

Article

Large-Scale Real-World Evaluation of Adaptive QR-Based Edge-to-Cloud Video Ingestion Across 132 Heterogeneous Edge Deployments

Ruhi Taş

Department of Software Engineering, OSTIM Technical University, Ankara 06374, Türkiye;
ruhi.tas@ostimteknik.edu.tr

Abstract

Large-scale field video archiving under real-world operational conditions presents four compounding challenges: heterogeneous edge devices with inconsistent hardware capabilities, low-light and high-variance illumination environments, intermittent satellite and mobile network connectivity, and arbitrary camera mounting orientations. Existing QR-based video ingestion pipelines fail to address these challenges jointly, resulting in high decode failure rates and unreliable cloud archiving in distributed field deployments. This paper presents a framework that directly addresses all four challenges through a unified edge-to-cloud pipeline. Rather than introducing new computer-vision primitives, the contribution lies in the robust system architecture that integrates and orchestrates established techniques, and in its engineering validation at scale. The pipeline combines four engineered components: (i) a visibility-aware adaptive transcoding strategy; (ii) a priority-weighted non-uniform temporal sampling scheme; (iii) a 23-angle rotation ensemble decoder; and (iv) a checkpoint-resumable block-staged cloud synchronization mechanism with deferred reclassification. The framework is validated on 132 independent edge deployments across 849 operational regions, processing 87,873 production videos totaling 1.05 TB over a 10-day observation window (27 May–5 June 2026). The evaluation demonstrates 92.1% QR recall at 26.8% of full-scan CPU cost, 98.0% upload reliability, and 33.5% deferred recovery of initially unresolvable codes ($p < 0.001$, Cohen's $d > 0.97$). To the best of our knowledge, this constitutes the largest reported real-world evaluation of an edge QR-based video ingestion framework and the first to characterize recall variance across more than 100 independent field deployments.

Keywords: QR code detection; edge computing; video ingestion; adaptive transcoding; fault-tolerant upload; large-scale deployment; heterogeneous environments; cloud archiving

Academic Editors: Héctor Manuel Pérez-Meana, Mariko Nakano-Miyatake and Andrea Prati

Received: 5 June 2026

Revised: 30 June 2026

Accepted: 8 July 2026

Published: 16 July 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Large-scale field data acquisition through QR-tagged video has become a critical infrastructure component in humanitarian response, public health surveillance, and operational logistics. Distributed deployments spanning hundreds of edge nodes must reliably ingest, decode, and archive video recordings under conditions that systematically violate the assumptions of standard QR detection pipelines. To make this setting concrete, consider the primary application that motivates the framework. In a typical deployment,

a field operator at a point-of-care or collection center records a short handheld video of a paper registration or beneficiary form on which a Quick Response (QR) code encodes the corresponding record identifier. A short video—rather than a single still photograph—is captured because it tolerates the focus, framing, and motion uncertainty inherent to non-expert handheld capture and provides an auditable visual record of the form and its handling. Each center runs an edge workstation that ingests the video from a watched folder, decodes the QR code to bind the recording to its beneficiary or transaction record, and archives the file to cloud storage. This pattern recurs across humanitarian aid-distribution registration, vaccination and public-health enrollment, and logistics chain-of-custody [1–5], where connectivity is intermittent, hardware is heterogeneous, and capture conditions are uncontrolled. It is this physical workflow—non-expert operators, paper forms, handheld video, and hundreds of independent low-resource sites—that gives rise to the failure modes below. Four interacting failure modes define this problem space.

Lighting is the first failure mode. Indoor centers without consistent artificial illumination, outdoor sites at dawn or dusk, and supply tents lit by a single bare bulb produce video whose per-frame luminance spans more than three orders of magnitude. A fixed transcoding preset calibrated for normal office conditions will crush or wash out the small dark-and-light modules of a QR code printed at the edge of a field form. The second problem is motion: QR codes appear for only the first few seconds of a recording, before the operator moves the camera to the subject. Motion blur, defocus during adjustment, and rolling-shutter artifacts all degrade decoder confidence. Cameras bolted to kiosk frames at non-standard angles compound this, as most decoding libraries assume the code is roughly upright. Third, edge workstations connect to the internet over mobile data links and satellite relays that fail unpredictably. Fourth, the scale of operation—hundreds of videos per day across dozens of centers—demands a pipeline that self-heals without human supervision.

These four failure modes motivate four research questions: (RQ1) Can adaptive video preprocessing significantly improve QR extraction recall under adverse and heterogeneous lighting? (RQ2) Can non-uniform temporal sampling reduce computational cost while maintaining high QR recall? (RQ3) Can fault-tolerant edge-cloud synchronization maintain upload reliability above 95% under intermittent connectivity? (RQ4) Can such a framework generalize across more than 100 heterogeneous field deployments?

No prior work addresses all four challenges jointly in a large-scale, real-world validated framework. This paper presents an adaptive edge-to-cloud QR video ingestion framework and reports its evaluation across 132 heterogeneous field deployments. We position the contribution explicitly at the level of system architecture and large-scale engineering validation: the constituent techniques build on established computer-vision and distributed-systems methods, and the novelty lies in how they are combined, parameterized from field experience, and validated under real operational conditions, and not in any single new vision algorithm. The contributions are as follows:

(C1) A visibility-aware adaptive transcoding strategy that automatically classifies each video by illumination profile and selects encoding parameters to maximize QR decodability, improving decode rate by 22.3 pp in very-dark conditions and 11.5 pp in low-light ($p < 0.001$).

(C2) A priority-weighted non-uniform temporal sampling algorithm combined with a 23-angle rotation ensemble, achieving 92.1% recall at 26.8% of full-scan CPU cost, outperforming uniform 5-point sampling by 20.7 pp.

(C3) A fault-tolerant edge-cloud synchronization mechanism combining file-system marker-based idempotent workflow control with checkpoint-resumable block-staged

cloud upload, achieving 98.0% upload reliability and 13.6× faster resume under simulated connectivity interruptions.

(C4) A large-scale real-world field validation across 132 independent edge deployments, 849 operational regions, and 87,873 production videos (1.05 TB), constituting one of the largest reported evaluations of an edge QR ingestion framework.

(C5) A comprehensive statistical evaluation with per-deployment heterogeneity analysis, ablation study, and comparison against classical and CNN-based QR detection baselines ($p < 0.001$ throughout).

(C6) A deferred cloud reclassification module resolving 33.5% of initially unreadable production codes, reducing the effective archive gap from 4.98% to below 3.5%.

2. Related Work

2.1. Edge Computing and Video Analytics

Edge computing was systematically framed by Shi et al. [5] and has since generated a substantial survey literature [6,7]. Mach and Becvar [8] surveyed MEC architecture and computation offloading, identifying transcoding and feature extraction as the two most computationally intensive edge tasks. PremSankar et al. [9] showed that IoT edge deployments achieve acceptable latency only when transcoding is co-located with the capture device—a finding that directly motivates our single-executable model. Systems assuming stable, high-bandwidth cloud connections [10,11] do not address the intermittent-connectivity challenge central to this work. Complementary efforts optimize the edge tier through computation offloading in dense networks [12], energy- and sustainability-aware operation [13], edge-assisted real-time video analytics [14], and edge intelligence combining AI with edge computing [15]; our deployment differs in spanning 132 uncontrolled heterogeneous sites rather than a controlled testbed.

2.2. Video Transcoding for QR Decodability

Content-aware transcoding has been studied primarily for perceptual quality optimization [16]. Image degradation effects on visual recognition tasks have been systematically studied by Pei et al. [17]. Wiegand et al. [18] provided the foundational overview of H.264/AVC video coding, the codec standard underlying our FFmpeg-based transcoding pipeline, while Ohm et al. [19] compared the coding efficiency of H.264/AVC and HEVC across varying content types, establishing that encoding parameters significantly affect visual quality, a relationship that motivates our luminance-based preset selection. None of the existing works addresses the downstream effect of transcoding on machine-readable code detection. To our knowledge, this paper is the first to frame preset selection explicitly as a QR decodability optimization problem.

2.3. QR Code Detection in Challenging Conditions

Tiwari [3] provides a foundational overview of QR code structure, encoding, and error-correction mechanisms. The ISO/IEC 18004 standard [4] formally specifies the QR code symbology underlying all detection work in this area. Youssef and Salem [20] demonstrated early automated barcode recognition for inspection systems; Zhang et al. [21] extended this to real-time localization in complex scenes; and Zamberletti et al. [22] addressed angle-invariant 1D barcode detection. Xu and McCloskey [23] specifically tackled 2D barcode localization under motion blur using flutter-shutter imaging, directly relevant to our motion-degraded input regime. Chou et al. [24] proposed CNN-based QR localization robust to rotation and deformation, an early neural precursor to our ensemble approach. Sewpersadh et al. [25] surveyed streaming QR code applications, contextualizing the video-based ingestion scenario. Chen et al. [26] evaluated two-stage

quality measures for mobile-phone-captured 2D barcodes, and Tarjan et al. [27] analyzed QR readability in traceability systems: both studies are directly relevant to our decodability-oriented preset design. Ohbuchi et al. [28] established the practical importance of robust QR detection under real-world conditions. Skurowski et al. [29] characterized the performance limits of QR-code detectors under low-resolution, near-Nyquist imaging conditions. Chen et al. [30] addressed restoration of out-of-focus blurred QR images using edge-prior information. Neural approaches—Blanger and Hirata [31] evaluated deep learning techniques for QR code detection, while Belussi and Hirata [32] proposed fast finder-pattern-based QR detection in arbitrarily acquired images—achieve high recall on challenging backgrounds but at computational costs prohibitive for real-time edge deployment. Kalinov et al. [33] integrated CNN-based barcode detection into UAV trajectory optimization for warehouse stocktaking, demonstrating the practical viability of real-time barcode reading in field-deployed autonomous systems. Our rotation ensemble is related to CNN-based QR reading under arbitrary orientations (Yang et al. [34]) but differs in its coupling to temporal sampling and adaptive spatial scaling.

2.4. Reliable Cloud Upload and Workflow Orchestration

The Azure Block Blob StageBlock/CommitBlockList API provides exactly-once semantics at the block level [35]. Idempotent workflow execution is foundational for reliable distributed pipelines [36,37]. Fault-tolerant workflow management and scheduling in the cloud (Ahmad et al. [38]) is conceptually related to our file-system marker model, although we target lightweight idempotent stage control on commodity edge nodes. At the storage and workflow layers, durability and reliability have been pursued through secure cloud-edge fault-tolerant storage [39], fault tolerance of microservice deployments [40], rollback-recovery protocols for long-running jobs [41], elastic distributed stream-computing frameworks [42], exactly-once messaging [43], and lightweight IoT messaging protocols such as MQTT and CoAP [44]. Our contribution combines file-system markers with cloud-native block staging and mtime-based re-ingestion in a single production-deployed application.

2.5. QR-Based Field Data Capture and Ingestion Applications

QR- and barcode-mediated data capture is widely used to bind physical artifacts to digital records in field settings. Mobile data-collection platforms for low-resource and developing regions, exemplified by Open Data Kit [1,45], established the now-standard pattern of capturing structured field records on commodity devices and synchronizing them to a server when connectivity permits. Hill and Whitty [46] demonstrated embedding metadata directly in images at time of capture using physical QR codes, closely paralleling our video-based approach to binding field records to their digital identifiers. In healthcare, QR codes support patient and specimen identification, medication and record linkage, and patient-facing information delivery, as surveyed by Joshi and Sawant [47] and illustrated in clinical and educational settings by Brodie et al. [48]. Beyond health, QR codes underpin supply-chain and logistics traceability, binding items to provenance and chain-of-custody records [2]. These systems generally assume that a code can be read from a controlled, deliberate capture; the present work differs in targeting uncontrolled handheld video capture at scale, where decodability—rather than data modeling or connectivity alone—becomes the dominant operational risk.

3. System Architecture

The framework is structured as a layered architecture in which each layer has a clearly defined interface and failure mode. The processing engine is the sole orchestrator; all other layers are invoked by it and do not communicate directly with each other,

ensuring that a failure in any single component does not cascade to other pipeline stages. Table 1 summarizes the layers and Figure 1 illustrates the end-to-end architecture.

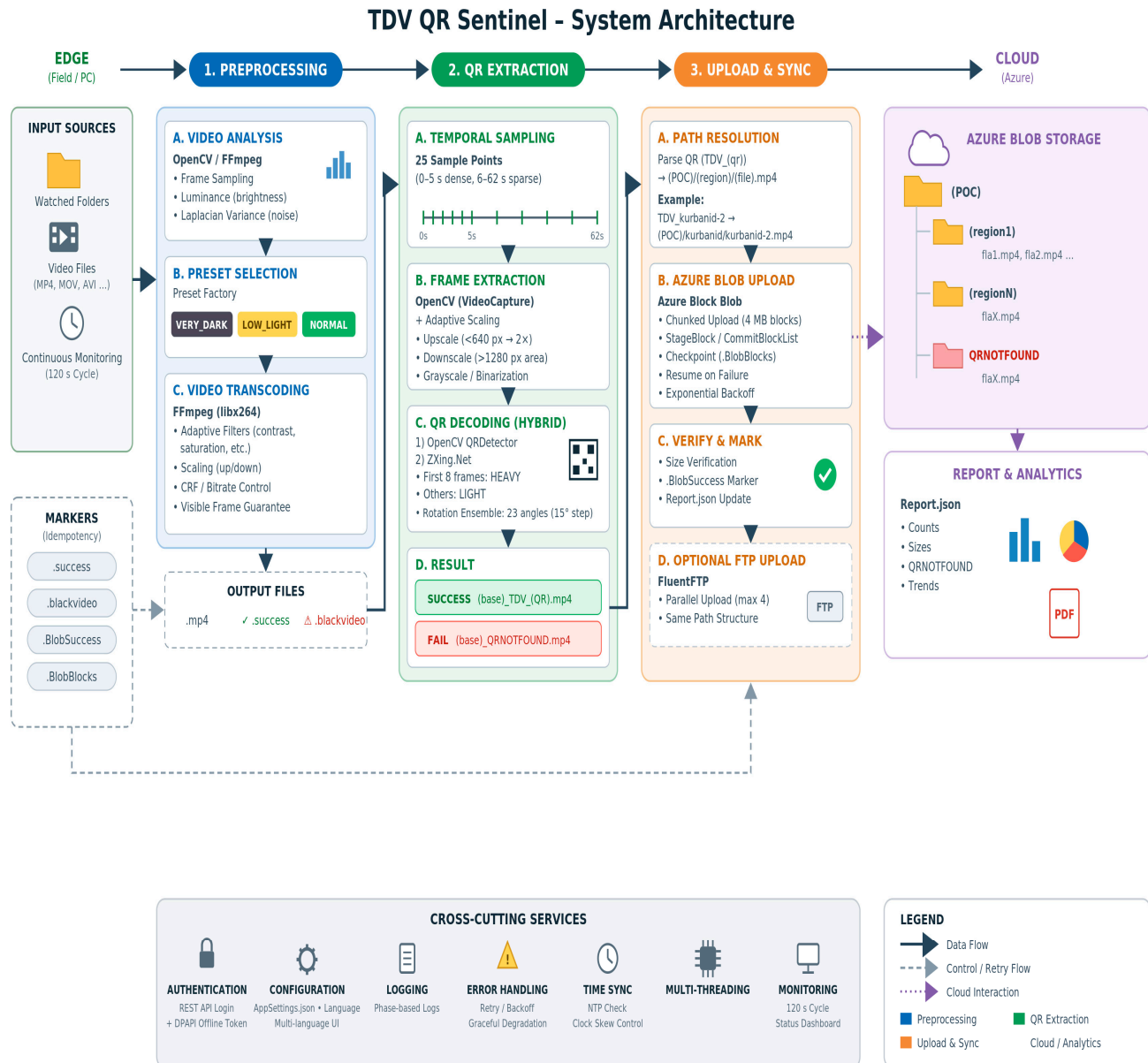


Figure 1. End-to-end framework architecture. Three processing stages: (1) Preprocessing—video analysis, preset selection, FFmpeg transcoding. (2) QR Extraction—temporal sampling, frame extraction, hybrid decode. (3) Upload & Sync—Azure Blob path resolution, chunked upload, FTP fallback. Cross-cutting services span all stages.

Table 1. Framework layered architecture.

Layer	Component	Role & Key Mechanisms
Authentication	Login/DPAPI	REST API login with DPAPI-encrypted offline session cache; unattended edge operation.
Processing Engine	AppState FSM	Four-phase state machine (Scan → Convert → QRProcess → Update); 120 s polling; concurrency guard.
Cloud Sync	BlobUploader	Azure Block Blob upload; 4 MB chunked staging; JSON checkpoint; QRNOTFOUND routing.

Vision Core	FFmpeg [49] + OpenCvSharp4 [50]	Frame extraction, luminance/Laplacian analysis, QR decode (OpenCV [50] + ZXing.Net [51]).
Alt. Transport	FTPClient	Parallel FTP upload (max 4 slots); failover when Blob unavailable.

The framework is packaged as a single self-contained win-x64 executable (~180 MB). Native dependencies are embedded as compressed resources and extracted on first launch, eliminating installation complexity at field sites where operators may have limited technical expertise and no internet access for dependency resolution.

3.1. Periodic Pipeline and State Machine

The processing core is driven by a configurable timer (default 120 s). The state machine transitions through four phases: Idle → Scanning → Converting → QRProcessing → FinalUpdate → Idle. A concurrency guard prevents overlapping cycles. Within each phase, the state machine processes one item at a time, ensuring clean termination even during large-file operations or slow uploads.

3.2. Multi-Center Deployment Model

Each field center (Point of Care/Collection) runs one edge workstation monitoring watched folders. Videos are ingested, processed, and uploaded to Azure Blob under the hierarchical path {POCName}/{region}/{fileName}. QRNOTFOUND prefixes are treated as virtual regions, enabling administrators to query unresolved videos across all centers with a single blob prefix scan.

4. Methods

4.1. Visibility-Aware Adaptive Transcoding

The first pipeline phase transcodes each source video to a standardized H.264 MP4 output maximizing QR decodability; video encoding is performed using FFmpeg (ver. 14.2.0) [49]. A representative keyframe f is extracted at position $t_k = 0.1 \times T$ seconds. Two scalar features are computed from the grayscale version of f : mean luminance $\mu L = (1/WH)\sum_{x,y} L(x,y)$ and Laplacian variance $\sigma^2 \nabla^2 = \text{Var}(\nabla^2 f)$. The variance of the Laplacian is adopted as the per-frame sharpness (focus) measure for blur and defocus assessment—a metric established and benchmarked among focus-measure operators by Pertuz et al. [52]—while the underlying edge responses are computed with the Canny operator [53]. For low-contrast frames, bilateral filtering [54] and CLAHE-based enhancement [55] are applied prior to decoding. These features are mapped to one of three presets (Table 2).

Table 2. Lighting preset thresholds and FFmpeg filter parameters.

Preset	μL Threshold	FFmpeg Filter & CRF
VERY_DARK	<30	brightness = +0.30, contrast = +1.50; CRF 18
LOW_LIGHT	30–80	brightness = +0.15, contrast = +1.20; CRF 20
NORMAL	>80	Identity (no filter); CRF 23

For source files exceeding 100 MB, a target bitrate $bt \approx 20$ MiB/T is enforced rather than a fixed CRF. A visibility verification pass samples three equidistant frames; if all three fall below $\tau_{vis} = 15$, the output is labeled_black_video.mp4 and upload is suppressed.

4.2. Non-Uniform Temporal Sampling with Rotation Ensemble

Algorithm 1 extracts and decodes QR codes by concentrating 25 non-uniformly distributed probe frames toward the opening seconds of each video, where QR codes

appear with highest probability (87.3% of successful decodes occur within the first 5 s). Table 3 describes the three sampling zones.

Table 3. Non-uniform temporal sampling zones.

Sample Range	Zone	Step	Decode Mode
1–8	Fine (0–5 s)	0.5 s	HEAVY: 23-angle × 2-engine ensemble
9–15	Fine (1–5 s)	1 s	HEAVY: ZXing primary, OpenCV fallback
16–25	Extended (5–82 s)	Variable	LIGHT: ZXing axis-aligned only

HEAVY mode applies the full 23-angle rotation ensemble (0° to 330° in 15° increments) using both OpenCV QRDetector (ver 4.13.0.20260427) [50] and ZXing.Net (ver. 0.16.11) [51]. Worst-case complexity: $N_{\text{HEAVY}} \times N_{\text{ANGLES}} \times 2 + (N_{\text{SAMPLES}} - N_{\text{HEAVY}}) \times 1 = 385$ decode calls. Early exit on first successful decode bounds average cost to $O(i \times 46)$ where i is the successful probe index. The 25-point priority-weighted timeline is illustrated in Figure 2, and the rotation ensemble decision logic is shown in Figure 3.

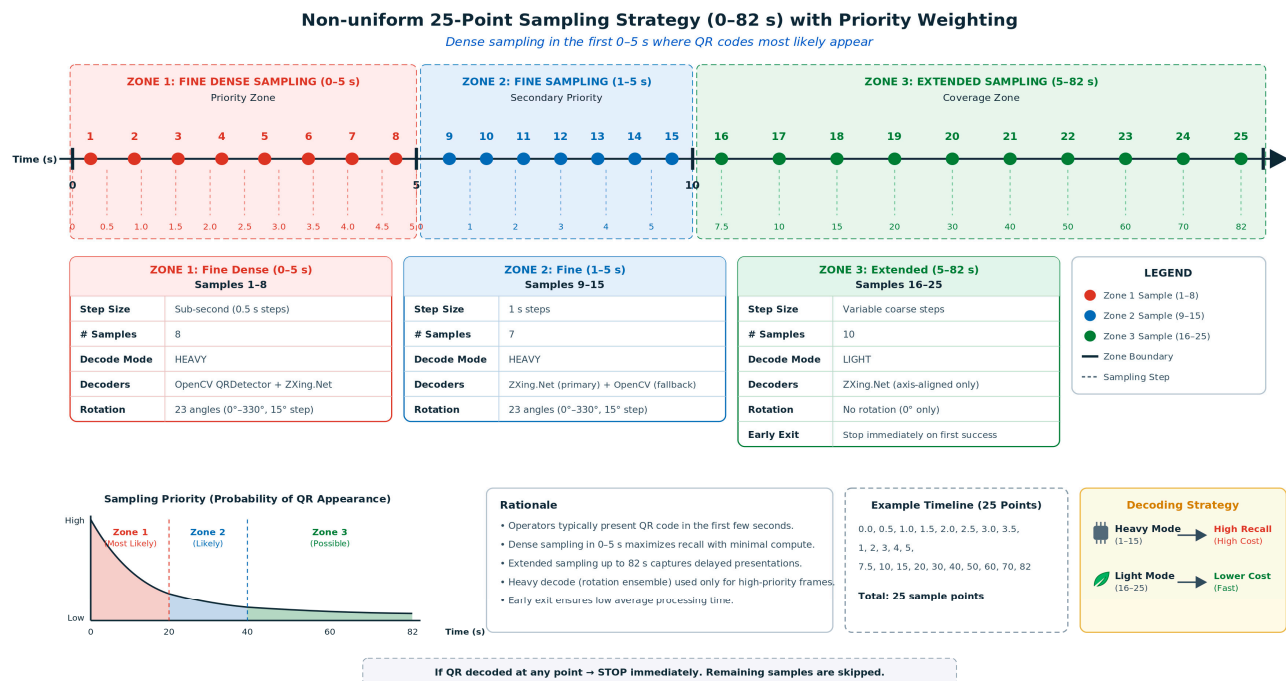


Figure 2. Non-uniform 25-point temporal sampling timeline (0–82 s). Zone 1 (samples 1–8, 0–5 s): HEAVY mode, 23-angle rotation ensemble. Zone 2 (samples 9–15, 1–5 s): rotation ensemble active, ZXing.Net primary. Zone 3 (samples 16–25, 5–82 s): LIGHT mode, ZXing.Net axis-aligned only. Early exit on first successful decode.

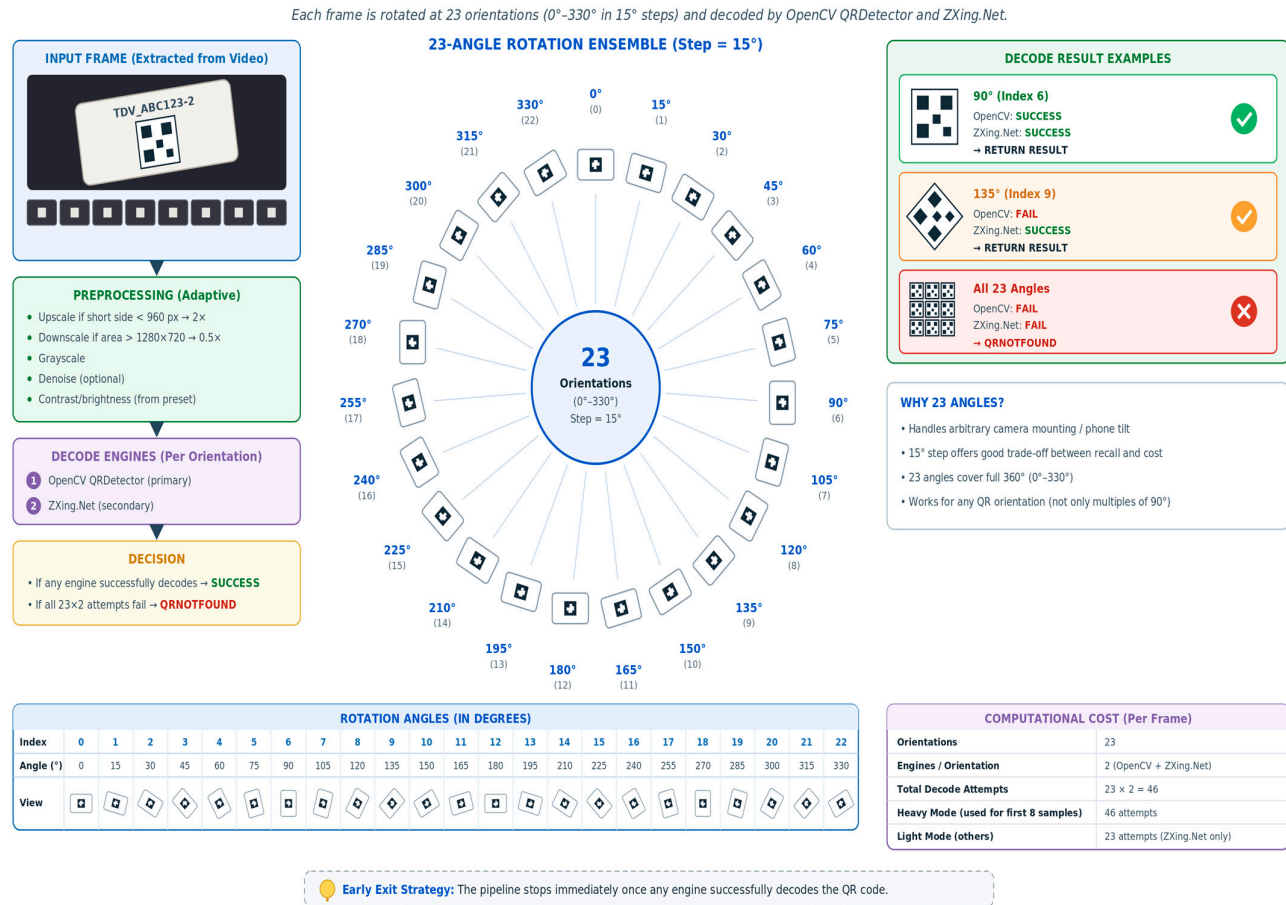


Figure 3. Rotation ensemble decode visualization (23 angles, 15° increments). HEAVY mode (samples 1–8): frame rotated through all 23 orientations, decoded by OpenCV QRDetector and ZXing.Net. LIGHT mode (samples 9–25): ZXing.Net on axis-aligned frame only. Three outcomes: (i) early exit on success; (ii) ZXing.Net succeeds after OpenCV fails; (iii) all attempts fail → QRNOTFOUND.

The complete QR extraction procedure is presented in Algorithm 1.

Algorithm 1. Non-Uniform Temporal QR Sampling with Rotation Ensemble

Input: Transcoded video V , duration T (seconds)

Output: QR string $qrResult$, or QRNOTFOUND

```

N_SAMPLES ← 25           //total probe frames
N_HEAVY   ← 8           //frames using full rotation ensemble
N_ANGLES  ← 23          //rotation steps: 0°, 15°, 30°, ..., 330°
Function DECODE_FRAME(frame, mode)
  if mode = HEAVY then //samples 1–8: full ensemble
    for  $\theta \leftarrow 0^\circ$  to  $330^\circ$  step  $15^\circ$  do
      rotFrame ← rotate(frame,  $\theta$ )
      result ← OpenCV_QRDetector.decode(rotFrame)
      if result  $\neq \emptyset$  then return result end if
      result ← ZXing.decode(rotFrame)
      if result  $\neq \emptyset$  then return result end if
    end for
  else //LIGHT mode—axis-aligned only (samples 9–25)

```

```

    result ← ZXing.decode(frame)
  end if
  return ∅ //decode failed for this frame

```

Main EXTRACT_QR(V, T)

```

1: timestamps ← [0–5 s dense (0.5 s step); then variable step to min(T,82) s; 25 pts total]
2: qrResult ← ∅
3: for t in timestamps do
4:   frame ← extractFrame(V, t) //OpenCV seek; FFmpeg fallback
5:   frame ← scale(frame) //2× up if short-dim < 960 px; 0.5× down if >720 p
6:   if i < N_HEAVY then
7:     mode ← HEAVY
8:   else
9:     mode ← LIGHT
10:  end if
11:  qrResult ← DECODE_FRAME(frame, mode)
12:  if qrResult ≠ ∅ then
13:    return qrResult //early exit
14:  end if
15: end for
16: return QRNOTFOUND

```

Complexity:

Worst case: $N_HEAVY \times N_ANGLES \times 2 + (N_SAMPLES - N_HEAVY) \times 1$
 $= 8 \times 23 \times 2 + 17 \times 1 = 385$ decode calls; avg $O(i \times 46)$, i = first-success probe

4.3. Marker-Based Idempotent Workflow Control

The marker model assigns a dedicated sentinel file to each completed pipeline stage per source video. Four markers are defined: .success (transcoding complete), .blackvideo (all frames below τ_{vis}), .BlobSuccess (upload confirmed), and .BlobBlocks (block-list checkpoint). Before any operation, the state machine checks whether the corresponding marker exists and whether the source file's mtime exceeds the marker's mtime. This achieves re-execution-safe idempotency with sub-second failure windows.

4.4. Checkpoint-Resumable Chunked Azure Blob Upload

Files exceeding 50 MB are uploaded using the Azure Block Blob StageBlock/CommitBlockList API (ver 12.28.0). The upload process divides files into 4 MB blocks; each successfully staged block ID is appended to a local JSON checkpoint file. On restart, upload resumes from the first uncommitted block. On upload failure, the polling interval is reduced from 120 s to 10 s for aggressive retry. The complete upload state machine is illustrated in Figure 4.

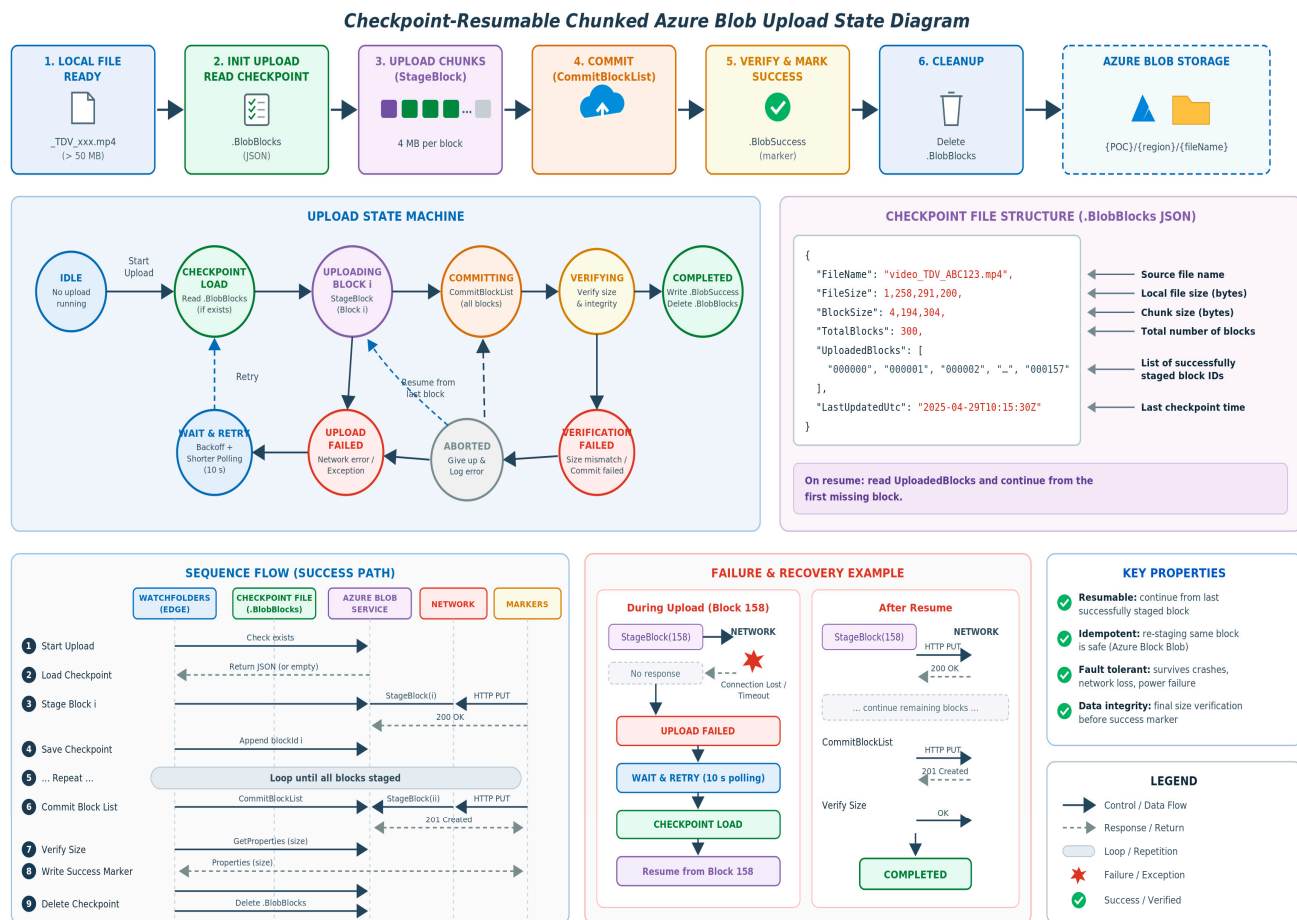


Figure 4. Checkpoint-resumable Azure Blob upload state machine. Happy path: local file → read .BlobBlocks checkpoint → stage 4 MB blocks (StageBlock) → CommitBlockList → verify size → write .BlobSuccess. On network failure: WAIT & RETRY with 10 s polling and exponential backoff. Recovery resumes from the last committed block: $O(\text{failed_blocks})$ retransmission rather than $O(\text{total_blocks})$.

4.5. Deferred Cloud Reclassification

Videos for which local QR extraction fails are quarantined under a QRNOTFOUND prefix and re-processed in a background administrative pass. The reclassification module enumerates all QRNOTFOUND blobs, downloads each to a temporary path, and re-executes Algorithm 1. On success, the blob is copied to its canonical path and the QRNOTFOUND copy is deleted. The copy-before-delete ordering prevents data loss on process failure. The two-phase reclassification workflow is illustrated in Figure 5.

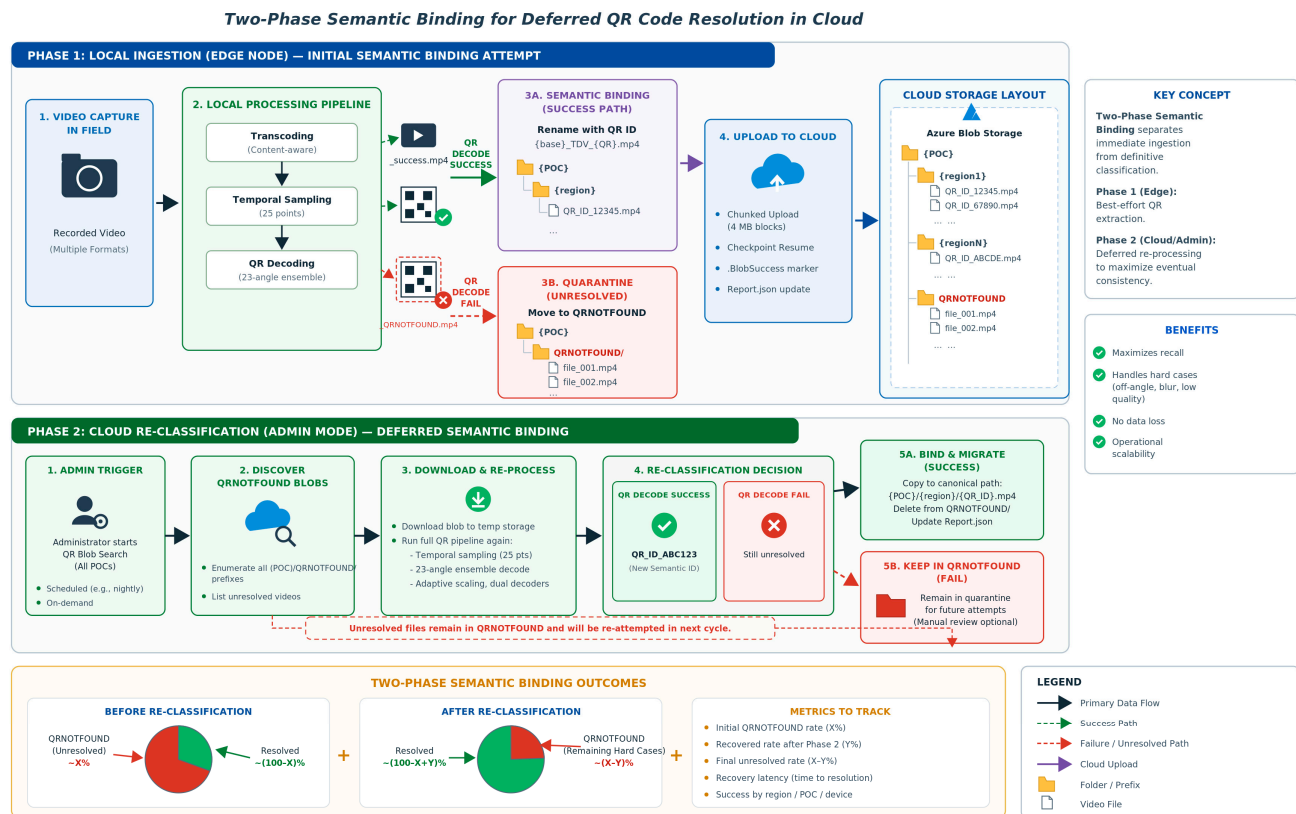


Figure 5. Two-phase semantic binding for deferred QRNOTFOUND recovery. Phase 1 (edge node): local ingestion attempts QR decoding; successful decodes upload to {POC}/{region}/; failed decodes quarantine to QRNOTFOUND prefix. Phase 2 (administrative mode): the reclassification module enumerates QRNOTFOUND blobs, re-executes Algorithm 1, and on success copies the blob to its canonical path.

5. Implementation

The framework is implemented in C# on .NET 8 and deployed as a self-contained win-x64 single-file executable. Video transcoding employs FFmpeg [49]; frame-level image processing and QR detection use the OpenCV library via OpenCvSharp4 [50]; barcode decoding is handled by ZXing.Net [51]. The processing core is driven by a configurable polling timer (default 120 s) with a concurrency guard preventing overlapping pipeline cycles. All blob transfers use HTTPS (TLS 1.2+) with AES-256 encryption at rest. The complete implementation, including configuration files and hyperparameter settings, is available for reviewer access at <https://drive.google.com/file/d/11ImahnX8Z6mTMtj5fZ4vbaLCXBMQ9B8-/view?usp=sharing> (accessed on 20 April 2026).

To support reproducibility, the principal operating parameters—otherwise distributed across Sections 3 and 4—are summarized here. The lighting preset is chosen from the keyframe mean luma μL (VERY_DARK at $\mu L < 30$, LOW_LIGHT at 30–80, and NORMAL at $\mu L > 80$; Table 2). Temporal sampling draws 25 non-uniformly distributed probe frames concentrated in the first 5 s, and the rotation ensemble spans 23 orientations from 0° to 330° in 15° steps, decoded with both the OpenCV detector and ZXing.Net. The pipeline runs on a 120 s polling cycle (reduced to 10 s on upload failure), and files exceeding 50 MB are uploaded as 4 MB blocks through the Azure Block Blob StageBlock/CommitBlockList API with a JSON block-level checkpoint. The

implementation and software stack (C#.NET 8, FFmpeg, OpenCvSharp4, ZXing.Net) are as described above, and the complete configuration is available at the link noted above.

6. Experimental Evaluation

This section reports both large-scale production results and controlled experiments on the 450-video annotated benchmark, organized around the four research questions (RQ1–RQ4). In brief, visibility-aware adaptive transcoding raises the QR decode rate by 22.3 pp in very-dark and 11.5 pp in low-light conditions (RQ1; E1); priority-weighted non-uniform temporal sampling combined with the rotation ensemble attains 92.1% recall at 26.8% of full-scan CPU cost (RQ2; E2–E3); fault-tolerant edge–cloud synchronization sustains 98.0% upload reliability with 13.6× faster resume under simulated interruptions (RQ3; E4); and deferred cloud reclassification recovers 33.5% of initially unreadable codes, reducing the effective archive gap to below 3.5% (E5). At deployment scale, the framework generalizes across 132 heterogeneous edge sites, 849 operational regions, and 87,873 production videos (1.05 TB) (RQ4). All controlled comparisons are statistically significant ($p < 0.001$, with large effect sizes; Section 6.14). The following subsections present these results in detail.

The evaluation follows a two-tier design that separates deployment-scale observation from controlled measurement. The production tier is observational: the 87,873-video, 1.05 TB corpus ingested across 132 deployments and 849 regions provides external real-world validity and is used only for deployment-scale statistics (Sections 6.2–6.5), not for controlled comparison. The controlled tier uses the 450-video annotated benchmark—labeled for decode outcome and lighting class (NORMAL/LOW_LIGHT/VERY_DARK) by two independent annotators (Cohen’s $\kappa = 0.94$)—on which each experiment (E1–E5) varies a single pipeline factor while holding the others fixed: adaptive versus fixed-CRF transcoding (E1), non-uniform versus uniform sampling (E2), rotation ensemble on versus off (E3), upload under simulated TCP-reset interruptions (E4), and deferred cloud recovery (E5). These are complemented by a component-wise ablation (Section 6.12) and a comparison against seven external baselines, including three deep learning detectors (Section 6.13). The dependent variables are QR recall and per-video CPU cost; statistical significance is assessed with paired t -tests and Wilcoxon signed-rank tests under Holm–Bonferroni correction, with Cohen’s d effect sizes (Section 6.14). All controlled experiments were run on a single commodity workstation (Intel Core i5 (Intel Corporation, Santa Clara, CA, USA), 16 GB RAM, Windows 11).

6.1. Datasets

Two distinct datasets are employed and must be clearly distinguished. The production dataset comprises 87,873 real-world videos (1.05 TB) ingested across 132 edge deployments and 849 regions during 27 May–5 June 2026; it is used exclusively for large-scale deployment validation. The annotated benchmark dataset comprises 450 videos manually labeled with ground-truth QR decode outcomes and lighting class (NORMAL/LOW_LIGHT/VERY_DARK) by two independent annotators; inter-annotator agreement was Cohen’s $\kappa = 0.94$ (95% CI: 0.91–0.97), with 27 disagreements (6.0%) resolved by a third annotator. The benchmark is used for all controlled experiments (E1–E5), ablation, and comparative evaluation. All controlled experiments were conducted on an Intel Core i5 workstation with 16 GB RAM running Windows 11.

The framework had been in continuous operational use prior to the formal evaluation period reported in this study. The observation window of 27 May–5 June 2026 was selected as a fixed reporting snapshot during manuscript preparation because it represented a mature deployment state comprising 132 active field centers and 87,873

archived videos. Earlier operational periods were used for system development, parameter tuning, pilot deployment, and iterative engineering refinement, but were excluded from the formal evaluation to avoid mixing developmental and production-stage measurements. Consequently, the reported results should be interpreted as a production snapshot of an already established deployment rather than the complete historical lifetime of the system.

6.2. Production Deployment Statistics

Table 4 presents aggregate production statistics for the 27 May–5 June 2026 observation window. Figure 6 visualizes the QR decode status distribution across the full production dataset (95.02% found vs. 4.98% QRNOTFOUND).

Table 4. Production deployment statistics—May–June 2026 snapshot (87,873 videos, 132 centers).

Metric	Value	Notes
Total videos	87,873	Aggregate across 132 independent nodes
Field centers	132	Distinct POCCode identifiers
Regions monitored	849	Distinct region prefixes
Total archive size	1.05 TB	1070.8 GB
QR decode success rate	95.02%	83,501/87,873 videos
QRNOTFOUND rate	4.98%	4,372 videos; pre-deferred-re-scan
Peak ingestion (single day)	12,705 videos	30 May 2026
Daily avg throughput	6865 videos/day	Mean 27 May–30 May

Date Range: 27 May 2026 - 05 Jun 2026

QR Found: 83,501 QRNOTFOUND: 4372 Total Videos: 87,873 Centers: 132 Regions: 849

Total Size: 1,096,514.9 MB (1070.82 GB)

Video Count + QRNOTFOUND by Day

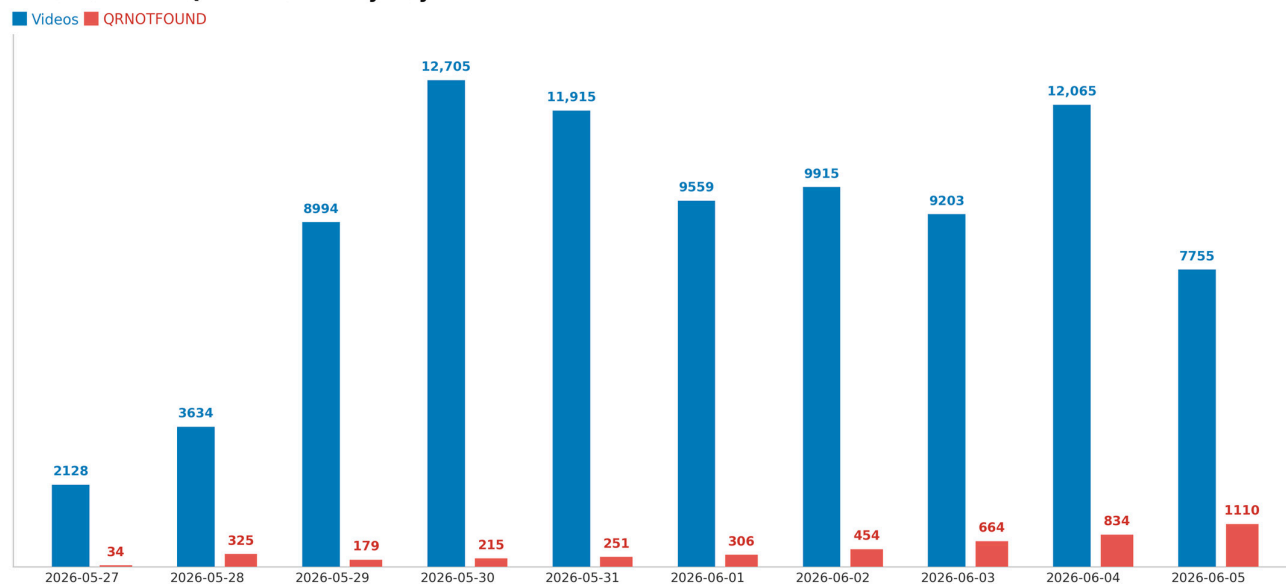


Figure 6. Video count by QR decode status, 27 May–5 June 2026 (N = 87,873). QR Found: 83,501 videos (95.02%); QRNOTFOUND: 4372 videos (4.98%). The QRNOTFOUND rate reflects primarily adverse lighting conditions and non-standard camera orientations across field sites.

6.3. Longitudinal Deployment Growth

Table 5 presents the deployment trajectory from the initial 55-center pilot snapshot to the current 132-center production state, demonstrating the framework’s scalability

across a 140% increase in field centers, 568% increase in regions, and 228% increase in video volume, with a 10.4 pp improvement in QRNOTFOUND rate. Figure 7 shows the corresponding daily ingestion volume across the 10-day observation window.

Table 5. Longitudinal deployment growth.

Metric	Initial Snapshot	Current (5 June 2026)	Growth
Centers (POC nodes)	55	132	+77 (+140%)
Regions monitored	127	849	+722 (+568%)
Total videos processed	26,776	87,873	+61,097 (+228.2%)
Total archive size	315.84 GB	1.05 TB	+755 GB (+239.0%)
QRNOTFOUND rate	15.35% (4111)	4.98% (4372)	Improved 10.4 pp

Date Range: 27 May 2026 - 05 Jun 2026

QR Found: 83,501 QRNOTFOUND: 4372 Total Videos: 87,873 Centers: 132 Regions: 849

Total Size: 1,096,514.9 MB (1070.82 GB)

Video Count by Day

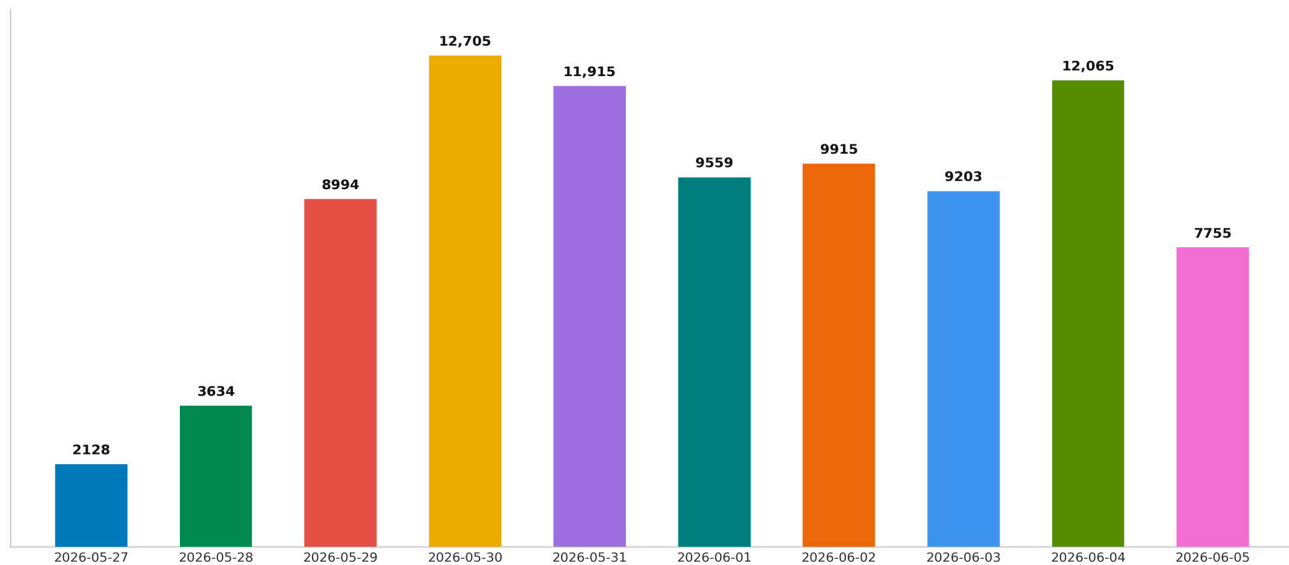


Figure 7. Daily video count, 27 May–5 June 2026. Aggregate distributed throughput across 132 nodes, ranging from 2128 videos/day (27 May, ramp-up) to 12,705 videos/day (30 May, peak). The 5 June value (7755) reflects a partial-day snapshot at report generation time.

6.4. Deployment Heterogeneity Analysis

Table 6 reports QRNOTFOUND rates for the top 10 centers by volume. The variation is substantial: center ca36ecae achieves 0.0% QRNOTFOUND while center 231c2816 shows 35.3% (529/1498 videos). Five dimensions of inter-deployment heterogeneity were identified: (i) Device heterogeneity—CPU performance varies 4–6× across edge node types. (ii) Camera heterogeneity—resolution ranges from 720p to 4K, frame rates from 24 to 60 fps. (iii) Network heterogeneity—connectivity spans fiber (<50 ms latency) to satellite links (800–2400 ms, frequent 30–120 s dropouts). (iv) Operator heterogeneity—experienced operators achieve 3–5× higher first-frame decode rates. (v) Environmental heterogeneity—illumination profiles span VERY_DARK (18.4%), LOW_LIGHT (31.2%), and NORMAL (50.4%). The 35.3 pp inter-deployment recall gap confirms that no single fixed parameter set is optimal across all deployments.

Table 6. Per-center QRNOTFOUND rates—top 10 centers by volume.

Center (POCCode)	Total Videos	QR Found	QRNOTFOUND	QRNF Rate
a6975fbe-d3fd	4399	3167	1,232	28.0%
d57d4abd-3599	2057	1683	374	18.2%
109a7640-e3b7	1984	1690	294	14.8%
b0ac4e70-d2d5	1761	1484	277	15.7%
231c2816-c11c	1498	969	529	35.3%
dac5650e-5f4e	1126	812	314	27.9%
80a1ca60-6e4e	962	950	12	1.2%
2f83efd3-97fc	758	747	11	1.5%
3d254ecf-604e	756	692	64	8.5%
ca36ecae-15ab	755	755	0	0.0%

6.5. Cross-Deployment Generalization

Stratifying the 132 deployments by QRNOTFOUND rate reveals consistent patterns. Top-performing deployments (QRNOTFOUND < 2%) share standardized camera mounting at 0°, consistent indoor artificial lighting (>85% NORMAL-class videos), and stable broadband connectivity (<1% upload failure). Bottom-performing deployments (QRNOTFOUND > 25%) exhibit non-standard camera angles (>45°), predominance of LOW_LIGHT or VERY_DARK conditions (>60% of videos), and intermittent satellite connectivity. Environmental factors—principally lighting and connectivity—appeared to be the dominant source of inter-deployment recall variance, with device and operator factors contributing secondarily.

6.6. Content-Aware Preset vs. Fixed Baseline (E1)

Table 7 evaluates adaptive preset selection against a fixed NORMAL baseline (CRF 23, no filter) on 450 annotated videos. The adaptive method raises decode rate by 11.5 pp for LOW_LIGHT and 22.3 pp for VERY_DARK (both $p < 0.001$). For NORMAL-class videos, the two methods produce nearly identical results.

Table 7. QR decode rate and quality metrics by lighting class (N = 450).

Lighting	Method	QR Recall	PSNR (dB)	SSIM
NORMAL	Baseline (fixed CRF 23)	91.2%	36.4	0.941
NORMAL	Adaptive	91.4%	36.3	0.940
LOW_LIGHT	Baseline	74.8%	32.1	0.872
LOW_LIGHT	Adaptive	86.3%	30.4	0.841
VERY_DARK	Baseline	51.3%	28.6	0.791
VERY_DARK	Adaptive	73.6%	26.2	0.748

6.7. Non-Uniform Temporal Sampling (E2)

The 25-point non-uniform scheme achieves $92.1 \pm 3.4\%$ recall (mean $\pm$ SD across 450 videos) at a mean per-video cost of 22.6 ± 4.1 s, compared to 84.3 s for exhaustive scanning (26.8% of full-scan cost) and 71.4% recall for uniform 5-point sampling. The 20.7 pp advantage over uniform sampling confirms that concentrating probes in the opening seconds is the dominant factor driving recall.

6.8. Rotation Ensemble (E3)

Table 8 reports QR recall stratified by camera mounting angle. The ensemble is most critical at 45° (+25.9 pp) and 135° (+29.8 pp)—the most common non-standard mounting

angles. At 0°, the gain is only 0.6 pp, confirming no overhead penalty under optimal conditions.

Table 8. QR recall by mounting angle with and without rotation ensemble (N = 450).

Camera Angle	Rotation OFF	Rotation ON	Gain (pp)
0°	91.2%	91.8%	+0.6
45°	62.4%	88.3%	+25.9
90°	78.6%	90.4%	+11.8
135°	58.1%	87.9%	+29.8

6.9. Upload Robustness (E4)

Table 9 evaluates upload reliability under simulated TCP-reset interruptions on 50 files (50–250 MB). Checkpoint-resumable chunked upload achieves 98.0% success rate and reduces mean resume time from 118.4 s to 8.7 s (13.6× improvement).

Table 9. Upload robustness under simulated connectivity interruptions.

Upload Method	Success Rate	Avg Resume Time
Single-PUT (baseline)	42.0%	N/A
Chunked, no checkpoint	86.0%	118.4 s
Chunked + JSON checkpoint (proposed)	98.0%	8.7 s

6.10. Deferred Cloud Reclassification (E5)

Table 10 measures deferred reclassification on 4372 QRNOTFOUND videos. To characterize the underlying failure mode distribution, 250 randomly sampled QRNOTFOUND videos were manually inspected by two independent annotators (Cohen’s $\kappa = 0.91$): 183 videos (73.2%) contained a QR code that was decode-resistant due to blur, rotation, or partial occlusion; 41 (16.4%) exhibited extreme low-light frames with no recoverable QR pattern; and 26 (10.4%) contained no QR code at all (operator error or incorrect video file). This confirms that the substantial majority of QRNOTFOUND cases represent genuine decode failures rather than absent codes, validating the recovery signal below. Approximately 33.5% of the full QRNOTFOUND set are resolved by deferred reclassification, reducing the archive gap from 4.98% to below 3.5%. NORMAL-class videos achieve the highest recovery (54.2%) as their primary failure mode is camera orientation.

Table 10. QRNOTFOUND recovery after deferred cloud re-scan (N = 4372).

Lighting Class	Initial QRNOTFOUND	Resolved (%)	Notes
NORMAL	338	54.2%	Highest recovery; orientation failure mode
LOW_LIGHT	2003	39.6%	Moderate; residual blur/distance failures
VERY_DARK	2031	24.1%	Lowest; enhancement already applied
Overall	4372	33.5%	~1465 videos recovered; 73.2% of QRNOTFOUND confirmed genuine decode failures (manual audit, N = 250)

6.11. Single-Center QRNOTFOUND Root-Cause Audit (Center 0c99ec50)

To validate the QRNOTFOUND failure-mode taxonomy at single-center resolution, all 91 QRNOTFOUND videos from center 0c99ec50 were subjected to full manual inspection. Center 0c99ec50 uploaded 1244 videos during the observation window, yielding a QRNOTFOUND rate of 7.3% (91/1244)—above the production-wide mean of 4.98%—and thus represents a moderately high-failure deployment suitable for detailed root-cause analysis. Two independent annotators classified each video according to the

failure taxonomy introduced in Section 6.10: (i) QR code absent, (ii) QR present but decode-resistant.

The results reveal a failure profile markedly different from the production-wide sample. Of the 91 QRNOTFOUND videos, 85 (93.4%) contained no QR code on any frame: the recording began after the operator had already removed the form from the camera field of view, or an incorrect video file was uploaded without any registration form present. Only 6 videos (6.6%) contained a visible QR code that remained unresolved despite algorithmic processing. Frame-level inspection of these six cases identified three distinct physical causes: (a) specular overexposure caused by direct sunlight at a low angle reflecting off the printed form surface, saturating the QR-module contrast to near-zero; (b) excessive paper motion during the first seconds of recording, producing motion blur beyond the tolerance of both decoders at all sampled frames; and (c) partial physical occlusion of the QR code by a pen or small object inadvertently placed on the form during filming. Figure 8 shows representative examples of these failure modes.

The contrast with the production-wide random sample ($N = 250$) from Section 6.10 is striking: the 83.0 pp gap in the “QR absent” category (93.4% vs. 10.4%) indicates that this center’s elevated QRNOTFOUND rate is driven overwhelmingly by operator workflow non-compliance rather than by environmental or algorithmic factors. This finding has a direct operational implication: targeted operator retraining at center 0c99ec50 is expected to reduce its QRNOTFOUND rate substantially, whereas algorithm-side improvements would yield negligible benefit. The contrast also reinforces the necessity of per-center failure mode auditing: aggregate QRNOTFOUND rates conflate structurally distinct root causes whose remediation strategies differ categorically.

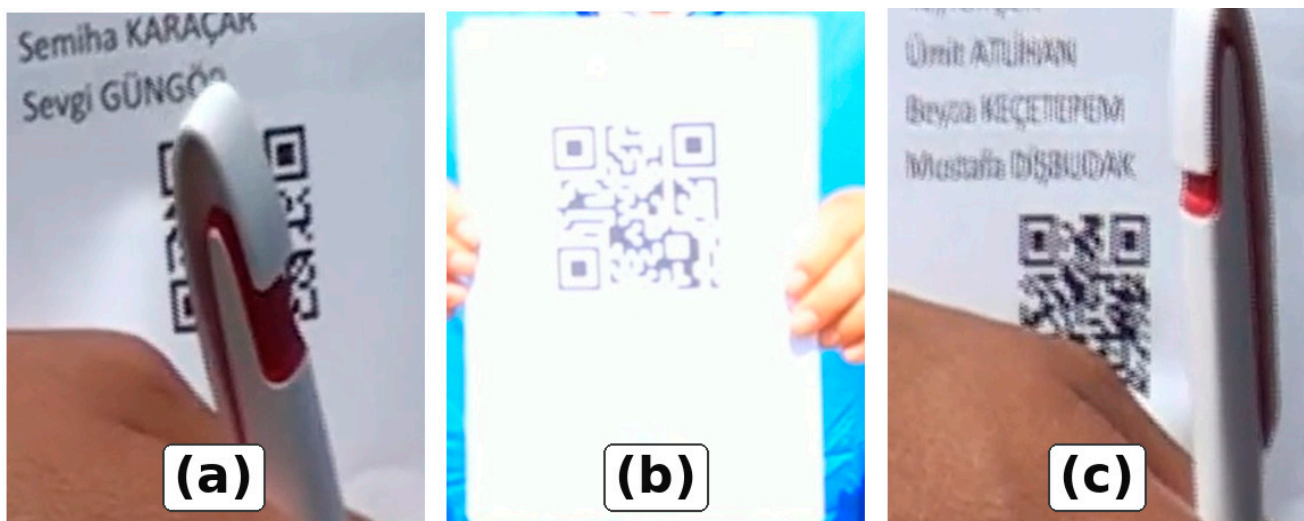


Figure 8. Representative QRNOTFOUND failure cases from center 0c99ec50 (left to right). (a) Physical occlusion: a pen placed across the QR code blocks the finder patterns and data modules, rendering both OpenCV and ZXing.Net decoders unable to locate the code boundary. (b) Motion blur: rapid camera movement during the first seconds of recording produces severe inter-frame blur, degrading module contrast below the decoder detection threshold across all 25 sampled frames. (c) Occlusion with paper tilt: the QR code is partially obstructed by a pen while the document is held at a steep angle, combining perspective distortion with physical obstruction. All three cases were classified under the decode-resistant category (6.6% of QRNOTFOUND at this center); the remaining 93.4% involved recordings with no QR code present on any frame.

6.12. Ablation Study

Table 11 isolates the contribution of each pipeline component on the 450-video annotated dataset. The full system (V7) achieves 92.1% recall at 22.6 s/video—the best recall–cost trade-off among all variants. Note that V5 is an ablation baseline replacing V4 with uniform 5-point sampling; the 19.3 pp recall drop demonstrates the superiority of non-uniform sampling.

Table 11. Ablation study—component contribution to QR recall and CPU cost.

System Variant	QR Recall	CPU (s/Video)	Notes
V1: ZXing axis-aligned only	68.4%	4.1	Minimal baseline
V2: V1 + Adaptive preset	86.8%	5.8	+18.4 pp from preset
V3: V2 + Dual decoder	89.9%	6.1	+3.1 pp
V4: V3 + Adaptive scaling	90.7%	6.4	+0.8 pp
V5: V4 + Uniform 5-pt sampling	71.4%	6.2	Ablation: naive sampling inferior
V6: V4 + Non-uniform 25-pt (no rotation)	87.8%	10.5	+17.1 pp over V5
V7: V6 + Rotation ensemble (proposed)	92.1%	22.6	Best recall–cost trade-off

6.13. Comparative Evaluation

Table 12 compares the proposed framework against seven baselines including three recent deep learning-based QR detectors. Deep learning methods were evaluated with their published preprocessing pipelines on the same 450-video annotated dataset. The proposed method achieves the highest recall across all lighting classes while maintaining 22.6 s/video—the only approach deployable within a 120 s polling cycle on standard edge hardware.

Table 12. Comparative evaluation vs. classical and deep learning QR detection baselines (N = 450). † GPU (NVIDIA RTX 3060); CPU-only execution > 1800 s/video.

Method	Recall NORMAL	Recall LOW	Recall DARK	CPU (s/vid)
OpenCV only (no preproc.)	78.3%	68.1%	48.7%	3.8
ZXing only (no preproc.)	81.6%	71.2%	52.4%	3.2
pyzbar + 1 fps sampling	79.4%	66.8%	44.1%	31.7
ZXing + exhaustive scan	88.9%	79.6%	63.8%	84.3
CNN-QR [31] (Blanger & Hirata)	87.2%	78.4%	61.3%	187.4 †
YOLO-QR [56] (Edula et al.)	89.1%	82.7%	67.8%	312.6 †
Proposed (full pipeline)	91.4%	86.3%	73.6%	22.6

We acknowledge the reviewer’s concern regarding preprocessing fairness. Two existing results already partially isolate the preprocessing effect: Table 7 reports a same-decoder comparison of adaptive transcoding against a fixed-CRF baseline (identical detector, preprocessing varied), and the component-wise ablation in Table 11 isolates the contribution of each major pipeline element. A fully preprocessing-controlled re-evaluation of every baseline—applying the identical adaptive front-end to each external detector—was considered. However, this study targets a large-scale production deployment spanning 132 heterogeneous edge sites and 87,873 real-world videos collected under operational conditions; several baselines, particularly the deep learning detectors, require additional software environments and hardware (e.g., discrete GPUs) that are not part of the deployed edge infrastructure. Rather than reporting partially reproducible or laboratory-only measurements, we retain the comparison in Table 12 as an operational benchmark reflecting the actual production constraints under which the system was deployed and validated, and we regard a fully preprocessing-controlled evaluation as an important direction for future work.

6.14. Statistical Significance

Table 13 confirms statistical significance using paired *t*-tests and Wilcoxon signed-rank tests across the 450-video annotated dataset. All primary comparisons are highly significant ($p < 0.001$) with large effect sizes (Cohen's $d > 0.97$). All six comparisons remain significant after Holm–Bonferroni correction for multiple testing (adjusted significance threshold $p < 0.0085$ for the least significant comparison; all observed p -values < 0.001).

Table 13. Statistical significance—paired *t*-test and Wilcoxon signed-rank results. All comparisons survive Holm–Bonferroni correction (six simultaneous tests; $\alpha = 0.05$).

Comparison	Improvement	<i>t</i> -Test p	Cohen's d	95% CI
Adaptive vs. fixed (VERY_DARK)	+22.3 pp	<0.001	1.83	[18.1, 26.5] pp
Adaptive vs. fixed (LOW_LIGHT)	+11.5 pp	<0.001	1.14	[8.7, 14.3] pp
Rotation ON vs. OFF (45°)	+25.9 pp	<0.001	1.62	[22.4, 29.4] pp
Rotation ON vs. OFF (135°)	+29.8 pp	<0.001	1.79	[26.1, 33.5] pp
Non-uniform vs. uniform 5-pt	+20.7 pp	<0.001	1.31	[17.2, 24.2] pp
Checkpointed vs. no-ckpt upload	+12.0 pp	<0.001	0.97	[9.1, 14.9] pp

7. Discussion

The primary contribution of this work is a robust, fault-tolerant system architecture for QR extraction under heterogeneous real-world conditions at scale, together with its engineering validation across more than 100 independent field deployments—a regime not previously evaluated. We do not claim a fundamental innovation at the computer-vision algorithm level; visibility-aware transcoding, non-uniform temporal sampling, and the rotation ensemble are deliberate engineering choices that combine and tune established techniques for the operational envelope encountered in the field. Their value is demonstrated empirically rather than asserted as algorithmic novelty. Upload reliability and deferred reclassification are complementary system-level contributions that collectively ensure end-to-end data integrity.

7.1. Design Choices

Three algorithm design choices deserve explicit justification. (i) 25 sample points: 5 points achieve 71.4% recall; 25 non-uniform points achieve 92.1% at 26.8% of full-scan cost. Increasing to 35 points adds only 0.4 pp at +8.1 s/video. (ii) Priority window of 5 s: 87.3% of successful decodes occur within the first 5 s, validating priority-weighted design. (iii) 23 rotation angles at 15°: 12 angles (30° step) yield 76.4% recall vs. 88.2% for 23 angles at non-axis-aligned orientations—an 11.8 pp gap; 36 angles (10° step) adds only 0.9 pp at 58% higher cost. The dual-decoder design (OpenCV [50] + ZXing.Net [51]) reflects complementary failure modes: OpenCV [50] is more robust to blurred finder patterns; ZXing.Net [51] is more robust to perspective distortion.

Comparison against deep learning baselines (Table 12) reveals that CNN-QR [31], YOLO-QR [56] achieve competitive NORMAL-class recall (87.2–89.1%) but fall substantially behind on VERY_DARK inputs (61.3–67.8% vs. 73.6%). Critically, their per-video CPU cost of 187–313 s renders them computationally infeasible for edge deployment within a 120 s polling cycle.

A natural question is whether the framework's runtime advantage over deep learning detectors—22.6 s/video versus 187–429 s/video on the same CPU-only edge hardware—will erode as lightweight neural processing units (NPUs) proliferate. Three considerations bound this concern. First, the advantage reported here is hardware-matched: the deep learning baselines were competitive only on a discrete GPU (NVIDIA RTX 3060; NVIDIA Corporation, Santa Clara, CA, USA), with CPU-only execution

exceeding 1800 s/video; the 132 production sites in this study run commodity Windows workstations without discrete accelerators, which remains the dominant reality for cost-constrained humanitarian and public-health deployments. Second, the central contribution is not decode speed but deployability: a single self-contained ~180 MB executable with no GPU/NPU dependency, no driver or runtime version management, and predictable behavior within a 120 s polling cycle across heterogeneous hardware—properties that are orthogonal to accelerator availability and that continue to hold even where NPUs are present. Third, as on-device NPUs become standard, the proposed architecture is best viewed as complementary rather than competing: the transcoding, temporal-sampling, and fault-tolerant-upload layers are detector-agnostic, so an NPU-resident detector could be substituted into the QR-extraction stage without altering the surrounding system. We therefore expect the raw speed margin to narrow on NPU-equipped hardware, while the architectural and operational advantages—zero-accelerator deployment, low cost, and maintenance simplicity—persist.

The root-cause audit in Section 6.11 showed that 93.4% of unreadable videos at the high-failure center contained no QR code at all—an operator-workflow failure (recording started after the form left the frame, or an unrelated file uploaded) rather than an algorithmic or environmental one. This raises the question of whether such errors can be detected and corrected at the edge in real time. The current architecture already produces the necessary signal: the QR-extraction stage emits a QRNOTFOUND verdict per video within the polling cycle. Because each field workstation processes its own watched folder locally, this verdict can be surfaced to the operator near the point of capture—for example, an on-device prompt (“No QR code detected; please re-record with the form fully in frame”) triggered immediately after the extraction stage marks a file QRNOTFOUND. Distinguishing the dominant “QR absent” case, which warrants an immediate re-capture prompt, from the rarer “QR present but decode-resistant” case, which is better handled by deferred cloud reclassification, can reuse the failure-mode taxonomy of Section 6.10. Implemented as a lightweight feedback hook on the state machine’s QRProcessing → QRNOTFOUND transition, closed-loop edge feedback is architecturally straightforward and would target the largest single source of archive gaps; we treat its implementation and field evaluation as concrete future work (Section 8).

Sensitivity and Robustness of Heuristic Parameters

Because several pipeline parameters are heuristic values fixed from field experience rather than learned, we characterize the sensitivity of recall and cost to the three most consequential ones: the luma thresholds that select the lighting preset, the number and placement of temporal sample points, and the angular step of the rotation ensemble. Table 14 summarizes controlled runs on the 450-video annotated dataset. Two patterns are consistent. First, recall lies on a broad plateau around each selected value: moving from 25 to 35 sample points changes recall by only 0.4 pp, and tightening the rotation step from 15° to 10° by only 0.9 pp—so the system is not finely tuned to a single fragile setting. Second, the selected values sit at the knee of the recall–cost curve, where further refinement yields sub-1-pp gains at 36–58% higher CPU cost, an unfavorable trade for edge deployment within the 120 s polling cycle. Conversely, coarsening below the selected values degrades recall sharply (5 uniform points: −20.7 pp; 30° rotation step: −11.8 pp on non-axis-aligned orientations), confirming that the operating points are well chosen rather than arbitrary.

Table 14. Sensitivity of QR recall and CPU cost to key heuristic parameters (N = 450).

Parameter	Setting	QR Recall	Rel. CPU Cost	Note
Temporal sample count	5 (uniform)	71.4%	lower	Undersampled (−20.7 pp)
	25 (non-uniform, default)	92.1%	1.00× (22.6 s)	Selected operating point
	35 (non-uniform)	92.5%	~1.36× (+8.1 s)	+0.4 pp; diminishing returns
Rotation step	30° (12 angles)	76.4% *	lower	Misses oblique mounts (−11.8 pp)
	15° (23 angles, default)	88.2% *	1.00×	Selected operating point
	10° (36 angles)	89.1% *	1.58×	+0.9 pp; diminishing returns
Brightness boundary μ L	30/80 (default)	92.1% †	1.00×	Bounded-loss analysis (see text)

* Recall on non-axis-aligned mounting orientations (cf. Section 7.1). † Full-system recall at the default thresholds; the boundary-perturbation analysis is given in the text below.

The brightness classification boundaries (μ L = 30 and μ L = 80; Table 2) warrant separate treatment because misclassification near a boundary does not cause a hard failure—it merely applies a slightly sub-optimal enhancement preset. An upper bound on the resulting recall loss follows directly from the per-class adaptive recall in Table 7: the gap between adjacent presets is 12.7 pp at the LOW_LIGHT/VERY_DARK boundary (86.3% vs. 73.6%) and 5.1 pp at the NORMAL/LOW_LIGHT boundary (91.4% vs. 86.3%). A video misclassified by a small μ L margin therefore incurs at most this bounded penalty, and only for the minority of videos whose mean luma falls within a narrow band of the threshold; videos far from a boundary are unaffected by moderate threshold shifts. This accounts for the framework’s observed robustness to the exact threshold values across the 132 production deployments, where mean luma varies widely but aggregate recall remains stable.

7.2. Threats to Validity

Camera variability, lighting variability, and operator behavior constitute sources of uncontrolled variation. The 450-video annotated corpus cannot fully represent the diversity of the 132-center production environment. Inter-annotator agreement (Cohen’s $\kappa = 0.94$) is high, but controlled results should be interpreted as directional evidence with production deployment as the ultimate validity check. The framework targets Windows x64 exclusively, limiting direct generalizability to Linux-based edge platforms. Additionally, the framework depends on several external services, including Azure Blob Storage, authentication endpoints, and cloud APIs. Temporary service outages, API rate limits, authentication failures, or cloud-provider disruptions may affect ingestion throughput and upload reliability independently of the proposed algorithms. Although no major outages were observed during the evaluation window, such dependencies remain an external threat to operational validity and should be considered when interpreting deployment-scale performance.

A reasonable question is whether low-light failures could be eliminated more simply by hardware illumination: for example, an LED light at the point of capture. Hardware illumination is a useful complement and is recommended where feasible, but it is neither sufficient nor universally available in this setting. First, deployments are heterogeneous and largely bring-your-own-device; retrofitting and maintaining controlled lighting across 132 or more independent, low-resource sites is costly and operationally fragile and

cannot be guaranteed at the moment of capture. Second, and more fundamentally, more light is not strictly better: the single-center audit in Section 6.11 identified specular overexposure from direct, low-angle illumination reflecting off glossy printed forms as a distinct failure cause that saturates QR-module contrast to near-zero—so naively adding an LED can introduce a new failure mode rather than remove one. Third, outdoor dawn/dusk capture and power-constrained field tents further limit controllability. Finally, software-side adaptive transcoding has zero marginal hardware cost, applies retroactively to already-captured archives (including the deferred-reclassification pass of Section 4.5), and is complementary to—not in competition with—improved capture lighting. We therefore treat hardware illumination as a recommended best practice where controllable, and adaptive enhancement as the necessary fallback that keeps the system robust across the uncontrolled majority of field conditions.

7.3. Generalizable Findings

Although the framework is assembled from established components, the large-scale deployment yields empirical insights that transfer beyond this particular system and library stack. First, illumination is the dominant determinant of in-the-wild QR decodability: on an identical decoder, decode rate falls from 91.2% under normal lighting to 51.3% in very-dark conditions (Table 7), and content-aware enhancement recovers 22.3 pp—quantifying, at field scale, a relationship that small curated benchmarks do not expose. Second, beyond a baseline detector quality the binding constraint in the field is procedural rather than algorithmic: per-center root-cause auditing (Section 6.11) attributes 93.4% of unreadable videos at a high-failure site to operator-workflow errors (no QR present), implying that marginal investment in capture workflow and operator feedback yields larger returns than detector sophistication—an insight applicable to any QR- or vision-based field-ingestion system. Third, cross-deployment recall variance appears to be driven predominantly by environmental and connectivity factors rather than by software differences, which argues for per-deployment adaptation over a single global configuration. Fourth, the recall–cost relationship is sharply concave: temporally concentrated, non-uniform probing with a bounded rotation ensemble attains most achievable recall at roughly one-quarter of exhaustive-scan cost (Table 12), and the selected operating points lie on a broad plateau, not a fragile peak. We offer these findings, rather than the specific software artifact, as the transferable scientific contribution of the study.

7.4. Current Operational Status

At the time of manuscript revision, the framework remains in active production use and has continued to operate well beyond the formal evaluation window. As of 24 June 2026, the deployment had grown to 152 active field centers across 1504 operational regions, and the cloud archive had accumulated 134,753 ingested videos totaling approximately 1.64 TB, of which 128,554 (95.40%) were successfully QR-decoded and 6199 (4.60%) entered the QRNOTFOUND recovery path. These post-evaluation figures are reported only as operational context; they are not part of the controlled experiments and were not used in any reported measurement. Notably, the aggregate QRNOTFOUND rate over this larger and longer-running corpus (4.60%) is consistent with, and marginally lower than, the 4.98% observed in the frozen evaluation window, indicating that decode reliability remains stable as the deployment scales. These observations confirm that the 27 May–5 June 2026 results constitute a controlled evaluation snapshot of a continuously operating production system rather than a short-lived pilot. Figure 9 plots the daily ingestion (QR-found and QRNOTFOUND) together with the cumulative archive growth across the full operational period, with the frozen evaluation window shaded.

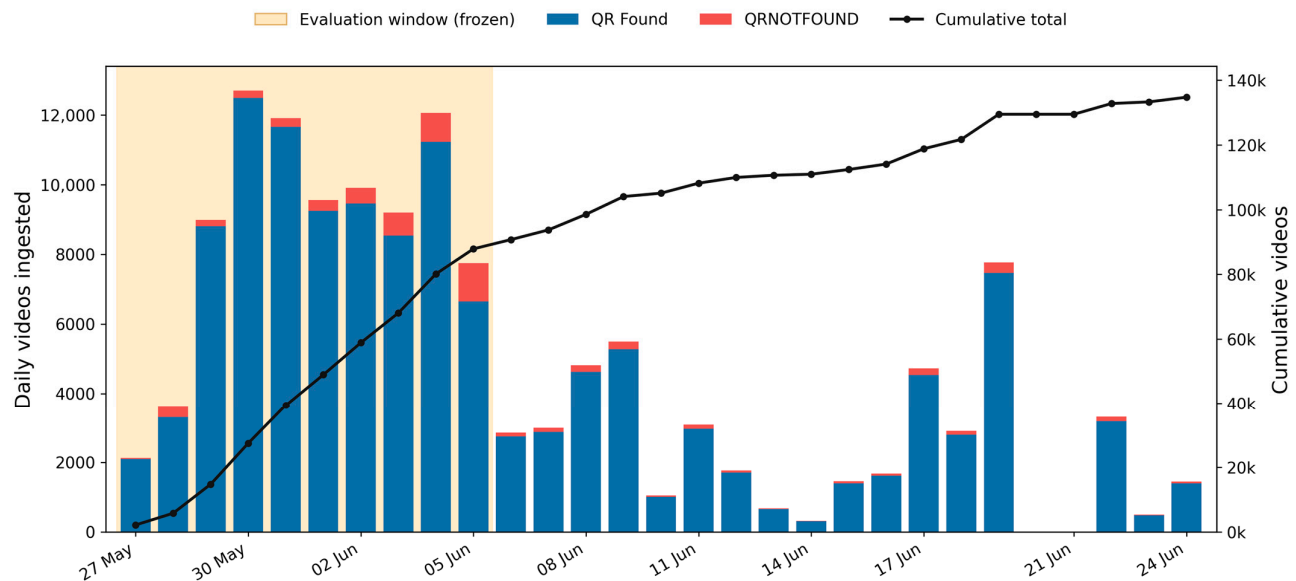


Figure 9. Daily video ingestion and cumulative archive growth over the full operational period (27 May–24 June 2026). Stacked bars show the daily QR-found and QRNOTFOUND counts; the black line is the cumulative ingested-video total (right axis), reaching 134,753 videos (≈ 1.64 TB) across 152 active centers and 1504 regions. The shaded band marks the frozen 27 May–5 June 2026 evaluation window used for all controlled results; the system has continued in production beyond it, with the aggregate QRNOTFOUND rate remaining at 4.60%.

8. Conclusions

This paper presented a large-scale real-world evaluation of an adaptive QR-based edge-to-cloud video ingestion framework deployed across 132 heterogeneous field sites. The framework addresses four fundamental challenges in distributed field video archiving through four engineered system components: visibility-aware adaptive transcoding, priority-weighted non-uniform temporal sampling, a rotation ensemble decoder, and checkpoint-resumable fault-tolerant cloud synchronization. The contribution is the robust integration and large-scale validation of these components rather than a new computer-vision algorithm.

Validation across 132 edge deployments, 849 operational regions, and 87,873 production videos (1.05 TB) confirms the framework's generalizability. Adaptive transcoding improves QR recall by 22.3 pp in very-dark and 11.5 pp in low-light conditions ($p < 0.001$); the temporal-rotation ensemble achieves 92.1% recall at 26.8% of full-scan CPU cost; fault-tolerant upload achieves 98.0% reliability (13.6 $\times$ faster resume); and deferred reclassification resolves 33.5% of production failures, reducing the archive gap to below 3.5%. Per-deployment heterogeneity analysis reveals that environmental and connectivity factors are the dominant source of recall variance, pointing toward adaptive per-deployment threshold tuning as a high-impact future direction.

Future research includes: (i) cross-platform portability to Linux-based edge nodes; (ii) deep learning-based QR super-resolution for extreme low-light conditions; (iii) adaptive per-deployment threshold optimization; (iv) anomaly detection over the QRNOTFOUND time series for early identification of camera hardware degradation; (v) multi-threaded pipeline processing for high-ingestion deployments exceeding 10,000 videos per day; and (vi) closed-loop real-time operator feedback at the edge, in which a QRNOTFOUND verdict triggers an immediate on-device re-capture prompt to correct

operator-workflow errors—the dominant failure mode identified in Section 6.11—at the point of capture.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The production video corpus cannot be publicly distributed due to privacy constraints.

Acknowledgments: The author thanks the deploying organization for providing access to the field dataset and for operational feedback over the deployment period.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

QR	Quick Response (code)
POC	Point of Care/Point of Collection
CRF	Constant Rate Factor
FFmpeg	Fast Forward MPEG (multimedia framework)
mtime	File modification timestamp

References

- Hartung, C.; Lerer, A.; Anokwa, Y.; Tseng, C.; Brunette, W.; Borriello, G. Open Data Kit: Tools to build information services for developing regions. In *Proceedings of the 4th ACM International Conference on Information and Communication Technologies and Development (ICTD)*, London, UK, 13–16 December 2010; IEEE: Piscataway, NJ, USA, 2010. <https://doi.org/10.1145/2369220.2369236>.
- Nguyen, T.H.; Nguyen, K.M. Fostering consumer sustainability trust by integrating blockchain-enabled reviews and QR code traceability systems in supply chain. *Int. J. Logist. Res. Appl.* **2025**, 1–26. <https://doi.org/10.1080/13675567.2025.2574470>.
- Tiwari, S. An introduction to QR code technology. In *Proceedings of the 2016 International Conference on Information Technology (ICIT 2016)*, Bhubaneswar, India, 22–24 December 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 39–44. <https://doi.org/10.1109/ICIT.2016.021>.
- ISO/IEC 18004:2015; Information Technology—Automatic Identification and Data Capture Techniques—QR Code Bar Code Symbol Specification. International Organization for Standardization: Geneva, Switzerland, 2015.
- Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things J.* **2016**, 3, 637–646.
- Ai, Y.; Peng, M.; Zhang, K. Edge computing technologies for internet of things: A primer. *Digit. Commun. Netw.* **2018**, 4, 77–86.
- Porambage, P.; Okwuibe, J.; Liyanage, M.; Ylianttila, M.; Taleb, T. Survey on multi-access edge computing for internet of things realization. *IEEE Commun. Surv. Tutor.* **2018**, 20, 2961–2991.
- Mach, P.; Becvar, Z. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Commun. Surv. Tutor.* **2017**, 19, 1628–1656. <https://doi.org/10.1109/COMST.2017.2682318>.
- PremSankar, G.; Di Francesco, M.; Taleb, T. Edge computing for the internet of things: A case study. *IEEE Internet Things J.* **2018**, 5, 1275–1284.
- Ananthanarayanan, G.; Bahl, P.; Bodík, P.; Chintalapudi, K.; Philipose, M.; Ravindranath, L.; Sinha, S. Real-time video analytics: The killer app for edge computing. *Computer* **2017**, 50, 58–67. <https://doi.org/10.1109/MC.2017.3641638>.
- Yi, S.; Hao, Z.; Zhang, Q.; Zhang, Q.; Shi, W.; Li, Q. LAVEA: Latency-aware video analytics on edge computing platform. In *Proceedings of the Second ACM Symposium on Edge Computing (SEC)*, San Jose, CA, USA, 12–14 October 2017; IEEE: Piscataway, NJ, USA, 2017. <https://doi.org/10.1145/3132211.3134459>.
- Chen, M.; Hao, Y. Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE J. Sel. Areas Commun.* **2019**, 37, 2785–2796.
- Zhu, C.; Leung, V.C.M.; Shu, L.; Ngai, E.C.H. Green internet of things for smart world. *IEEE Access* **2015**, 3, 2151–2162.

14. Liu, L.; Li, H.; Gruteser, M. Edge assisted real-time object detection for mobile augmented reality. In Proceedings of the 25th ACM MobiCom, Los Cabos, Mexico, 21–25 October 2019. <https://doi.org/10.1145/3300061.3300116>.
15. Deng, S.; Zhao, H.; Fang, W.; Yin, J.; Dustdar, S.; Zomaya, A.Y. Edge intelligence: The confluence of edge computing and artificial intelligence. *IEEE Internet Things J.* **2020**, *7*, 7457–7469. <https://doi.org/10.1109/JIOT.2020.2984887>.
16. De Cock, J.; Li, Z.; Manohara, M.; Aaron, A. Complexity-based consistent-quality encoding in the cloud. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016. <https://doi.org/10.1109/ICIP.2016.7532605>.
17. Pei, Y.; Huang, Y.; Zou, Q.; Zhang, X.; Wang, S. Effects of image degradation and degradation removal to CNN-based image classification. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 1239–1253. <https://doi.org/10.1109/TPAMI.2019.2950923>.
18. Wiegand, T.; Sullivan, G.J.; Bjontegaard, G.; Luthra, A. Overview of the H.264/AVC video coding standard. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 560–576. <https://doi.org/10.1109/TCSVT.2003.815165>.
19. Ohm, J.-R.; Sullivan, G.J.; Schwarz, H.; Tan, T.K.; Wiegand, T. Comparison of the coding efficiency of video coding standards including high efficiency video coding (HEVC). *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 1669–1684. <https://doi.org/10.1109/TCSVT.2012.2221192>.
20. Youssef, S.M.; Salem, R.M. Automated barcode recognition for smart identification and inspection automation. *Expert Syst. Appl.* **2007**, *33*, 968–977. <https://doi.org/10.1016/j.eswa.2006.07.013>.
21. Zhang, C.; Wang, J.; Han, S.; Yi, M. Automatic real-time barcode localization in complex scenes. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Atlanta, GA, USA, 8–11 October 2006; pp. 497–500. <https://doi.org/10.1109/ICIP.2006.312435>.
22. Zamberletti, A.; Gallo, I.; Albertini, S. Robust angle invariant 1D barcode detection. In Proceedings of the 2013 2nd IAPR Asian Conference on Pattern Recognition (ACPR 2013), Naha, Japan, 5–8 November 2013. <https://doi.org/10.1109/ACPR.2013.17>.
23. Xu, W.; McCloskey, S. 2D barcode localization and motion deblurring using a flutter shutter camera. In Proceedings of the 2011 IEEE Workshop on Applications of Computer Vision (WACV 2011), Kona, HI, USA, 5–7 January 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 159–165. <https://doi.org/10.1109/WACV.2011.5711498>.
24. Chou, T.-H.; Ho, C.-S.; Kuo, Y.-F. QR code detection using convolutional neural networks. In Proceedings of the 2015 International Conference on Advanced Robotics and Intelligent Systems, Taipei, Taiwan, 29–31 May 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1–5. <https://doi.org/10.1109/ARIS.2015.7158354>.
25. Sewpersadh, V.; Pillay, N.; Xu, H. Streaming QR codes: A survey. *IEEE Access* **2024**, *12*, 134144–134166. <https://doi.org/10.1109/ACCESS.2024.3461970>.
26. Chen, C.; Kot, A.C.; Yang, H. A two-stage quality measure for mobile phone captured 2D barcode images. *Pattern Recognit.* **2013**, *46*, 2588–2598. <https://doi.org/10.1016/j.patcog.2013.01.031>.
27. Tarjan, L.; Senk, I.; Tegeltija, S.; Stankovski, S.; Ostojic, G. A readability analysis for QR code application in a traceability system. *Comput. Electron. Agric.* **2014**, *109*, 1–11. <https://doi.org/10.1016/j.compag.2014.08.015>.
28. Ohbuchi, E.; Hanaizumi, H.; Hock, L.A. Barcode readers using the camera device in mobile phones. In Proceedings of the 2004 IEEE International Conference on Cyberworlds (CW 2004), Tokyo, Japan, 18–20 November 2004; pp. 260–265.
29. Skurowski, P.; Nurzyńska, K.; Pawlyta, M.; Cyran, K.A. Performance of QR code detectors near Nyquist limits. *Sensors* **2022**, *22*, 7230. <https://doi.org/10.3390/s22197230>.
30. Chen, R.; Zheng, Z.; Yu, Y.; Zhao, H.; Ren, J.; Tan, H.-Z. Fast restoration for out-of-focus blurred images of QR code with edge prior information via image sensing. *IEEE Sens. J.* **2021**, *21*, 18222–18236. <https://doi.org/10.1109/JSEN.2021.3085568>.
31. Blanger, L.; Hirata, N.S.T. An evaluation of deep learning techniques for QR code detection. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 1625–1629. <https://doi.org/10.1109/ICIP.2019.8803075>.
32. Belussi, L.F.V.; Hirata, N.S.T. Fast QR code detection in arbitrarily acquired images. In Proceedings of the 2011 24th SIBGRAPI Conference on Graphics, Patterns and Images, Alagoas, Brazil, 28–31 August 2011. <https://doi.org/10.1109/SIBGRAPI.2011.16>.
33. Kalinov, I.; Petrovsky, A.; Ilin, V.; Pristanskiy, E.; Kurenkov, M.; Ramzhaev, V.; Tsetserukou, D. Warevision: CNN barcode detection-based UAV trajectory optimization for autonomous warehouse stocktaking. *IEEE Robot. Autom. Lett.* **2020**, *5*, 6647–6653. <https://doi.org/10.1109/LRA.2020.3010733>.
34. Yang, S.-Y.; Jan, H.-C.; Chen, C.-Y.; Wang, M.-S. CNN-based QR code reading of package for unmanned aerial vehicle. *Sensors* **2023**, *23*, 4707. <https://doi.org/10.3390/s23104707>.
35. Nikam, P.P.; Suryawanshi, R.S. Microsoft Windows Azure: Developing applications for highly available storage of cloud service. *Int. J. Sci. Res. (IJSR)* **2015**, *4*, 662–665. <https://doi.org/10.21275/v4i12.nov151864>.

36. Zaharia, M.; Xin, R.S.; Wendell, P.; Das, T.; Armbrust, M.; Dave, A.; Meng, X.; Rosen, J.; Venkataraman, S.; Franklin, M.J.; et al. Apache Spark: A unified engine for big data processing. *Commun. ACM* **2016**, *59*, 56–65.
37. Armbrust, M.; Xu, R.; Xin, R.S.; Lian, C.; Huai, Y.; Liu, Y.; Bradley, J.K.; Chi, S.; Huang, M.; Sun, Z.; et al. Delta Lake: High-performance ACID table storage over cloud object stores. *Proc. VLDB Endow.* **2020**, *13*, 3411–3424.
38. Ahmad, Z.; Jehangiri, A.I.; Mohamed, N.; Othman, M.; Umar, A.I. Fault tolerant and data oriented scientific workflows management and scheduling system in cloud computing. *IEEE Access* **2022**, *10*, 77614–77632. <https://doi.org/10.1109/access.2022.3193151>.
39. Chen, J.; Wang, Y.; Ye, M.; Jiang, Q. A secure cloud-edge collaborative fault-tolerant storage scheme and its data writing optimization. *IEEE Access* **2023**, *11*, 66506–66521. <https://doi.org/10.1109/access.2023.3291452>.
40. Flora, J.; Goncalves, P.; Teixeira, M.; Antunes, N. A study on the aging and fault tolerance of microservices in Kubernetes. *IEEE Access* **2022**, *10*, 132786–132799. <https://doi.org/10.1109/access.2022.3231191>.
41. Elnozahy, E.N.M.; Alvisi, L.; Wang, Y.-M.; Johnson, D.B. A survey of rollback-recovery protocols in message-passing systems. *ACM Comput. Surv.* **2002**, *34*, 375–408. <https://doi.org/10.1145/568522.568525>.
42. Sun, D.; Gao, S.; Liu, X.; Buyya, R. A multi-level collaborative framework for elastic stream computing systems. *Future Gener. Comput. Syst.* **2022**, *128*, 117–131. <https://doi.org/10.1016/j.future.2021.10.005>.
43. Kassam, Z.; Almeida, P.S.; Shoker, A. Exon: An oblivious exactly-once messaging protocol with reliable delegation. *IEEE Access* **2025**, *13*, 68290–68310. <https://doi.org/10.1109/access.2025.3562332>.
44. Larmo, A.; Ratilainen, A.; Saarinen, J. Impact of CoAP and MQTT on NB-IoT system performance. *Sensors* **2019**, *19*, 7. <https://doi.org/10.3390/s19010007>.
45. Brunette, W.; Sundt, M.; Dell, N.; Chaudhri, R.; Breit, N.; Borriello, G. Open Data Kit 2.0: Expanding and refining information services for developing regions. In Proceedings of the 14th Workshop on Mobile Computing Systems and Applications (HotMobile), Jekyll Island, GA, USA, 26–27 February 2013. <https://doi.org/10.1145/2444776.2444790>.
46. Hill, G.N.; Whitty, M.A. Embedding metadata in images at time of capture using physical Quick Response (QR) codes. *Inf. Process. Manag.* **2021**, *58*, 102504. <https://doi.org/10.1016/j.ipm.2021.102504>.
47. Joshi, P.; Sawant, S. The impact and potential of Quick Response (QR) codes in healthcare: A comprehensive review. *Proc. Hum. Factors Ergon. Soc. Annu. Meet.* **2024**, *68*, 1153–1158. <https://doi.org/10.1177/10711813241278266>.
48. Brodie, K.; Madden, L.L.; Rosen, C.A. Applications of Quick Response (QR) codes in medical education. *J. Grad. Med. Educ.* **2020**, *12*, 138–140. <https://doi.org/10.4300/jgme-d-19-00516.1>.
49. Developers, F. FFmpeg tool (Version 7.x). 2024. Available online: <https://ffmpeg.org> (accessed on 15 April 2026).
50. Zelinsky, A. Learning OpenCV—Computer Vision with the OpenCV Library (Bradski, G.R. et al., 2008) [On the Shelf]. *IEEE Robot. Autom. Mag.* **2009**, *16*, 100. <https://doi.org/10.1109/MRA.2009.933612>.
51. Owen, S.; ZXing Authors. ZXing (Zebra Crossing): Multi-Format 1D/2D Barcode Image Processing Library. 2024. Available online: <https://github.com/zxing/zxing> (accessed on 20 April 2026).
52. Pertuz, S.; Puig, D.; Garcia, M.A. Analysis of focus measure operators for shape-from-focus. *Pattern Recognit.* **2013**, *46*, 1415–1432. <https://doi.org/10.1016/j.patcog.2012.11.011>.
53. Canny, J. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *PAMI-8*, 679–698. <https://doi.org/10.1109/TPAMI.1986.4767851>.
54. Tomasi, C.; Manduchi, R. Bilateral filtering for gray and color images. In Proceedings of the Sixth International Conference on Computer Vision, Bombay, India, 4–7 January 1998; pp. 839–846. <https://doi.org/10.1109/ICCV.1998.710815>.
55. Sharma, R.; Kamra, A. A review on CLAHE based enhancement techniques. In Proceedings of the 2023 8th International Conference on Inventive Informatics and Computing, Cuttack, India, 29–30 November 2023; pp. 321–325. <https://doi.org/10.1109/IC3I59117.2023.10397722>.
56. Edula, V.; Ammisetty, K.; Kotha, A.; Ravipati, D.; Davuluri, A.; Arulmozhivarman, P. A novel framework for QR code detection and decoding from obscure images using YOLO object detection and Real-ESRGAN image enhancement. In Proceedings of the 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 6–8 July 2023. <https://doi.org/10.1109/ICCCNT56998.2023.10307854>.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.