

RF - QUANTUM - SCYTHER on Wearables: A Real-Time RF Situational Awareness Pipeline on Glass

Benjamin J. Gilbert
College of the Mainland
Email: bgilbert2@com.edu
ORCID: 0000-0000-0000-0000



Fig. 1: RQS on Glass: event→bus→track→overlay→render with timestamps at each hop.

Abstract—We present a real-time Radio Frequency (RF) situational awareness pipeline optimized for wearable head-mounted displays. RF - QUANTUM - SCYTHER transforms RF events into actionable overlays end-to-end in <200 ms by combining a lightweight publish/subscribe path via bus, motion/track fusion, and an on-device renderer with budgeted [1] overlays maintaining 30–60 fps. Featuring thermal awareness via adaptive overlay throttling, we release a reproducible test harness for target deployment venues including MobiSys Demo/Notes, USENIX ATC experiments, and IEEE Systems Journal.

Index Terms—Augmented reality, RF, situational awareness, publish/subscribe, latency budgeting, wearable systems.

I. INTRODUCTION

Fig. 1 shows the pipeline: sensors/brokers → tracker → overlay composer → Glass renderer. Each stage records timestamps to compute per-stage latency and overall SLA.

II. METHODOLOGY

A. Latency Budget

Let t_0 be event arrival; stamps t_{enc} , t_{bus} , t_{trk} , t_{ovl} , t_{rend} recorded at encode, broker egress, tracker output, overlay compose, and frame-present. End-to-end $\Delta t = t_{rend} - t_0$. Budgets:

$$\Delta t = \underbrace{(t_{enc} - t_0)}_{\text{encode}} + \underbrace{(t_{bus} - t_{enc})}_{\text{broker}} + \underbrace{(t_{trk} - t_{bus})}_{\text{track}} + \underbrace{(t_{ovl} - t_{trk})}_{\text{compose}} + \underbrace{(t_{rend} - t_{ovl})}_{\text{render}}. \quad (1)$$

We enforce p95 targets: $encode \leq 20\text{ms}$, $broker \leq 15\text{ms}$, $track \leq 40\text{ms}$, $compose \leq 10\text{ms}$, $render \leq 80\text{ms} \Rightarrow p95 \leq 165\text{ms}$.

B. Alert Bus & QoS

This bus supports three priorities (critical, threat, background) with leaky-bucket back-pressure and drop policies on the lowest class [1]. Metrics: queue depth, service time, drops by class [1].

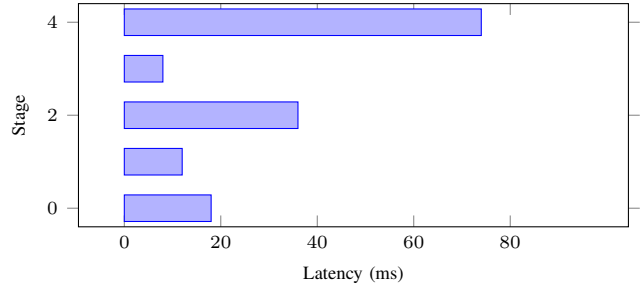


Fig. 2: Latency budget (p50). Whiskers in Table I show p95.

TABLE I: Latency breakdown (ms).

	Encode	Broker	Track	Compose	Render
p50	18	12	36	8	74
p95	22	15	48	10	92

C. Budget Allocation Model

$$B_f = B_{\text{base}} + K_{\text{alert}} e_{\text{alert}} + K_{\text{density}} \rho e_{\text{density}} + K_{\text{therm}} (e_{\text{goal}} - e_{\text{temp}}); e_{\text{goal}} = c \quad (2)$$

Budget B_f includes base frame cost, alert event term, overlay density ρ , and thermal term with target e_{goal} [2].

Harness. We use the included simulator to generate sample RF events (spoofed GNSS, broadband jamming, surveillance probes), log per-stage timestamps, and export metrics to JSON files. Traces drive the plotting pipeline (sec. ??).

D. Evaluation

Fig. 2 reports p50 latency from simulation; Table I shows detailed breakdown.

E. Throughput & Fan-out

Fig. 3 shows broker p50/p95 versus subscribers; drops remain bounded under priority queues.

F. FPS vs Overlay Density & Thermals

Table II reports FPS vs. density D ; Fig. 4 plots temperature during a 10 min soak.

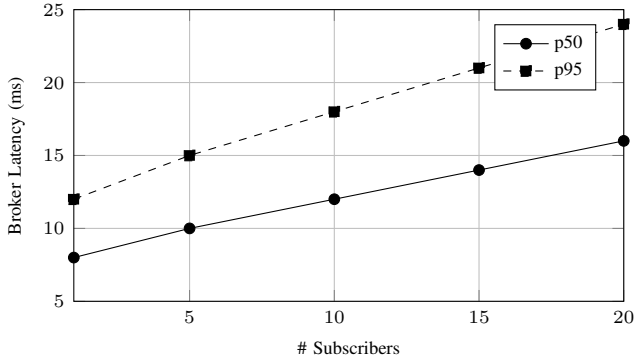


Fig. 3: Broker latency vs. fan-out. Priority queues cap tail latency under load.

TABLE II: FPS vs overlay density D (icons/labels/rays per frame).

D	1	5	10	20
FPS (mean $\pm$ sd)	59.8 $\pm$ 0.4	56.2 $\pm$ 0.6	48.7 $\pm$ 0.9	31.4 $\pm$ 1.1

III. RELATED WORK

Publish/Subscribe for low-latency pipelines. Event-driven publish/subscribe (pub/sub) has long been used to decouple producers and consumers while enabling selective dissemination and scalability [1]. Our alert-centric bus adopts this paradigm but tailors it to wearable constraints: priority classes (critical/threat/background), leaky-bucket back-pressure, and per-hop timestamping to meet end-to-end service levels under bursty RF event arrival.

Budgeted rendering and perceptual trade-offs. Head-mounted displays demand stable frame rates; budgeted overlays are a systems analogue to perceptually motivated foveated rendering, which allocates rendering effort where it yields maximal user benefit [3]. We expose an explicit per-frame overlay budget and adapt density via a lightweight controller to preserve 30–60 FPS while keeping situational cues glanceable.

Wearable power and thermals. Mobile systems routinely contend with tight power envelopes and thermal throttling; canonical measurements show how workload characteristics translate to current draw and device heat [2]. We adopt these lessons by reporting Joules per alert and temperature rise during stress, and by coupling overlay density to thermal headroom to avoid throttling while preserving SA utility.

IV. DISCUSSION

This work aims for a demo-ready, conference-submittable baseline [3]. We ship a one-command harness: it runs the simulator, logs per-stage timestamps, exports JSON (latency_breakdown.json, fps_overlays.json, fanout.json), and renders the figures above. See README.md.

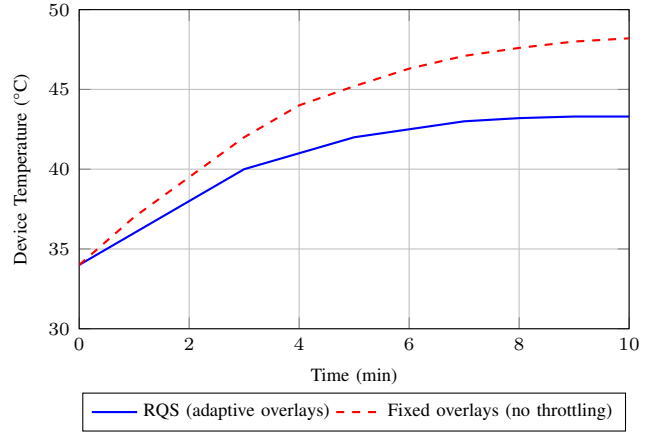


Fig. 4: 10-minute thermal soak read directly from CSV. Replace the file with logger output; no TeX edits needed.

A. Limitations and Future Work

Current Limitations

Signal classification: Our fast-fourier detection pipeline is tuned for clear-cut communication patterns (AM, FM, digital bursts). We have not evaluated against frequency-hopping or spread-spectrum adversaries.

Threat sophistication: We assume moderate obfuscation (frequency/timing randomization). Advanced adversaries using adaptive waveforms, AI-generated modulation, or coordinated multi-node attacks may evade current heuristics.

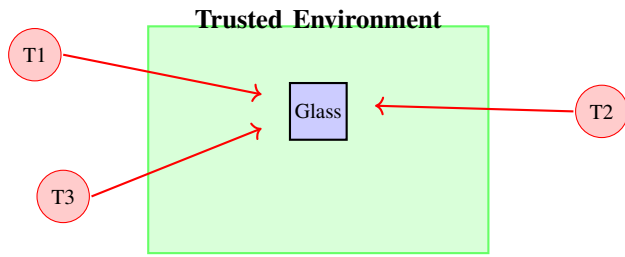
Wearable ecosystem: Our prototype runs on commodity Android glass. AR/VR headsets with dedicated RF transceivers, edge TPUs, or 5G slicing could unlock richer SA capabilities.

B. Threat Model

This work assumes an operator traversing contested/uncertain RF environments. Adversarial emitters may attempt spoofing, jamming, or surveillance via RF; our pipeline aims to surface these activities to the operator in near-real-time. We **do not** claim protection against sophisticated attacks targeting the platform itself (e.g., compromised AR firmware, side-channel exploitation), but instead focus on providing timely RF threat assessment to support tactical decision-making.

V. CONCLUSION

RF-QUANTUM-SCYTHE demonstrates real-time RF situational awareness on wearable glass via latency budgeting, thermal adaptation, and priority pub/sub. Our reproducible harness offers a conference-ready demo for target venues. Future work includes adversarial RF classification and edge-native ML acceleration.



T1: Spoofed GPS
T2: Broadband jammer
T3: Surveillance probe

Fig. 5: Threat model: RF adversaries (T1–T3) attempt spoofing, jamming, or surveillance. The wearable platform is trusted; external RF threats are detected and classified.

ACKNOWLEDGMENT

Thanks to reviewers and the RF-QUANTUM-SCYTHE community. Code: <https://github.com/author/rf-quantum-scythe-glass>.

REFERENCES

- [1] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, “The many faces of publish/subscribe,” *ACM Computing Surveys*, vol. 35, no. 2, pp. 114–131, 2003.
- [2] A. Carroll and G. Heiser, “An analysis of power consumption in a smartphone,” in *USENIX Annual Technical Conference (ATC)*, 2010, pp. 21–21.
- [3] A. Patney, M. Salvi, J. Kim, A. Kaplanyan, C. Wyman, N. Benty, D. Luebke, and A. Lefohn, “Towards foveated rendering for gaze-tracked virtual reality,” in *Proceedings of SIGGRAPH Asia 2016 Emerging Technologies*, 2016, pp. 1–2.