KI-gestützte Erkennung und Entfernung ringförmiger Flat-Artefakte („Donuts“) in astronomischen Bildern (FITS)[1]

Ausgangslage / Problem

In Deep-Sky-Aufnahmen treten durch Staub/Tröpfchen im Strahlengang und unpassende Flats ringförmige Artefakte („Donuts“) auf. Diese stören visuell, verschlechtern Hintergrund-Uniformität und können Photometrie/Detektion verfälschen. Reine Bildverarbeitung (Hough, DoG, Inpainting) hilft, ist aber empfindlich gegenüber Parameterwahl und Feldvariation.

Eine weitere Ursache sind Reflexionen im Strahlengang, insbesondere von Netwon-Teleskopen. Diese entstehen vor alle, wenn ein lichtstarkes Objekt (etwa Vollmond) den Himmelshintergrund stark aufhellt und sich das fotografierte Objekt in der Nähe des Mondes befindet.

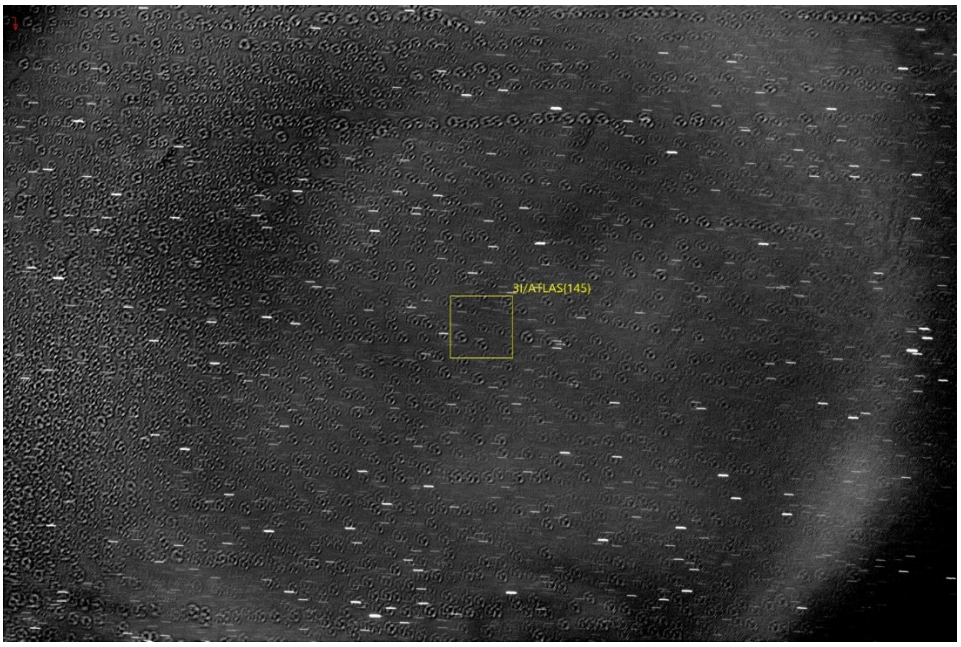

Abbildung 1: Donuts/Kringel in einer astronomischen Aufnahme. Hier der Komet 3I/Atlas.

Bei hellem, außerhalb des Bildfelds liegendem Licht (z. B. Vollmond in Bildnähe) wird das Innere des Tubus, der Fangspiegelhalter, die Auszugshülse oder blanke Schrauben „angeleuchtet“. Dieses Streulicht streut/ reflektiert bis zum Sensor, hebt den Himmelshintergrund an („veiling glare“) und verursacht Gradienten, Bögen oder schwache „Geisterbilder“. Newtons sind dafür anfällig, weil der offene Tubus und

[1] **Anmerkung:** Diese Projektbeschreibung wurde mit ChatGPT erzeugt. Sie beruht auf verschiedenen Experimenten, die im Rahmen des Experimentierens mit Donut-Aufnahmen durchgeführt wurden.

Bauteile wie Spinne/Fokussierer seitlich einfallendes Licht leicht ins System lassen. Zusätzliche Reflexionen entstehen an Korrektoren und Filtern (inkl. Rückreflex zwischen Filter und Sensorfenster), was sich als Halos um helle Sterne äußern kann.

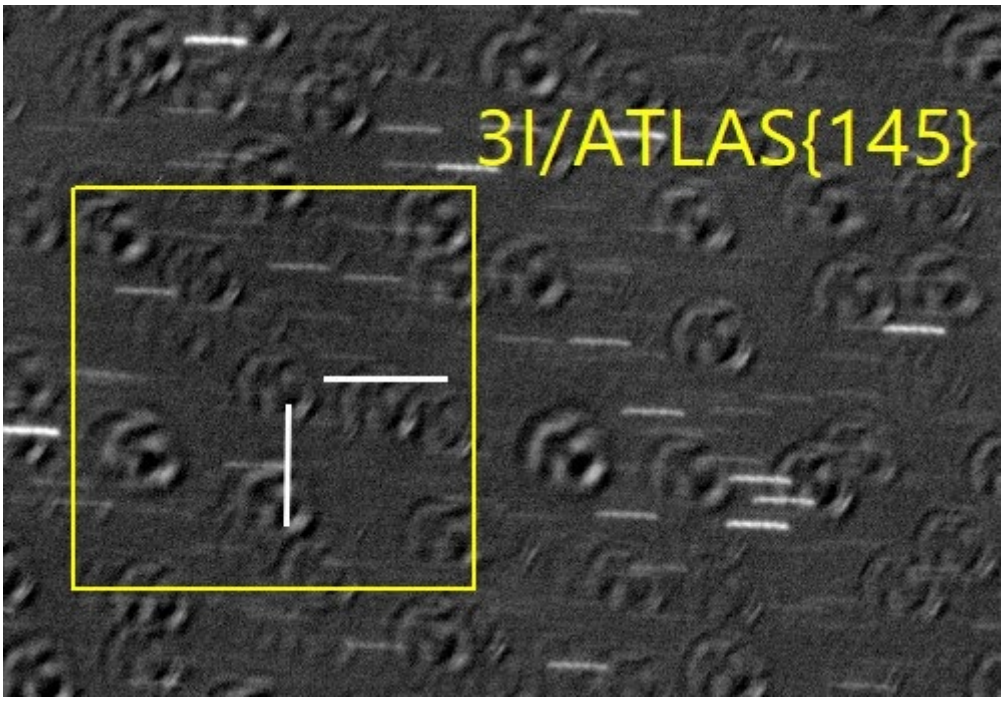

Abbildung 2: Donuts/Kringel vergrößert aus Abbildung 1.

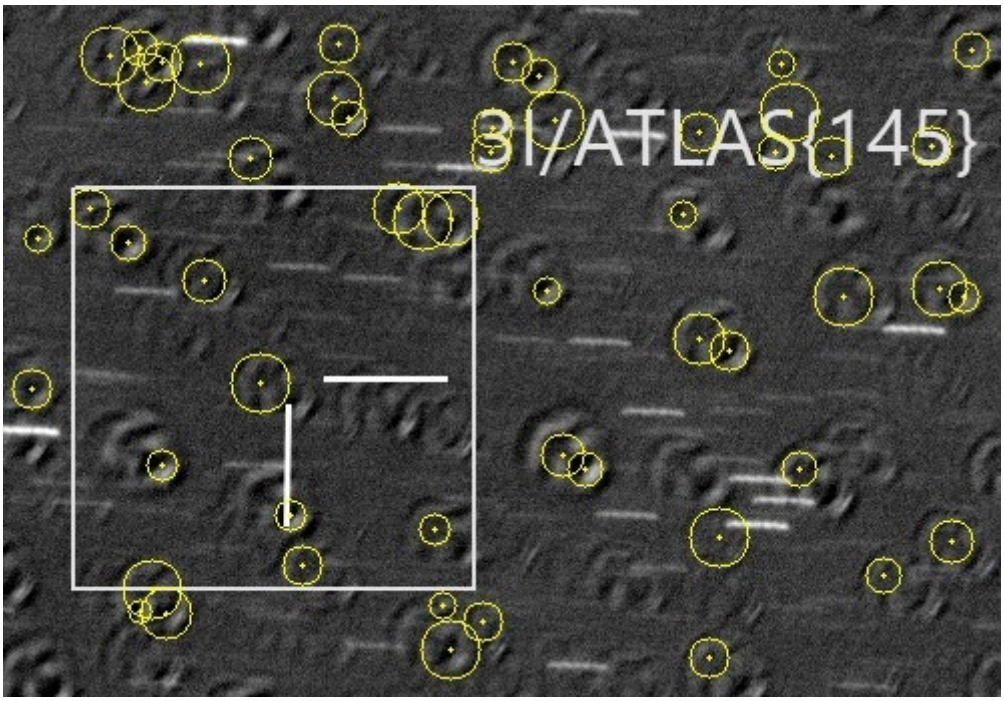

Abbildung 3: Donuts/Kringel mit KI markiert aus Abbildung 2.

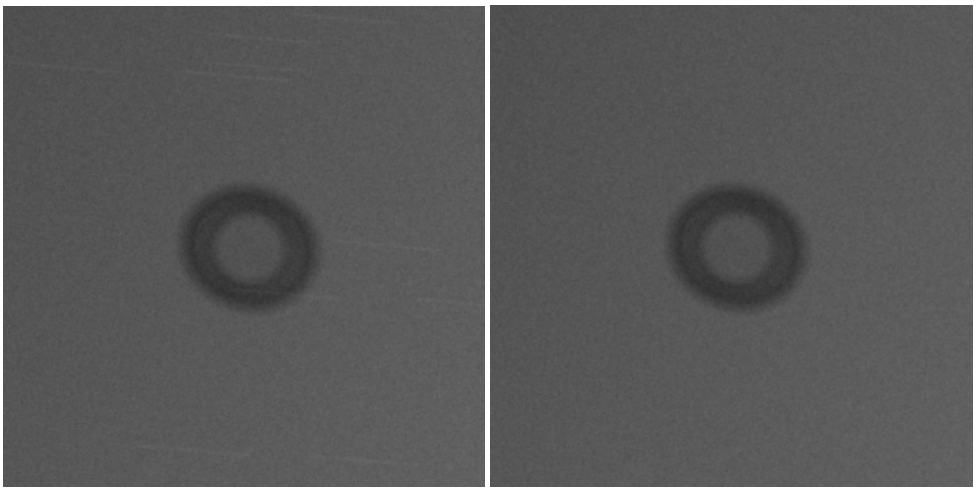

Abbildung 4: Synthetisch erzeugte Donuts.

Abbildung 5: Newton-Teleskop. Deutlich erkennbar, dass Licht seitlich in den Hauptspiegel eintreten kann.

Ziel(e)

Ziel ist eine robuste, photometrie-schonende **KI-Lösung**.

1. **Automatische Segmentierung** der Donuts (Pixel-genaue Maske) in Luminanz- und RGB-FITS.
2. **Artefaktentfernung** mit minimaler Beeinflussung echter Objekte (Sterne, Galaxien, Kometen).
3. **Photometrische Integrität** nachweisen (z. B. ΔMag < 0,02–0,05 für Sterne bis SNR-Grenze).
4. Laufzeit-taugliche **Inference** (≤ 1–2 s pro 4–20 MP-Frame auf GPU/CPU).
5. Saubere **Pipeline** inkl. Batch-Verarbeitung und CLI/Notebook-Demo.

Abgrenzung / Nicht-Ziele

- Kein Re-Calibration-Workflow (Flats) – Fokus ist rein bildbasiertes Postprocessing.
- Sterne, Kometenkerne, Galaxienarme dürfen **nicht** „weggeglättet“ werden.

Daten & Datensätze

- **Primärdaten:** FITS-Lights (mono/RGB) mit und ohne Donuts – ideal aus verschiedenen Setups (Pixelmaß, Seeing, F/Ratio).
- **Ground-Truth-Strategien:**
    - **Synthetic-Injection:** „Saubere“ Bilder + synthetisch generierte Donuts (Ringe mit realistischen Radien, Kontrasten, Unregelmäßigkeiten) ⇒ perfekter Ground-Truth für PSNR/SSIM/ΔPhot.
    - **Paare mit/ohne Artefakte** (selber Frame mit/ohne passendem Flat) – falls verfügbar.
    - **Pseudo-GT:** Klassischer CV-Baseline (z. B. DoG+Hough+Inpainting) als schwaches Label zur Vorannotation (für aktives Lernen/Refinement).
- **Annotation:** Binäre Masken (Donut/kein Donut), optional Klassentrennung (Ring, Ring-Kern). Tools: z. B. cvat, labelme.
- **Splits:** Stratifiziert nach Kamera/Pixelmaß/Seeing (Train/Val/Test z. B. 70/15/15). Kein Leak zwischen Sessions!

Methodische Ansätze (Teams wählen 1–2 Pfade + Baselines)

Baseline 0 (klassisch, geliefert): DoG/Hough-Detektion + OpenCV-Inpainting (Telea/Navier-Stokes) als Referenz.

A) Segmentierung + Inpainting (2-stufig):

- **U-Net / ResU-Net / Attention U-Net** für Donut-Masken (BCE+Dice-Loss, Focal-Loss).
- Inpainting-Schritt: klassisch (Telea) **oder** lernbasiert (Partial Convolutions / Gated Convolutions).
  B) End-to-End Restoration:
- **U-Net-Denoiser** (Residual Learning): Input=Artefaktbild, Output=Clean; Verlust: L1 + SSIM + Perceptual (VGG-Features) + Photometry-Penalty.
  C) Generative Verfahren (Stretch-Goal):
- **Conditional GAN** (pix2pix-ähnlich) oder leichtgewichtiges **Diffusions-Inpainting** mit Maske.
  D) Un/selbstüberwacht:
- **Denoising Autoencoder** / **Noise2Void/Noise2Self**-Varianten (wenn GT knapp) – kombiniert mit Donut-Masken.

Verluste & Domänenspezifische Constraints

- **Segmentation:** BCE + Dice (oder Tversky/Focal), class-balance beachten.
- **Restoration:** L1/L2 + SSIM, **Photometrie-Loss** (z. B. Differenz der PSF-integrierten Sternflüsse vor/nach außerhalb der Donut-Maske), **Star-PSF-Loss** (FWHM-Stabilität).
- **Mask-Aware Training:** In Loss nur Bereiche innerhalb/um die Donuts stärker gewichten.

Metriken (technisch & astronomisch)

- **PSNR/SSIM** gegenüber Ground-Truth.
- **Photometrische Abweichung**: ΔMag auf Referenzsternen (Aperture Photometry) – Median/95-Perzentil; Ziel < 0,03–0,05 mag.
- **FWHM-Änderung** Sterne (%).
- **Hintergrund-Uniformität**: σ-BG und Gradienten-Maß (vor/nach).
- **False-Positive-Rate**: Anteil „wegretuschierter“ echter Objekte.
- **Runtime/Memory**.

Pipeline / Deliverables

1. **Datenaufbereitung** (FITS I/O, 16-bit/float, Header-Erhalt).
2. **Synthetic-Donut-Generator** (Parameter: Radius 3–30 px, elliptisch, unregelmäßige Kanten, varying contrast, Feldverzerrungen).

3. **Training-Code** (PyTorch/JAX/TensorFlow) mit Konfig (YAML), Logging (TensorBoard/W&B).
4. **Baseline-Vergleich** (bereitgestellte CV-Pipeline).
5. **CLI-Tool** & **Notebook-Demo**: donut_clean_fits --input <file/dir> --method <model/baseline> --save-mask.
6. **Bericht (8–12 Seiten)**: Datendesign, Modell, Experimente, Metriken, Fehleranalyse, Limitationen.
7. **Kurzes Video (≤5 min)**: Ergebnisvergleich „Vorher/Nachher“, Photometrie-Plots.

Zeitplan (Vorschlag, 6–8 Wochen)

- **W1:** Domäne, Datenaufnahme, Baseline lauffähig, Metriken definieren.
- **W2–3:** Synth-Generator & Labeling, erstes U-Net, erste Resultate.
- **W4–5:** Hyperparameter/Augmentierung (Rot/Flip, Helligkeit, künstl. Gradienten), Photometrie-Loss integrieren.
- **W6:** Benchmarking vs. Baseline, Cross-Setup-Generalisation.
- **W7–8:** Hard-Cases, Fehleranalyse, Bericht/Video, Packaging.

Tooling / Infrastruktur

- **Datenformat:** FITS mono/RGB (float32 bevorzugt), Header erhalten.
- **Libs:** astropy, numpy, opencv, photutils (Aperture-Photometrie), PyTorch, lightning/keras, wandb/tensorboard.
- **Reproduzierbarkeit:** Seeds, Versionierung (git), requirements.txt/environment.yml.

Risiken & Gegenmaßnahmen

- **Kein echter Ground-Truth:** Synth-Injection + Photometrie-Metriken.
- **Over-smoothing / Objektverlust:** Star-Mask-Schutz, Mask-Aware-Loss, manuelle Vis-Checks.
- **Domänenwechsel (andere Kameras):** Augmentierungen (Pixelmaß, Seeing), Test auf fremden Setups.

Bewertung (Rubrik, 100 P)

- Funktionsfähigkeit & Qualität (Metriken ggü. Baseline) – 35 P
- Modell & Methodik (Begründung, Loss-Design) – 20 P
- Daten/GT-Strategie & Synth-Generator – 15 P
- Photometrie-Nachweis & Analysen – 15 P
- Code-Qualität, Repro, Doku & Demo – 15 P

Startpaket

- **Baseline-Skripte** (DoG/Hough + Inpainting, FITS-I/O) – als Vergleich und zum Label-Bootstrapping.
- Beispiel-Frames mit starkem/mäßigem Donut-Befall.
- Grundgerüst für **Synthetic-Donut-Generator**.