UAV Passive ISAC on OFDM: Minimal Demonstrator

Haytam Zerdani

01/08/2025

Abstract

This report presents a minimal but complete passive Integrated Sensing and Communication (ISAC) demonstrator for UAV scenarios. The system reuses an OFDM downlink–like waveform as an *opportunistic illuminator*, forms reference and surveillance channels, and estimates delay and Doppler of moving targets via cross-ambiguity processing and a slow-time FFT. The same baseband also supports a conventional OFDM receiver to measure EVM and BLER (hooks provided). The code is modular (Python), reproducible through YAML-like configs, and designed as a starting point for Thales 6G ISAC investigations.

1 Objectives

- Build a modular passive sensing chain reusing OFDM waveforms.
- Produce range–Doppler (RD) maps and basic radar KPIs (e.g., qualitative P_d at a target bin).
- Prepare hooks for communications KPIs (EVM/BLER) under the same channel and impairments.
- Enable trade-off studies between sensing performance and communications throughput.

2 System Model

Let $x[n]$ be a discrete-time complex baseband OFDM signal with sampling rate f_s. We consider a parametric delay–Doppler channel

$$y[n] = \sum_{k=1}^{K} \alpha_k \, x(n - \tau_k) \; e^{j2\pi\nu_k n/f_s} + w[n], \tag{1}$$

where $\alpha_k \in \mathbb{C}$ are path gains, τ_k are (possibly fractional) delays in samples, ν_k the Doppler frequencies in Hz, and $w[n]$ complex AWGN. The **reference** channel observes $x[n]$. The **surveillance** channel observes $y[n]$ which contains the direct path, clutter, multipath, and moving targets.

The *cross-ambiguity* (fast-time correlation) is estimated blockwise:

$$r_m[\ell] = \sum_{n=0}^{L-1} s_m[n] \, r_m^*[n-\ell], \quad m = 0, \ldots, M-1, \tag{2}$$

with windowed blocks of length L and slow-time index m. A slow-time FFT across m forms the 2-D RD map:

$$R[d, \ell] = \sum_{m=0}^{M-1} r_m[\ell] \; e^{-j2\pi md/M}. \tag{3}$$

3 Methodology

Waveform. We generate CP-OFDM with QPSK data and comb pilots. Each OFDM symbol is IFFT-shifted, CP-padded, and concatenated.

Channel and Targets. The surveillance signal is synthesized by superimposing delayed and Doppler-shifted copies of the reference (fractional delays via frequency-domain linear phase) plus a scaled clutter term and weak random multipaths. AWGN level is set by an SNR parameter.

Processing. We segment the streams into overlapping blocks. For each block we compute cross-correlation in the frequency domain, then perform a slow-time FFT to obtain the RD magnitude map. A simple CA-CFAR routine is provided and can be enabled to produce a binary detection mask.

KPIs. In this minimal demonstrator we visualize RD peaks; planned extensions add ROC (P_d vs P_{fa}) via CFAR and communications EVM/BLER under shared conditions. Joint trade-offs (throughput vs P_d) are obtained by reserving a fraction α of PRBs or power for sensing pilots.

4 Implementation Overview

The repository is organized as follows:

- `waveform/ofdm.py`: OFDM generator with QPSK and comb pilots.
- `channel/delay_doppler.py`: fractional delay and Doppler operators; surveillance mixer.
- `radar_rx/rdmap.py`: block cross-correlation + slow-time FFT for RD maps.
- `radar_rx/cfar.py`: basic 2D CA-CFAR (optional).
- `metrics/plots.py`: plotting utilities for RD visualization.
- `sims/uav_passive.py`: end-to-end demonstrator script.

5 Key Equations

Fractional delay. Using an N-point FFT, a fractional delay Δ samples is applied by a linear phase ramp in frequency:

$$Y[k] = X[k]\, e^{-j2\pi k\Delta/N}, \qquad y[n] = \mathrm{IFFT}\{Y[k]\}. \tag{4}$$

Slow-time Doppler FFT. After fast-time correlation $r_m[\ell]$, Doppler bins are obtained by

$$R[d,\ell] = \mathrm{FFT}_m\{r_m[\ell]\}, \quad d = 0,\ldots,M-1. \tag{5}$$

CFAR (sketch). With a cell-under-test (CUT), guard cells, and N_{ref} reference cells, a CA-CFAR threshold is

$$T = \alpha \cdot \hat{\sigma}^2, \qquad \alpha = N_{\mathrm{ref}}\big(P_{fa}^{-1/N_{\mathrm{ref}}} - 1\big). \tag{6}$$

6 Results

Figure 1 shows a representative RD map with two synthetic moving targets on top of clutter and AWGN. Bright spots localize delay bins (*range*) and Doppler bins (*velocity*). Peaks align with the injected delays and Dopplers in the simulator configuration.

7 Reproducibility

Environment. Python 3.11 with `numpy`, `scipy`, `matplotlib`.

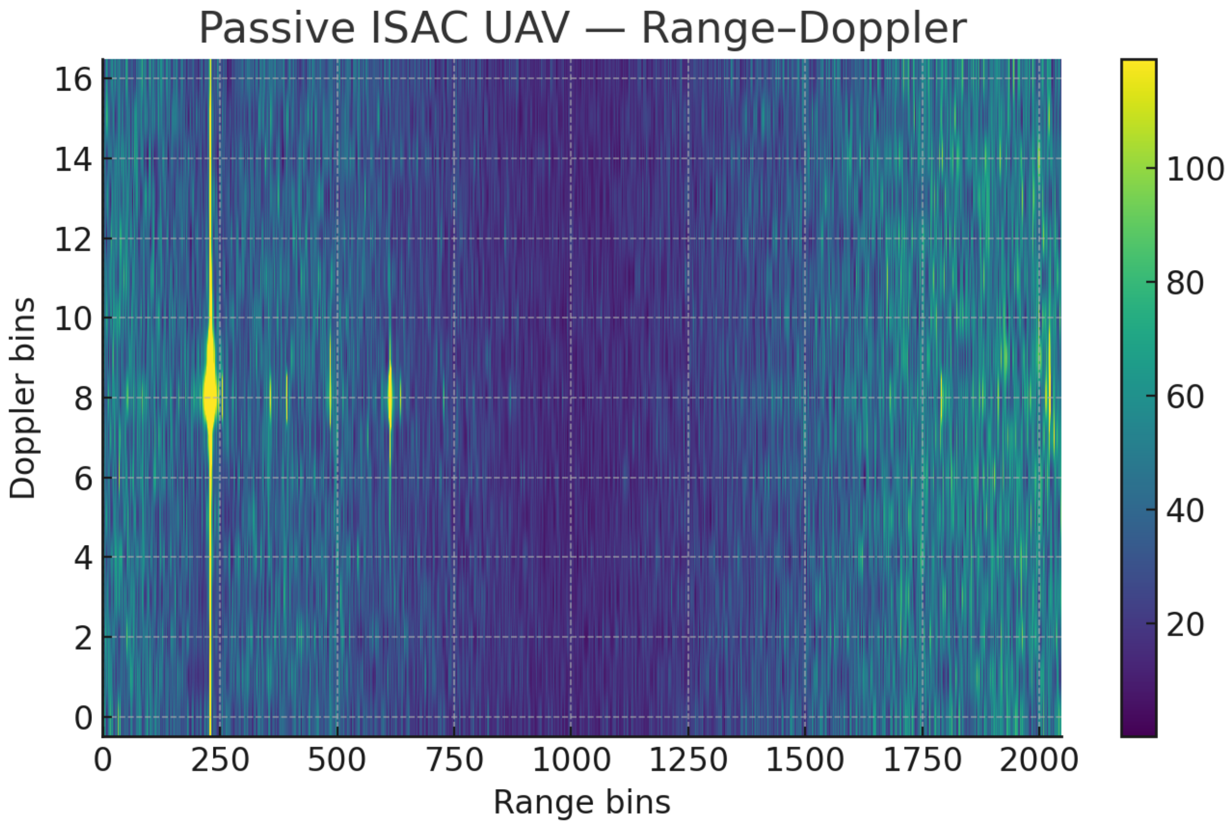

Figure 1: Range–Doppler map of the surveillance vs. reference channels (two targets visible).

Run. From the repository root:

```
python -m sims.uav_passive
# RD plot saved under data/results/rd_map.png
```

Key parameters. Main simulation arguments inside `sims/uav_passive.py`:

Parameter	Default
Subcarriers N_{sc}	256
OFDM symbols N_{sym}	64
CP length	32
Pilot spacing	16
Sampling rate f_s	15.36 MHz
Block length L	2048 samples
Slow-time blocks K	48
Overlap	0.5
SNR (surveillance noise)	20 dB

8 Planned Extensions

- **CFAR overlay and ROC**: enable the included CA-CFAR, sweep SNR and clutter to obtain P_d vs P_{fa}.
- **Communications KPIs**: add OFDM synchronization, LS/MMSE equalization, EVM and BLER estimation under the same channel and impairments.

- **Resource split** α: reserve PRBs or power for sensing pilots, generate throughput vs P_d Pareto curves.
- **Impairments**: CFO, phase noise, quantization; sensitivity analysis for each.
- **Real IQ ingestion**: replay USRP logs in `data/iq_logs/` to validate with field captures.

9 Conclusions

The demonstrator delivers a compact, extensible pipeline for passive ISAC with OFDM illumination. It produces interpretable RD maps, isolates target signatures in delay–Doppler, and provides the scaffolding to quantify joint sensing–communications trade-offs required by 6G ISAC use cases.

Repository: `/mnt/data/isac` **Zip:** `UAV_Passive_ISAC_OFDM_Demo.zip`