

cs-imageindex

Handbuch

Fotos und Videos in eine durchsuchbare CSV indizieren - EXIF, GPS, Orte, Gesichter, Szenenbeschreibung, Duplikate

Version 0.4.0 · 13. Oktober 2026

Inhalt

1. Überblick	3
2. CSV-Spalten	3
2.1 Metadaten und Qualität	3
2.2 Gesichter und Szene	4
3. Unterstützte Formate	4
4. Verwendung	4
4.1 Optionen	5
4.2 Beispiele	5
5. Gesichtserkennung	5
6. Geocoding und Threads	6
7. Installation und Builds	6
8. Sicherheit und Datenschutz	6
9. Lizenz	6

1. Überblick

cs-imageindex durchsucht einen Ordner mit Fotos und Videos (rekursiv) und schreibt pro Mediendatei eine CSV-Zeile. Aus dem Index lassen sich Auswahlen bauen - Ordner mit Links auf passende Bilder -, ohne die Originale zu verändern oder zu duplizieren.

Empfohlene Ordnerstruktur:

```
/tank/images
/tank/images/data/*      (Originale)
/tank/images/index      (Suchindex, z.B. index.csv)
/tank/images/selections/* (z.B. selections/selection_1)
```

Ablauf:

1. Alle Bilder und Videos als napp-it CS Job indizieren (wiederaufnehmbar, bereits indizierte Dateien werden mit --resume übersprungen).
2. Aus dem Index eine Auswahl bauen (Ordner mit Links auf die passenden Bilder; Originale bleiben unverändert).

Das Tool ist ein einzelnes statisches Rust-Binary plus ein mitgeliefertes, minimales statisches LGPL-ffmpeg/ffprobe. Es muss keine Laufzeitumgebung installiert werden und es gibt keine Systembibliotheken als Abhängigkeit.

2. CSV-Spalten

2.1 Metadaten und Qualität

Spalte	Bedeutung
media_type	image / raw / heic / video
date_taken	EXIF DateTimeOriginal
gps_lat / gps_lon	EXIF-GPS in Dezimalgrad (leer, falls keine)
place	per Reverse-Geocoding ermittelter Ortsname (OpenStreetMap/Nominatim, best effort)
camera	EXIF Make + Model
resolution	Pixelmaße, z.B. 4032x3024
duration	Videodauer in Sekunden
orientation	landscape / portrait / square
flash	yes / no / leer
blur_score	Schärfe nach Laplace-Varianz (höher = schärfer, relative Skala)
phash	64-Bit Wahrnehmungs-Hash (hex) zur Erkennung ähnlicher Bilder
duplicate_group	gemeinsame Gruppennummer für Beinahe-Duplikate (Hamming-Distanz im Schwellwert)
is_screenshot	yes, wenn keine Kamera-Make/Model-EXIF vorhanden ist (Heuristik)

2.2 Gesichter und Szene

Spalte	Bedeutung
people	kommagetrennte Namen, abgeglichen mit Referenzfotos
unknown_faces	erkannte, aber nicht zugeordnete Gesichter
face_count	Anzahl erkannter Gesichter insgesamt
tags	Szenen-Tags des Vision-LLM
ocr_text	vom Vision-LLM gelesener Text im Foto
description	einsätzliche Szenenbeschreibung

3. Unterstützte Formate

- **Standbilder:** jpg/jpeg, png, bmp, tif/tiff (Erkennung am Inhalt).
- **RAW:** cr2, cr3, nef, arw, dng, orf, rw2, raf, pef, srw - in reinem Rust dekodiert, ohne externe Tools.
- **HEIC/HEIF:** Dekodierung über das mitgelieferte ffmpeg; EXIF wird direkt aus der Datei gelesen.
- **Video:** mp4, mov, avi, m4v, mkv - ein repräsentatives Bild durchläuft dieselbe Pipeline; Aufnahmedatum, Dauer und GPS (auch das ISO-6709-Tag von iPhone-.mov) stammen aus dem Container.

Fehlt das mitgelieferte ffmpeg, wird ffmpeg/ffprobe aus dem PATH verwendet. Gibt es keines von beiden, werden HEIC- und Videodateien mit einem Hinweis beim Start übersprungen; alles andere läuft normal.

4. Verwendung

```
cs-imageindex --folder /path/to/photos --out index.csv \
  --refdir /path/to/reference --models-dir /path/to/models \
  --legacy-cfg /path/to/cs-aihelp.cfg
```

4.1 Optionen

Option	Bedeutung
--folder	zu indizierender Ordner (rekursiv)
--out	Ausgabe-CSV
--refdir	reference/<Name>/*.jpg für den Gesichtsabgleich (weglassen = keine Erkennung)
--models-dir	Ordner mit yunet.onnx und sface.onnx (Standard: models neben dem Binary)
--config	eigene Standalone-Konfigurationsdatei (siehe --print-config-example)
--legacy-cfg	Fallback: endpoint2/model2/api_key2 aus einer napp-it cs-aihelp.cfg
--provider / --endpoint / --model / --api-key	Vision-Provider-Einstellungen
--ollama / --ollama-model	lokaler Ollama-Endpoint und Modell (Standard llama3.2-vision)
--no-vision	Szenenbeschreibung überspringen (nur Ort + Gesichter)
--no-geocode	kein Netzwerkaufruf an Nominatim
--no-dedup	Duplikatgruppierung überspringen (spart einen $O(n^2)$ -Durchlauf)
--dedup-threshold N	Hamming-Schwelle 0-64, niedriger = strenger (Standard 6)
--threads N	Arbeits-Threads, 0 = automatisch (bis zu 4)
--resume	Dateien aus einer vorhandenen --out-CSV überspringen und neue Zeilen anhängen
--print-config-example	Beispiel-Konfigurationsdatei ausgeben

Priorität der Vision-Einstellungen: CLI-Optionen, dann die Umgebungsvariablen CS_IMAGEINDEX_PROVIDER / _ENDPOINT / _MODEL / _API_KEY / _MAX_TOKENS, dann --config, dann --legacy-cfg. Wird nichts aufgelöst, entfällt die Vision stillschweigend (wie --no-vision).

4.2 Beispiele

```
# Lokales Ollama statt Cloud-Provider
cs-imageindex --folder ./photos --out index.csv \
  --ollama http://127.0.0.1:11434 --ollama-model llama3.2-vision

# Gar keine Netzwerkaufufe (nur EXIF + Gesichter)
cs-imageindex --folder ./photos --out index.csv --no-vision --no-geocode

# Unterbrochenen Lauf fortsetzen
cs-imageindex --folder ./photos --out index.csv --resume

# Strengere oder keine Duplikatgruppierung
cs-imageindex --folder ./photos --out index.csv --dedup-threshold 2
cs-imageindex --folder ./photos --out index.csv --no-dedup
```

5. Gesichtserkennung

Pro Person ein oder mehrere Referenzfotos unter <refdir>/<Name>/*.jpg ablegen. Erkennung (YuNet) und Wiedererkennung (SFace) sind ONNX-Modelle, die mit der reinen Rust-Engine tract laufen - ohne OpenCV und ohne onnxruntime. Die Mathematik für Erkennung und Ausrichtung ist ein Neuport des OpenCV-C++-Codes. Beide Modelle liegen aus dem OpenCV Zoo in models/ (MIT bzw. Apache 2.0, Lizenzdateien liegen bei).

6. Geocoding und Threads

Das Reverse-Geocoding nutzt die öffentliche Nominatim-API und hält deren Nutzungsregeln ein: etwa eine Anfrage pro Sekunde, ein sprechender User-Agent und ein prozessinterner Cache mit ca. 1 km Raster. Das Geocoding ist unabhängig von der Thread-Zahl global begrenzt. Mit `--no-geocode` lässt es sich abschalten.

Die Dateien werden auf einem kleinen Worker-Pool verarbeitet (`--threads`). Vision- und Geocode-Aufrufe werden sicher geteilt; jeder Thread hat eine eigene Vision-API-Verbindung.

7. Installation und Builds

Release-Archive folgen der napp-it-CS-Konvention `<asset>-<platform>.<arch>.tar.gz` (z.B. `cs-imageindex-linux.amd64.tar.gz`) und werden in napp-it CS unter About > Download cs-tools nach `data/cs_server/tools/cs-imageindex/<platform>.<arch>/` installiert.

GitHub Actions baut Linux, Windows und macOS (arm64 + x86_64) bei jedem v*-Tag. illumos/OmniOS wird von Hand mit `illumos/cs-imageindex_omnios_1a.sh` gebaut (auch in cs-scripts) und in dasselbe Release hochgeladen; das Binary ist vollständig eigenständig. Tests: `cargo test`.

8. Sicherheit und Datenschutz

- Bei einem Cloud-Provider wird jedes Foto als Base64-JPEG an den konfigurierten Endpoint gesendet - private Bilder verlassen den Rechner. Für rein lokale Verarbeitung Ollama (`--ollama`) oder `--no-vision` verwenden.
- `--config` oder `CS_IMAGEINDEX_API_KEY` bevorzugen; `--api-key` ist in der Prozessliste sichtbar.
- Verzeichnis-Symlinks werden nicht verfolgt, die Suche ist auf 64 Ebenen begrenzt; Datei-Symlinks werden als Blätter indiziert.
- Zellwerte, die mit `=`, `+`, `-` oder `@` beginnen, werden unverändert geschrieben (die Spalte file ist ein Pfad). CSV im Texteditor oder Datenwerkzeug öffnen, nicht per Doppelklick in einer Tabellenkalkulation.

9. Lizenz

BSD 2-Clause, wie die anderen Standalone-cs-tools. FFmpeg liegt unter der LGPL bei (siehe `LICENSE-ffmpeg.txt`).