CAR-READER®-DLE

Benutzerhandbuch und Integrationsdokumentation

Inhaltsübersicht

Hinweis: Endpunkte und Optionen können abhängig von Lizenz, Plattform und installierter Produktedition verfügbar sein.

1. Produktüberblick

CAR-READER®-DLE ist eine lokal betriebene Runtime zur automatischen Erkennung von Fahrzeugkennzeichen in Bildern und Kamerastreams. Die Verarbeitung erfolgt auf dem Kundensystem. Für die Erkennung ist keine Cloudverbindung erforderlich.

1.1 Typische Einsatzbereiche

- Zufahrts- und Schrankensteuerung
- Parkplatz- und Parkraummanagement
- Werkszufahrten und Logistikstandorte
- Besucher- und Flottenmanagement
- OEM-Lösungen für Kameras, Embedded-Systeme und Branchensoftware
- Lokale Video- und Sicherheitssysteme

1.2 Produkteigenschaften

- Lokale REST-API für einfache Integration
- Einzelbild-, Batch- und RTSP-Verarbeitung
- Unterstützung mehrerer Kameras und Fahrspuren
- Optimierte parallele Verarbeitung und automatische Lastbegrenzung
- Konfigurierbare parallele Verarbeitung
- Profile für internationale Kennzeichen und zweizeilige Kennzeichen
- Optionale annotierte Ergebnisbilder
- Datenschutzfreundliche Standardwerte ohne dauerhafte Speicherung hochgeladener Bilder
- Windows-, Linux-x64- und Linux-ARM64-Betrieb, je nach Paket

1.3 Grundprinzip

Eine Anwendung sendet ein Bild oder verbindet eine Kamera mit der Runtime. Die Runtime liefert ein strukturiertes JSON-Ergebnis mit erkanntem Text, formatierter Darstellung, Confidence, Profil- und Layoutinformationen. Die aufrufende Anwendung entscheidet anschließend über Schranke, Protokollierung, Datenbankabgleich oder weitere Geschäftslogik.

2. Systemvoraussetzungen und Installation

2.1 Unterstützte Plattformen

Plattform	Typischer Einsatz
Windows x64	Arbeitsplatz, Server, Leitstand, Integration in Windows-Anwendungen
Linux x64	Server, Appliance, Container-nahe Systemintegration
Linux ARM64	Edge-Verarbeitung für einzelne oder wenige Kameras

2.2 Hardwareempfehlungen

Szenario	Empfehlung
Einzelkamera / gelegentliche Bilder	Moderner Mehrkernprozessor, mindestens 4 bis 8 GB RAM empfohlen
4 Kameras, ereignisgesteuert	Moderner Büro-PC, vorzugsweise 8 oder mehr logische Prozessoren und 8 bis 16 GB RAM
8 Kameras, ereignisgesteuert	Leistungsfähiger Mehrkern-PC, 16 GB RAM oder mehr; Lasttest empfohlen
Große Installationen	Leistungsfähiger Server oder mehrere Runtime-Instanzen; projektspezifische Dimensionierung

Die erreichbare Geschwindigkeit hängt von Prozessor, Bildauflösung, Profil, Anzahl gleichzeitiger Ereignisse und gewünschter Latenz ab. Vor Produktivbetrieb sollte ein Lasttest auf der Zielhardware erfolgen.

2.3 Installation

Windows

```
CAR-READER-DLE_Runtime_Setup.exe als Administrator ausführen.
```

Linux

```
sudo bash ./CAR-READER-DLE_Runtime_Setup.run
```

Nach der Installation läuft die Runtime standardmäßig als lokaler Dienst oder wird über die mitgelieferten Start-/Stop-Tools gesteuert.

2.4 Funktion prüfen

```
curl -s http://127.0.0.1:5099/health
```

Eine erfolgreiche Antwort enthält den Runtime-, Lizenz- und grundlegenden Betriebsstatus..

3. Grundkonzept und Betriebsarten

3.1 Einzelbildmodus

Ein Bild wird unmittelbar ausgewertet. Dieser Modus eignet sich für Anwendungen, die bereits Kamerabilder oder Snapshots erzeugen.

3.2 Batch-Modus

Ein lokaler Ordner wird verarbeitet. Dieser Modus eignet sich für Tests, Migrationen, Qualitätsprüfungen und größere Bildbestände auf demselben Rechner.

3.3 RTSP-/Live-Modus

Die Runtime verbindet sich mit einem RTSP-Stream, hält aktuelle Frames bereit und kann Bilder manuell oder ereignisgesteuert analysieren. Mehrere Kameras werden über eindeutige camera_id- beziehungsweise lane_id-Werte getrennt verwaltet.

3.4 Einzel- und Mehrfachkennzeichen

Im Standardmodus wird das beste Kennzeichen eines Bildes zurückgegeben. Im Mehrfachkennzeichenmodus können mehrere sichtbare Kennzeichen erkannt und als geordnete Liste geliefert werden. Dieser Modus eignet sich beispielsweise für Parkplatzübersichten oder Szenen mit mehreren Fahrzeugen.

4. Erkennungsmodi und Profile

4.1 Profile im Überblick

Profil	Verwendung	Empfehlung
international_latin	Allgemeiner Modus für lateinische Kennzeichen	Gute Standardwahl bei bekanntem, überwiegend einzeiligem Kennzeichenbestand
auto	Automatische Auswahl geeigneter Format- und Layoutlogik	Empfohlen bei gemischten Ländern, niederländischen oder ein- und zweizeiligen Kennzeichen
de	Deutschland	Wenn ausschließlich deutsche Kennzeichen erwartet werden
at	Österreich	Wenn ausschließlich österreichische Kennzeichen erwartet werden
ch	Schweiz	Wenn ausschließlich schweizerische Kennzeichen erwartet werden
nl	Niederlande	Für niederländische Formatierung mit mehreren Separatoren
two_line	Zweizeilige Kennzeichen	Wenn die Kamera überwiegend zweizeilige Kennzeichen aufnimmt

4.2 Auswahlhilfe

- Bekanntes einzeiliges Kennzeichenformat: international_latin oder passendes Länderprofil.
- Gemischte Länder und Layouts: auto.
- Überwiegend zweizeilige Kennzeichen: two_line.
- Niederländische Kennzeichen: nl oder auto.
- Höchster Durchsatz: möglichst spezifisches Profil wählen, sofern das Einsatzgebiet eindeutig ist.

4.3 Formatierter und roher Text

Das Ergebnis kann sowohl den normalisierten Text ohne Separatoren als auch die formatierte Darstellung enthalten. Beispiel: text = BMS1234, formatted_text = B-MS1234. Integrationen sollten für Vergleiche meist den normalisierten Text und für Anzeigezwecke den formatierten Text verwenden.

4.4 Confidence

Die Confidence ist ein Wert zwischen 0 und 1. Höhere Werte weisen auf eine höhere Erkennungssicherheit hin. Ein geeigneter Grenzwert hängt von Bildqualität und Geschäftsprozess ab. Für automatische Schrankenentscheidungen empfiehlt sich zusätzlich ein Abgleich mit einer erlaubten Kennzeichenliste und eine projektspezifische Validierung.

5. Lokale REST-API

5.1 Basisadresse

```
http://127.0.0.1:5099
```

Standardmäßig ist die API nur lokal erreichbar. Remotezugriff sollte nur bewusst aktiviert und mit API-Token sowie geeigneten Netzwerkmaßnahmen geschützt werden.

5.2 Authentifizierung

Bei aktiviertem API-Token wird dieser in einem der folgenden Header übertragen:

```
X-CAR-READER-DLE-Token: <TOKEN>

oder

Authorization: Bearer <TOKEN>
```

5.3 Öffentliche Endpunkte

Methode	Endpunkt	Zweck
GET	/health	Runtime- und Lizenzstatus
POST	/api/single	Ein Bild erkennen
POST	/api/batch	Lokalen Bildordner verarbeiten
GET	/api/license/status	Lizenzstatus
GET	/api/runtime/config	Effektive öffentliche Konfiguration
GET	/api/runtime/capabilities	Verfügbare Funktionen und Profile
GET	/api/diagnostics/package	Supportpaket erzeugen
GET	/shutdown	Runtime kontrolliert beenden
POST	/api/rtsp/connect	RTSP-Kamera verbinden
GET	/api/rtsp/status	Kamerastatus lesen
GET	/api/rtsp/frame	Aktuelles Vorschaubild abrufen
POST	/api/rtsp/lane_roi	ROI setzen
POST	/api/rtsp/analyze	Aktuelles Bild oder Triggerereignis analysieren
POST	/api/rtsp/disconnect	Kamera(s) trennen
GET	/api/rtsp/cameras	Registrierte Kameras auflisten

5.4 HTTP-Statuscodes

Status	Bedeutung	Empfohlene Reaktion
200	Anfrage erfolgreich	Ergebnis auswerten
400	Ungültige Anfrage oder kein verwertbares Ergebnis	Request und Eingabedaten prüfen
401/403	Authentifizierung oder Zugriff nicht erlaubt	Token und Zugriffskonfiguration prüfen
409	Maximale Kamerazahl erreicht	Kamera trennen oder max_cameras anpassen
429	Für dieselbe Fahrspur läuft bereits eine Auswertung	Bei Livevideo Frame verwerfen; nicht sofort erneut senden
503	Verarbeitungskapazität ausgelastet	Kurz warten und mit Backoff wiederholen

6. Einzelbilder verarbeiten

6.1 Minimaler Request

```
curl -s -X POST "http://127.0.0.1:5099/api/single" \
 -F "image=@C:/Bilder/kennzeichen.jpg" \
 -F "plate_profile=international_latin"
```

6.2 Wichtige Request-Felder

Feld	Typ	Beschreibung
image	Datei, erforderlich	JPEG, PNG, BMP oder WebP
plate_profile	Text, optional	Erkennungsprofil; Standard aus Config
country_profile	Text, optional	Alternative Länderangabe
lane_id / camera_id	Text, optional	Stabile Kennung für Fahrspur-/Frame-Schutz
use_gap_reconstruction	Boolean, optional	Formatierung/Separatorrekonstruktion
use_gap_ranker	Boolean, optional	Erweiterte Separatorauswahl

6.3 Typische Antwort

```
{
 "ok": true,
 "text": "BMS1234",
 "formatted_text": "B-MS1234",
 "confidence": 0.9931,
 "plate_layout": "single_line",
 "line_count": 1,
 "separator_positions": [1],
 "country_profile": "INTERNATIONAL_LATIN",
 "profile": "international_latin",
 "lane_id": "lane-01"
}
```

6.4 Mehrere Fahrspuren

```
curl -s -X POST "http://127.0.0.1:5099/api/single" \
 -F "image=@lane-01.jpg" \
 -F "lane_id=lane-01" \
 -F "plate_profile=auto"
```

Verwenden Sie für jede Kamera oder Fahrspur immer dieselbe stabile Kennung. Dadurch kann die Runtime veraltete Folgeframes derselben Spur kontrolliert verwerfen, während andere Spuren parallel verarbeitet werden.

7. Batch-Verarbeitung

7.1 Beispiel

```
curl -s -X POST "http://127.0.0.1:5099/api/batch" \
 -H "Content-Type: application/json" \
 -d "{"folder_path":"D:/Bilder/Test","plate_profile":"auto"}"
```

7.2 Typische Optionen

Option	Beschreibung
folder_path	Lokaler Ordner auf dem Runtime-Rechner
plate_profile	Profil für alle Bilder
use_gap_reconstruction	Formatierte Kennzeichen rekonstruieren
use_gap_ranker	Erweiterte Separatorauswahl
save_non_matches	Nicht passende Ergebnisse exportieren, sofern aktiviert
export_char_boxes	Zeichenbox-Annotationen exportieren, sofern benötigt
allow_fast_path	Qualitätserhaltende Schnellpfade zulassen

Batch-Verarbeitung arbeitet auf lokalen Dateien. Für entfernte Systeme sollten Bilder über /api/single übertragen oder vorher auf den Runtime-Rechner kopiert werden.

8. RTSP- und Mehrkamera-Betrieb

8.1 Kamera verbinden

```
curl -s -X POST "http://127.0.0.1:5099/api/rtsp/connect" \
  -H "Content-Type: application/json" \
  -d "{"camera_id":"lane-01","profile_name":"Einfahrt 1","rtsp_url":"rtsp://kamera/stream"}"
```

8.2 Zugangsdaten getrennt übergeben

```
{
  "camera_id": "lane-01",
  "rtsp_url": "rtsp://kamera/stream",
  "username": "BENUTZER",
  "password": "PASSWORT"
}
```

Zugangsdaten sollten nicht in öffentlich einsehbare Skripte, Quelltexte oder Protokolle geschrieben werden.

8.3 Kamera analysieren

```
curl -s -X POST "http://127.0.0.1:5099/api/rtsp/analyze" \
  -H "Content-Type: application/json" \
  -d "{"camera_id":"lane-01","auto_trigger":true,"plate_profile":"international_latin"}"
```

8.4 Kameras auflisten und Status prüfen

```
curl -s "http://127.0.0.1:5099/api/rtsp/cameras"
curl -s "http://127.0.0.1:5099/api/rtsp/status?camera_id=lane-01"
```

8.5 ROI verwenden

Eine Region of Interest (ROI) begrenzt die relevante Fahrbahnfläche. Dadurch werden irrelevante Bildbereiche reduziert und die Live-Auswertung kann effizienter und stabiler arbeiten. Die ROI sollte die erwartete Kennzeichenposition vollständig abdecken und ausreichend Rand enthalten.

8.6 Ereignisgesteuerter Betrieb

Bei Livevideo sollte nicht jeder Videoframe vollständig erkannt werden. Empfohlen wird eine Vorprüfung auf Bewegung oder Kennzeichenereignisse. Während eine Spur bereits ausgewertet wird, können weitere Frames derselben Spur verworfen werden. Dadurch bleibt die Warteschlange aktuell und reagiert schneller auf neue Fahrzeuge.

9. Konfigurationsdatei

9.1 Grundregeln

- Vor Änderungen eine Sicherung der Config erstellen.
- JSON-Syntax muss gültig bleiben.
- Änderungen werden in der Regel nach einem Neustart der Runtime wirksam.
- Nur dokumentierte Optionen ändern.
- Nach Performanceänderungen Health-Endpunkt und Lasttest prüfen.

9.2 Hauptbereiche

Bereich	Zweck
recognition	Standardprofil, minimale Textlänge, Umlautdarstellung und Mehrfachkennzeichen
grouping	Separator- und Formatierungslogik
output	Optionale Detail-, Annotation- und Bildausgabe
input_limits	Zulässige Bildabmessungen und Pixelzahl
privacy	Speicherung, Kennzeichentext in Protokollen und Diagnosebilder
logging	Kundenprotokoll und Rotation
security	Lokaler oder freigegebener Netzwerkzugriff und Tokenpflicht
live_trigger	Aktivierung des Live-Triggers und maximale Kameraanzahl

9.3 recognition

Option	Standard/Beispiel	Beschreibung
plate_profile	international_latin	Standardprofil, wenn der Request kein Profil vorgibt
min_plate_length	2	Minimale akzeptierte Textlänge
preserve_umlauts	true	Umlaute in der Ausgabe erhalten
restore_known_umlaut_prefixes	true	Bekannte Präfixe lesefreundlich wiederherstellen
multi_plate.enabled	false	Mehrfachkennzeichen aktivieren
multi_plate.mode	single_best / all_visible	Bestes Kennzeichen oder alle sichtbaren Kennzeichen
multi_plate.max_plates	8	Maximale Anzahl zurückgegebener Kennzeichen
multi_plate.min_ocr_confidence	0.5	Mindest-Confidence für Mehrfachergebnisse

9.4 grouping

Option	Beschreibung
enabled	Formatierte Ausgabe und Separatorrekonstruktion aktivieren
use_ranker	Verbesserte Wahl plausibler Separatorpositionen
separator	Standardseparator, üblicherweise Bindestrich
multi_separator_enabled	Mehrere Separatoren zulassen, z. B. Niederlande
country_profile_formatting_enabled	Länderbezogene Darstellung anwenden

9.5 output

Option	Empfehlung
include_annotated_image	Nur aktivieren, wenn das Bild tatsächlich benötigt wird
include_plate_details	Für Integrationen meist true
include_image_payloads	Bei Bandbreiten- oder Datenschutzanforderungen false

9.6 privacy

Option	Empfohlener Standard
store_uploaded_images	false
store_annotated_images	false
log_plate_text	false
diagnostics_include_images	false
retention_days	Nach jeweiliger Richtlinie; Standard 7
customer_log_enabled	true
logging.level	INFO
max_file_size_mb	5
max_retained_files	5

9.7 security

Option	Beschreibung
allow_remote_clients	Remotezugriff bewusst aktivieren oder deaktivieren
require_api_token_for_remote	Für Remotezugriff immer true empfohlen

9.8 live_trigger

Option	Beschreibung
enabled	Live-Trigger aktivieren
mode	Triggerstrategie der Runtime
max_cameras	Maximal registrierbare RTSP-Kameras
idle_rearm_seconds	Zeit bis zur erneuten Bereitschaft nach einem Ereignis
duplicate_cooldown_seconds	Unterdrückt kurzzeitig wiederholte Ergebnisse
recent_frame_buffer_size	Anzahl kürzlich gehaltener Frames
partial_plate_reject_enabled	Unvollständige Kennzeichenereignisse reduzieren

Die weiteren Live-Trigger-Werte dienen der Feinabstimmung für konkrete Kameraszenen. Änderungen sollten anhand realer Streams getestet werden. Für typische Installationen sind die mitgelieferten Standardwerte ein guter Ausgangspunkt.

10. Performance und Presets

10.1 Wichtige Grundsätze

- Die Runtime verwaltet Parallelität, Ressourcenbegrenzung und Bildvorbereitung automatisch.
- Bei Livevideo die vollständige Erkennung ereignisgesteuert ausführen, nicht für jeden Videoframe.
- Eine passend gesetzte ROI reduziert irrelevante Bildbereiche und stabilisiert die Liveverarbeitung.
- Bei bekanntem Kennzeichenbestand ein passendes Länder- oder Layoutprofil statt auto verwenden.
- Annotationen, JPEG- und Base64-Bilddaten nur bei tatsächlichem Bedarf anfordern.

10.2 Systemdimensionierung

Für typische Installationen mit vier bis acht Kameras ist nicht die Videoframerate entscheidend, sondern die Zahl gleichzeitig auftretender Fahrzeugereignisse. Vier Kameras mit je 25 FPS erzeugen nicht automatisch 100 OCR-Aufträge pro Sekunde. Die vollständige Erkennung sollte nur bei einem geeigneten Ereignis ausgelöst werden.

Situation	Benötigter mittlerer Durchsatz
4 Ereignisse in 2 Sekunden	2 Erkennungen/s
4 Ereignisse in 1 Sekunde	4 Erkennungen/s
8 Ereignisse in 2 Sekunden	4 Erkennungen/s
8 Ereignisse in 4 Sekunden	2 Erkennungen/s

10.7 Lasttest bewerten

- successful entspricht der angefragten Zahl
- failed = 0
- keine HTTP-503-Antworten im normalen Zielbetrieb
- HTTP 429 kann bei bewusstem Frame-Dropping derselben Lane normal sein
- p95-Latenz innerhalb des Prozessziels
- Queue bleibt nicht dauerhaft voll
- Arbeitsspeicher bleibt ausreichend und das System lagert nicht stark aus

11. Datenschutz und Sicherheit

11.1 Datenschutzfreundliche Grundeinstellung

Die Standardkonfiguration kann hochgeladene und annotierte Bilder verwerfen, Kennzeichentext aus Kundenlogs heraushalten und Diagnosepakete ohne Bilder erzeugen. Betreiber sind dennoch für die rechtliche Zulässigkeit, Information Betroffener, Aufbewahrungsfristen und Zugriffskontrollen verantwortlich.

11.2 Empfohlene Maßnahmen

- API möglichst nur lokal oder in einem geschützten Netzwerk bereitstellen.
- Remotezugriff nur mit Token und Firewall-Regeln aktivieren.
- Kennzeichen nur speichern, wenn der Geschäftsprozess dies erfordert.
- Aufbewahrungszeiten begrenzen.
- Zugriffe und Administrationsrechte nach dem Minimalprinzip vergeben.
- RTSP-Zugangsdaten geschützt verwalten.
- Supportpakete vor Weitergabe gemäß Unternehmensrichtlinie behandeln.

11.3 Netzwerkbetrieb

Für externe Clients sollte die Runtime nicht ungeschützt direkt ins Internet gestellt werden. Empfohlen werden ein internes Netzwerk, VPN oder ein vorgelagerter Reverse Proxy mit TLS und Zugriffskontrolle.

12. Lizenz und Betrieb

12.1 Lizenzstatus prüfen

```
curl -s http://127.0.0.1:5099/api/license/status
```

Der Endpunkt liefert, ob die installierte Lizenz gültig ist und welche freigegebenen Funktionen verfügbar sind.

12.2 Hardware- oder Systemwechsel

Eine Lizenz kann an die lizenzierte Installation gebunden sein. Vor Hardwaretausch, Betriebssystem-Neuinstallation oder Migration sollte die Lizenzübertragung mit dem Anbieter abgestimmt werden.

12.3 Updates

- Vor Updates Konfiguration sichern.
- Releasehinweise des Kundenpakets beachten.
- Nach Update Health-, Lizenz- und Erkennungstest durchführen.
- Bei Performanceänderungen Preset erneut auf Zielhardware prüfen.

13. Fehlerbehandlung und Diagnose

13.1 Health-Endpunkt

```
curl -s http://127.0.0.1:5099/health
```

Wichtige öffentliche Felder sind status, license_ok, uptime_seconds, rtsp_camera_count und rtsp_connected_count.

13.2 Typische Fehler

Symptom	Mögliche Ursache	Maßnahme
Runtime nicht erreichbar	Dienst nicht gestartet oder falscher Port	Starttool/Dienst und Firewall prüfen
401/403	Token fehlt oder ist falsch	Header und Konfiguration prüfen
429 LANE_BUSY_FRAME_DROPPED	Für dieselbe Lane läuft bereits ein Job	Bei Livevideo normal; aktuellen Frame später senden
503 INFERENCE_QUEUE_FULL	Verarbeitungskapazität vorübergehend ausgelastet	Nach retry_after_seconds mit Backoff wiederholen oder Last reduzieren
RTSP nicht verbunden	URL, Zugangsdaten oder Netzwerkproblem	Stream mit Player prüfen, Netzwerk und Credentials prüfen
Hohe Latenz	Viele gleichzeitige Ereignisse, große Bilder oder auto-Profil	ROI, Bildgröße, Profil und Zielhardware prüfen
Hoher RAM-Verbrauch	Viele Kameras, große Bilder oder hohe Ereignisdichte	Last reduzieren oder RAM erhöhen

13.3 Supportpaket

```
curl -s http://127.0.0.1:5099/api/diagnostics/package
```

Das Supportpaket enthält kundenbezogene Betriebsinformationen zur Fehleranalyse. Die Aufnahme von Bildern richtet sich nach der Datenschutzkonfiguration.

14. Integrationsbeispiele

14.1 Python

```
import requests

with open("kennzeichen.jpg", "rb") as image_file:
    response = requests.post(
        "http://127.0.0.1:5099/api/single",
        files={"image": image_file},
        data={
            "plate_profile": "international_latin",
            "lane_id": "lane-01"
        },
        timeout=30
    )
response.raise_for_status()
result = response.json()
print(result.get("formatted_text"), result.get("confidence"))
```

14.2 C#

```
using var client = new HttpClient();
using var content = new MultipartFormDataContent();
content.Add(new ByteArrayContent(File.ReadAllBytes("kennzeichen.jpg")),
        "image", "kennzeichen.jpg");
content.Add(new StringContent("international_latin"), "plate_profile");
content.Add(new StringContent("lane-01"), "lane_id");

var response = await client.PostAsync(
    "http://127.0.0.1:5099/api/single", content);
response.EnsureSuccessStatusCode();
var json = await response.Content.ReadAsStringAsync();
```

14.3 JavaScript

```
const form = new FormData();
form.append("image", file);
form.append("plate_profile", "auto");
form.append("lane_id", "lane-01");

const response = await fetch("http://127.0.0.1:5099/api/single", {
  method: "POST",
  body: form
});
const result = await response.json();
```

14.4 Retry-Verhalten

Bei HTTP 503 sollte der Client mit kurzer, ansteigender Verzögerung erneut versuchen. Bei HTTP 429 und Liveframes sollte der verworfene alte Frame normalerweise nicht erneut gesendet werden; stattdessen wird der nächste aktuelle Frame verwendet.

15. Planung typischer Installationen

15.1 Eine Kamera

- Stabile Kameraposition, passende ROI und spezifisches Profil verwenden
- Spezifisches Profil verwenden
- ROI und Bildausschnitt optimieren
- Annotation nur für Bedienoberfläche aktivieren

15.2 Vier Kameras

- Gleichzeitige Ereignisse aller Kameras in einem Burst-Test prüfen
- Jede Kamera erhält stabile camera_id
- Ereignisgesteuerte Analyse verwenden
- Burst-Test mit vier gleichzeitigen Requests durchführen

15.3 Acht Kameras

- Zielhardware mit realistischen Acht-Kamera-Lastspitzen prüfen
- 4 x 2 nur bei ausreichenden CPU-Ressourcen und RAM testen

15.4 Bis zu 30 Kameras als Sonderfall

Eine große Zahl angebundener Kameras ist möglich, wenn vollständige Erkennungen nur ereignisgesteuert ausgelöst werden. Bei dauerhaft hoher Ereignisdichte sollten mehrere Runtime-Instanzen oder zusätzliche Systeme eingeplant werden.

15.5 Abnahmetest

1. Health- und Lizenzstatus prüfen.
2. Mindestens 100 repräsentative Bilder je Kamerasituation testen.
3. Vier-/Acht-Spur-Burst simulieren.
4. Mehrstündigen Kamerabetrieb prüfen.
5. p95-Latenz, Auslastungsverhalten, Fehlercodes und RAM dokumentieren.
6. Datenschutz- und Aufbewahrungseinstellungen freigeben.

16. Checklisten und FAQ

16.1 Schnellstart-Checkliste

- Runtime installiert und gestartet
- /health liefert status ok
- Lizenzstatus gültig
- Testbild über /api/single erfolgreich
- Passendes Profil ausgewählt
- Datenschutzwerte geprüft
- Kamera-IDs eindeutig
- Performancepreset auf Zielhardware getestet
- Client behandelt 429 und 503 korrekt
- Supportweg dokumentiert

16.2 Häufige Fragen

Muss die Runtime mit dem Internet verbunden sein?

Nein. Die Kennzeichenerkennung arbeitet lokal und offline. Netzwerkzugriff kann für Kamerastreams oder die Kommunikation mit der Kundenanwendung erforderlich sein.

Kann die Runtime mehrere Kameras parallel verarbeiten?

Ja. Mehrere Kameras können parallel betrieben werden. Die geeignete Systemgröße hängt von Bildauflösung, Ereignisdichte, Profil und Zielhardware ab.

Warum werden bei Livevideo Frames mit HTTP 429 verworfen?

Dadurch bleibt die Verarbeitung aktuell und veraltete Frames verdrängen keine neuen Fahrzeugereignisse.

Welches Profil ist am besten?

Bei einem klar bekannten Kennzeichenbestand ist ein spezifisches Länder- oder Layoutprofil oft am effizientesten. Bei gemischten Kennzeichen ist auto die bequemste Wahl.

Kann ich annotierte Bilder erhalten?

Ja, sofern die Bildausgabe aktiviert und angefordert wird. Für reine Backend-Integrationen sollten Annotationen aus Performance- und Datenschutzgründen deaktiviert bleiben.

Wie viele Kameras unterstützt ein Büro-PC?

Typische Installationen mit vier bis acht ereignisgesteuerten Kameras sind auf geeigneten modernen Büro-PCs möglich. Eine belastbare Aussage erfordert einen Test auf dem konkreten System.