Reliable Edge Intelligence for Urban Flood Alerts

A latency-aware architecture for decisions under disrupted connectivity

Leila Rahman Mateo Silva

Astra Systems Laboratory
Illustrative Research Presentation

LetX Research Forum ▪ August 2026

The path from warning signs to trusted action

An alert is useful only if it arrives before conditions change

- Sensors observe rainfall, channel depth, and flow at different rates.

Decision gap

Raw measurements do not protect a neighborhood. A dependable pipeline must turn partial evidence into a bounded-risk action.

Core question

How much intelligence must remain local when cloud connectivity disappears?

An alert is useful only if it arrives before conditions change

- Sensors observe rainfall, channel depth, and flow at different rates.
- Central processing adds a network dependency exactly when links are least reliable.

Decision gap

Raw measurements do not protect a neighborhood. A dependable pipeline must turn partial evidence into a bounded-risk action.

Core question

How much intelligence must remain local when cloud connectivity disappears?

An alert is useful only if it arrives before conditions change

- Sensors observe rainfall, channel depth, and flow at different rates.
- Central processing adds a network dependency exactly when links are least reliable.
- A late or unstable alert erodes operator trust, even when the model is accurate.

Decision gap

Raw measurements do not protect a neighborhood. A dependable pipeline must turn partial evidence into a bounded-risk action.

Core question

How much intelligence must remain local when cloud connectivity disappears?

An alert is useful only if it arrives before conditions change

- Sensors observe rainfall, channel depth, and flow at different rates.
- Central processing adds a network dependency exactly when links are least reliable.
- A late or unstable alert erodes operator trust, even when the model is accurate.
- The design target is therefore timely, explainable, and graceful degradation.

Decision gap

Raw measurements do not protect a neighborhood. A dependable pipeline must turn partial evidence into a bounded-risk action.

Core question

How much intelligence must remain local when cloud connectivity disappears?

Three principles keep the system useful during disruption

1. Decide locally

Each gateway maintains a compact risk model and can issue a provisional alert without a round trip.

2. Fuse uncertainty

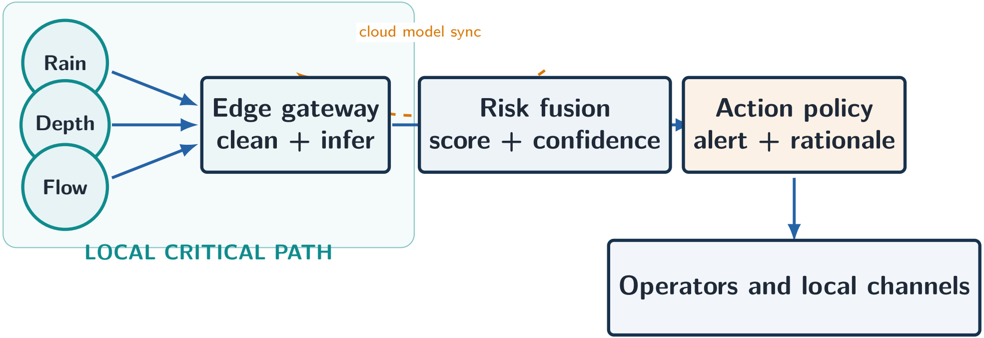
Missing sensors reduce confidence explicitly instead of silently changing the prediction.

3. Reconcile later

The cloud coordinates models and audit history, but it is not on the critical alert path.

Local autonomy shortens the path; explicit uncertainty keeps that autonomy accountable.

The critical path ends at the edge gateway



The cloud improves coordination; the gateway preserves continuity.

Risk combines hazard, confidence, and the cost of waiting

For gateway g at time t, the policy evaluates a bounded risk score

$$R_{g,t} = \sigma\left(\boldsymbol{w}^{\mathsf{T}} \boldsymbol{x}_{g,t} - \lambda u_{g,t} + \gamma \Delta_{g,t}\right), \qquad \sigma(z) = \frac{1}{1 + e^{-z}},$$

where

- $\boldsymbol{x}_{g,t}$: recent sensor features,
- $u_{g,t} \in [0, 1]$: uncertainty from missing or stale evidence,
- $\Delta_{g,t}$: estimated cost of deferring a decision,
- λ, γ: policy-controlled trade-offs.

Action rule

Issue a local alert when $R_{g,t} \geq \tau$ and confidence exceeds the operational floor; otherwise request corroboration or escalate to a human operator.

The evaluation stresses the failure modes that matter operationally

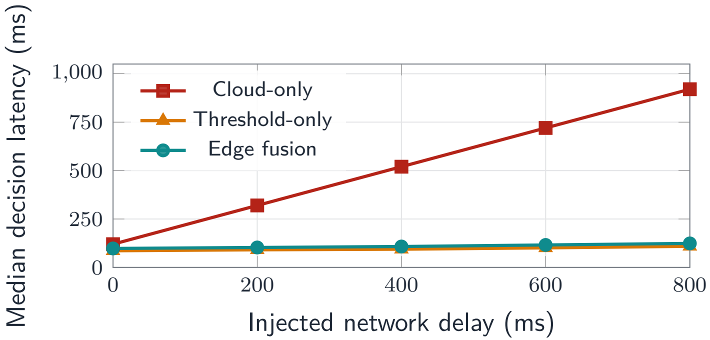
1. Replay 500 synthetic storm sequences across 24 virtual gateways.
2. Vary sensor loss, packet delay, and cloud availability independently.
3. Compare cloud-only, threshold-only, and edge-fusion policies.
4. Record decision latency, alert stability, and calibration error.

Stress envelope

Sensor dropout	0–40%
Link delay	20–800 ms
Cloud outage	0–30 min
Gateway budget	2 W

All values in this section are illustrative template data, not field-study claims.

Local inference keeps latency bounded as the network degrades



Key result

Edge fusion stays below **125 ms** across the full delay sweep.

Threshold rules are slightly faster, but the next test asks whether their alerts remain dependable.

Speed matters only when alerts remain stable and calibrated

Policy	Latency	Alert flips ↓	Calibration error ↓
Cloud-only	520 ms	4.8%	0.061
Threshold-only	**94 ms**	12.6%	0.142
Edge fusion	108 ms	**3.1%**	**0.038**

Interpretation

The edge policy gives up 14 ms relative to a simple threshold, then recovers substantially better stability and confidence calibration.

Illustrative medians at 400 ms injected delay; lower values are better.

Resilience improves, but deployment still requires guardrails

What the architecture handles well

It maintains a short decision path, exposes uncertainty, and records why each alert crossed the policy threshold.

What can still fail

Correlated sensor faults can create confident but wrong evidence; local models can also drift if reconnection is rare.

Operational guardrail

Combine plausibility checks, model-expiry limits, and a human-review path for ambiguous or unusually severe events.

A staged rollout earns trust before expanding autonomy

01

Shadow

Run beside the current process; compare recommendations without issuing alerts.

02

Supervised

Let operators approve alerts while monitoring calibration and disagreement.

03

Bounded autonomy

Automate only verified scenarios; escalate low-confidence cases by policy.

Promotion gate

Advance only when latency, calibration, false-alert burden, and operator acceptance meet predefined thresholds together.

Reliable alerts come from architecture, not accuracy alone

1. Keep the critical decision path local and bounded.
2. Represent missing evidence as uncertainty, not as silence.
3. Use the cloud for coordination and learning, not mandatory availability.
4. Expand autonomy gradually, with measurable promotion gates.

Takeaway

A resilient edge system does more than predict risk: it preserves a trustworthy decision process when infrastructure is under stress.

Questions and discussion ▪ research@astra.example