1 which causes vertical shift in the z coordinate. The landing point was on the floor. Note that no special handling in the filter is needed in the steady initial phase when the robot does not move since no feature triangulation is required but direct depth information is provided by the radar – an important advantage over vision based approaches.

### *C. Evaluation of the Impact of the Online Calibration on the Accuracy and Consistency*

Estimation accuracy is impacted by the precision of the extrinsic calibration parameters of the sensors. Also, as stated in [24], the unmodelled uncertainty of these parameters when treated as static negatively impacts the filter's consistency. Therefore, we propose to estimate these parameters online during the operation of the algorithm. This renders the platform easier to use as the calibration parameters are not needed to be measured manually nor any complex calibration procedure is required. Online calibration estimation also increases the accuracy which can be seen in Fig. 4 where we note the reduction of the $\|MAE\|$ for the same dataset used in [15], but with the here proposed online calibration implemented. Additionally, based on the analysis done in [25] and [26], we know that these parameters are observable for general trajectories.

In the Fig. 5 and Fig. 6, we can see comparisons of the position estimation errors (with $\pm 3\sigma$ envelopes) and the position NEES respectively for the case of manual and online calibration for one of the trajectories from the above

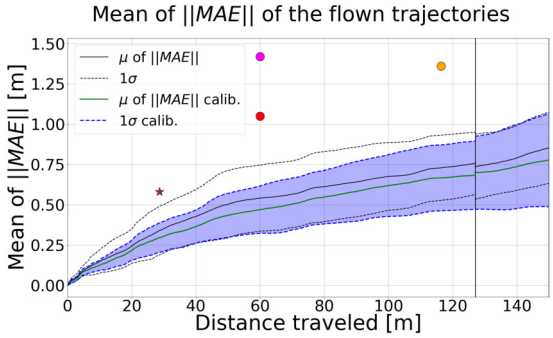


Fig. 4. Mean of the norm of MAE for the flown dataset of seven trajectories from [15]. Black dashed lines are from [15] whereas the coloured lines are the values obtained for the same dataset but with the online calibration implemented. As can be noticed there is an increase in accuracy (mean of $\|MAE\|$ reduced). Marked with the brown star is the mean value obtained for the closed-loop flights presented in the previous paragraph (Tab. I). For comparison, red and orange dots represent results presented in [14] and the magenta dot represent the state-of-the-art result in [8] for the flown trajectory. From 127 m to 150 m six trajectories are used to compute the mean since trajectory 1 ended at 127 m (solid black vertical line).

mentioned dataset. In Fig. 5, we can observe that our system with online calibration exhibits better characterization of the actual uncertainty as the estimation errors are bounded by the $\pm 3\sigma$ envelopes across the vast majority of the trajectory and the errors oscillate closer to zero. The bounded segments are less frequent for the case of the manual calibration and the errors are shifted further away from zero. These aspects are also reflected in Fig. 6 which depicts the comparison of the NEES for the same data as for the position errors above. The above comparisons indicate much better consistency for the case with the online calibration since the NEES is greatly reduced (Bar-Shalom et al. [23]). In the case of manual calibration, we paid close attention to measuring as accurately as possible the extrinsic parameters of the sensors. Manually measured calibration parameters were used as the initial values for the case of online calibration.


Fig. 5. Comparison of the position estimation errors and the $\pm 3\sigma$ uncertainty envelopes for one of the trajectories from the dataset in [15] with the manual and the online calibration (green and red respectively). As can be seen in the case of the online calibration, the errors are more thoroughly bounded by the uncertainty across the trajectory than in the case of the manual calibration. Since the estimator is inherently inconsistent [24], one cannot expect the errors to be bounded $99.7\,\%$ of the time.

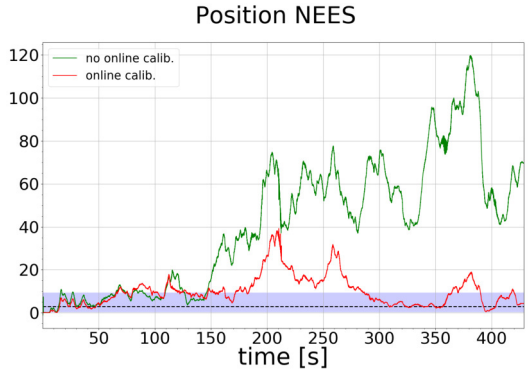


Fig. 6. Comparison of the position NEES for one of the trajectories with the manual and the online calibration (green and red respectively). The filter consistency is greatly improved with the online calibration which can be seen in the significant reduction of the NEES, which in the case of the online calibration does not continue to grow and more often remains bounded within the two-sided 95 % probability concentration region (shaded blue) than in the case of manual calibration.

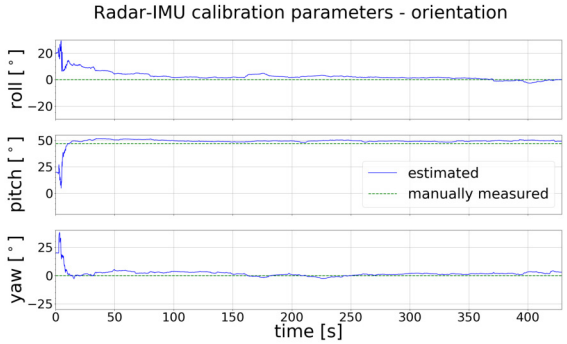


Fig. 7. Convergence of the rotational extrinsic calibration parameters for the system with extremely high initial calibration errors. Including the extrinsic calibration parameters in the state vector enables the system to recover from extremely inaccurate initial calibration values and eventually converge to the real values reflecting the tilt of the radar of about 45 ° around the y axis (we manually measured 47 °) around the y-axis. Green dashed lines are used to mark the manually measured extrinsic calibration parameters values which we consider the true values since we do not have a precise ground truth on these quantities.

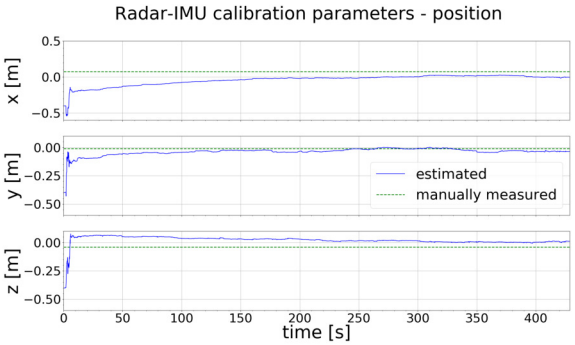


Fig. 8. Convergence of the translation extrinsic calibration parameters for the system with extremely high initial calibration errors. Green dashed lines are used to mark the manually measured extrinsic calibration parameters values which we consider the ground truth.

In the Fig. 7 and Fig. 8 we plot the convergence of the extrinsic calibration parameters to their measured values for one of the trajectories in the case where the initial extrinsic calibration parameters were initialized with vastly erroneous values, that is, $\mathbf{c}_{trans}^{init} = [x = -40\,\text{cm}, y = -40\,\text{cm}, z = -40\,\text{cm}]$ in position and $\mathbf{c}_{rot}^{init} = [roll = 20\,°, pitch = 20\,°, yaw = 20\,°]$ in orientation. Extrinsic calibration parameters which we measured manually with simple tools such as a ruler and a protractor were equal to $\mathbf{c}_{trans}^{meas} = [7.5\,\text{cm}, -1.0\,\text{cm}, -4.0\,\text{cm}]$ and $\mathbf{c}_{rot}^{meas} = [0\,°, 47\,°, 0\,°]$. Given the lack of ground truth for these parameters, we consider the manually measured values the true ones. As observed, our filter successfully recovers from such high initial calibration errors and the parameters converge very close to their manually measured values. The difference between measured and estimated values are $\mathbf{c}_{trans}^{diff} = [6.4\,\text{cm}, 2.6\,\text{cm}, 5.0\,\text{cm}]$ and $\mathbf{c}_{rot}^{diff} = [0.087\,°, 2.50\,°, 3.27\,°]$ in position and rotation respectively for this particular run which is representative for the majority of the runs. These numbers are comparable to [27] where the authors used synthetic data with a dedicated calibration procedure while we use real data during a regular UAV flight with no dedicated calibration procedure.

Even though the estimated and measured values are similar in the estimator's asymptotic behavior, the previously discussed consistency improvement is noticeable. This stems from the filter's ability to include the calibration states for its energy dissipation rather than being hindered to adapt these states online.

For completeness of the discussion about our estimator consistency, we note that we have not implemented yet the approach suggested in [28]. Thus we expect further consistency improvement by doing so in future work.

### *D. Evaluation of the State Estimation in the Artificial Fog*

As depicted in the Fig. 1, we evaluate our estimation framework in dense artificial fog in order to showcase the benefits of the radar-based navigation in visually degraded conditions. To perform the comparison, we moved the UAV seen in the Fig. 1 arbitrarily in a hand-held fashion through a dense artificial fog. We moved the platform manually instead of flying, since as our experiments showed, with the propellers turned on, the amount of the artificial fog we were able to create with the fog machine was quickly dispersed and the evaluation of the impact of the fog became impossible. We further compare the performance of our RIO to a state-of-the-art EKF-based VIO implementation, *OpenVINS* [29]. For VIO, a Matrixvision mvBlueFOX-MLC USB2.0 camera is mounted rigidly next to the radar sensor (seen in the Fig. 1) providing images at 20 Hz. For closed-loop computation performance reasons, the VIO is post-processed offline on the desktop hardware described in the Subsection V-A. Sequence of the results produced by the feature tracker of OpenVINS is shown in Fig. 9. As expected, the tracker struggles to find sufficient features and to track the few found ones correctly.

The performance of the two approaches is shown in Fig. 10. VIO diverges almost immediately when the vehicle

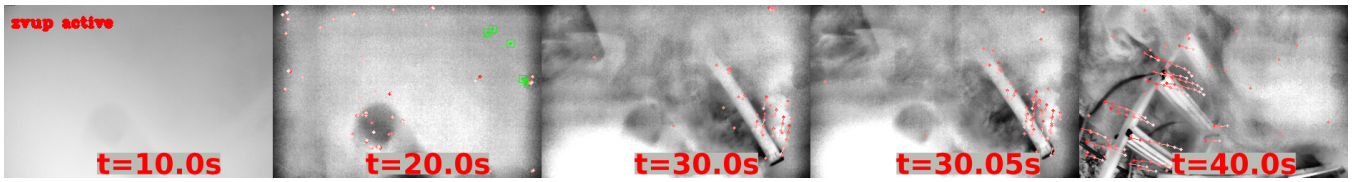


Fig. 9. Sequence of the images used by the VIO framework OpenVINS for tracking features and state estimation. Due to the fog, features can barely be tracked in the first 30 seconds. Thus, OpenVINS struggles to keep track of persistent features (marked in green) and to perform proper state estimation.

starts moving as it cannot track features correctly in the dense fog. In comparison, our RIO framework successfully continues to estimate the robot pose despite the fog since the radar waves can penetrate it. Thus, in environments such as dense fog, darkness or strong light our RIO approach can be successfully used to estimate the vehicle states correctly and thus be employed to close the feedback control loop in autonomous flights in settings such as disaster zones.

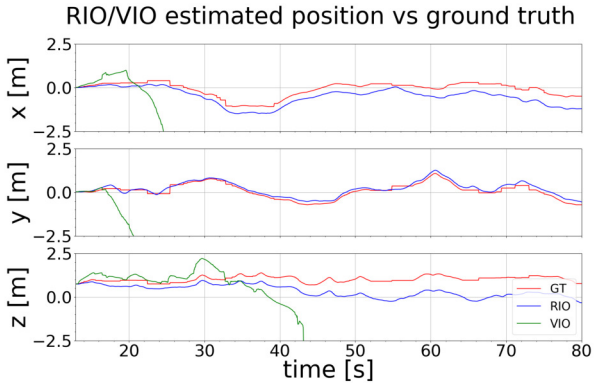


Fig. 10. Comparison of position estimates of a VIO implementation [29] and our RIO approach. Note the almost immediate divergence of the VIO when faced with the dense artificial fog. As can also be seen, our RIO method proves to be robust against it.

## VI. Conclusions

In this paper we demonstrated in real-world flights the real-time closed-loop control capabilities of the RIO framework presented in [15] and extended it with the online extrinsic parameters calibration capability. Closed-loop experiments showed that our method scales well towards commonplace issues related to real-time onboard operation of state estimation frameworks on resource-modest platforms such as sporadic measurement losses due to CPU load spikes. We also showed how the online calibration impacts the accuracy and the consistency of our state estimator. To our knowledge, it is the first time that a RIO system relying *solely* on a low-cost and lightweight IMU and radar sensors is shown to successfully enable an autonomous closed-loop flight of a UAV equipped with a resource-constrained computing hardware. Moreover, the presented framework needs no knowledge of the environment to operate, can be used in GNSS-denied settings, and, as also shown in the present paper, can be deployed in visually degraded environments.

## References

[1] C. D. Monaco and S. N. Brennan, "Radarodo: Ego-motion estimation from doppler and spatial data in radar images," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 3, pp. 475–484, 2020.

[2] S. H. Cen and P. Newman, "Precise ego-motion estimation with millimeter-wave radar under diverse and challenging conditions," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6045–6052.

[3] D. Barnes and I. Posner, "Under the radar: Learning to predict robust keypoints for odometry estimation and metric localisation in radar," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9484–9490.

[4] Y. S. Park, Y.-S. Shin, and A. Kim, "Pharao: Direct radar odometry using phase correlation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2617–2623.

[5] R. Aldera, M. Gadd, D. De Martini, and P. Newman, "What goes around: Leveraging a constant-curvature motion constraint in radar odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7865–7872, 2022.

[6] K. Burnett, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, "Radar odometry combining probabilistic estimation and unsupervised feature learning," *CoRR*, vol. abs/2105.14152, 2021. [Online]. Available: https://arxiv.org/abs/2105.14152

[7] Y. Z. Ng, B. Choi, R. Tan, and L. Heng, "Continuous-time radar-inertial odometry for automotive radars," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 323–330.

[8] C. Doer and G. F. Trommer, "An ekf based approach to radar inertial odometry," in *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2020, pp. 152–159.

[9] ——, "Radar inertial odometry with online calibration," in *2020 European Navigation Conference (ENC)*. IEEE, 2020, pp. 1–10.

[10] ——, "Yaw aided radar inertial odometry using manhattan world assumptions," in *2021 28th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)*, 2021, pp. 1–9.

[11] C. Doer, J. Atman, and G. F. Trnmmer, "Gnss aided radar inertial odometry for uas flights in challenging conditions," in *2022 IEEE Aerospace Conference (AERO)*, 2022, pp. 1–10.

[12] A. Kramer, C. Stahoviak, A. Santamaria-Navarro, A.-A. Agha-Mohammadi, and C. Heckman, "Radar-inertial ego-velocity estimation for visually degraded environments," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5739–5746.

[13] A. Kramer and C. Heckman, *Radar-Inertial State Estimation and Obstacle Detection for Micro-Aerial Vehicles in Dense Fog*, 03 2021, pp. 3–16.

[14] J. Michalczyk, R. Jung, and S. Weiss, "Tightly-coupled ekf-based radar-inertial odometry," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 12 336–12 343.

[15] J. Michalczyk, R. Jung, C. Brommer, and S. Weiss, "Multi-state tightly-coupled ekf-based radar-inertial odometry with persistent landmarks," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 4011–4017.

[16] M. Scheiber, A. Fornasier, R. Jung, C. Böhm, R. Dhakate, C. Stewart, J. Steinbrener, S. Weiss, and C. Brommer, "CNS Flight Stack for Reproducible, Customizable, and Fully Autonomous Applications," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 283–11 290, 2022.

[17] H. Sommer, I. Gilitschenski, M. Bloesch, S. Weiss, R. Siegwart, and J. Nieto, "Why and how to avoid the flipped quaternion multiplication," *Aerospace*, vol. 5, no. 3, p. 72, 2018.

[18] S. I. Roumeliotis and J. W. Burdick, "Stochastic cloning: A generalized framework for processing relative state measurements," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2. IEEE, 2002, pp. 1788–1795.

[19] P. Maybeck, *Stochastic Models, Estimation and Control*, ser. Mathematics in science and engineering. Academic Press, 1979, no. pt. 1. [Online]. Available: https://books.google.at/books?id=eAdRAAAAMAAJ

[20] C. Stewart, "SkiffOS: Minimal Cross-compiled Linux for Embedded Containers," Mar. 2021, arXiv:2104.00048 [cs]. [Online]. Available: http://arxiv.org/abs/2104.00048

[21] S. Atoev, K.-R. Kwon, S.-H. Lee, and K.-S. Moon, "Data analysis of the MAVLink communication protocol," in *2017 International Conference on Information Science and Communications Technologies (ICISCT)*. Tashkent: IEEE, Nov. 2017, pp. 1–3. [Online]. Available: http://ieeexplore.ieee.org/document/8188563/

[22] L. Meier, D. Honegger, and M. Pollefeys, "PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, USA: IEEE, May 2015, pp. 6235–6240. [Online]. Available: http://ieeexplore.ieee.org/document/7140074/

[23] Y. bar shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*, 01 2004.

[24] M. Li and A. Mourikis, "High-precision, consistent ekf-based visual–inertial odometry," *The International Journal of Robotics Research*, vol. 32, pp. 690–711, 05 2013.

[25] E. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *I. J. Robotic Res.*, vol. 30, pp. 407–430, 12 2011.

[26] J. Kelly and G. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *I. J. Robotic Res.*, vol. 30, pp. 56–79, 01 2011.

[27] E. Wise, J. Peršić, C. Grebe, I. Petrović, and J. Kelly, "A continuous-time approach for 3d radar-to-camera extrinsic calibration," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 164–13 170.

[28] G. Huang, M. Kaess, and J. J. Leonard, "Towards consistent visual-inertial navigation," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 4926–4933.

[29] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "Openvins: A research platform for visual-inertial estimation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4666–4672.