



Resampling Detection

Kirchner's Fast Detection Method - Forensische Analyse

Dominik Barbist, Lukas Egger

Agenda

- ① **Fall-Erinnerung** - Kartoffel-Contest Betrug
- ② **Interpolations-Methoden** - Einfache vs. fortgeschrittene Algorithmen
- ③ **Kirchner's Methode [2]** - Das "Rezept" zur Detektion
- ④ **Praktische Anwendung** - Fallauflösung mit Ergebnissen
- ⑤ **Grenzen & Lessons Learned** - Was funktioniert (nicht)?

Der Fall: Kartoffel-Contest Betrug

Das Problem:

- Online-Wettbewerb: Größte Kartoffel gewinnt
- Maßband als Größenreferenz
- Verdacht auf digitale Manipulation

Betrugsmethode:

- Kartoffel vergrößern
- Maßband verkleinern
- Proportionen bleiben stimmig

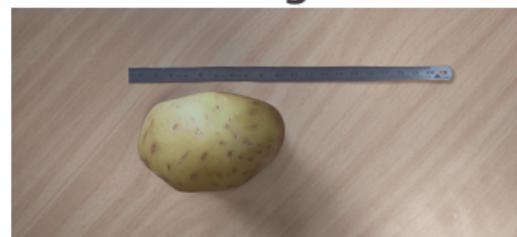
Die Herausforderung:

- Visuelle Inspektion **reicht nicht**
- Automatische Detektion **erforderlich**

Das echte Original:



Verdächtiges Bild:



Interpolations-Methoden: Einfache Algorithmen

Nearest Neighbor Mechanismus

50	80	120	150
70	(1.3, 1.7) 100	140	170
90	130 Neuer Pixel	160	190
110	150	180	200

(0,3) (1,3) (2,3) (3,3)

Berechnung:

1. Nächster zu (1.3, 1.7)
→ (1, 2)
2. Wert kopieren: 130

Ergebnis: 130

Bilinear Mechanismus

50	80	120	150
70	(1.3, 1.7) 100	140	170
90	130 Neuer Pixel	160	190
110	150	180	200

(0,3) (1,3) (2,3) (3,3)

Berechnung:

Gewichte:
 $w_1 = 0.21 \times 100 = 21.0$
 $w_2 = 0.09 \times 140 = 12.6$
 $w_3 = 0.49 \times 130 = 63.7$
 $w_4 = 0.21 \times 160 = 33.6$

Summe: 130.9

Blockige Kanten, keine Glättung^a

^a Bild mit Hilfe von KI erstellt

Glatte Übergänge, leichte Unschärfe^a

^a Bild mit Hilfe von KI erstellt

Detektierbarkeit

Einfach zu detektieren - erzeugen starke periodische Muster

Interpolations-Methoden: Fortg. Algorithmen

Bicubic Mechanismus

50	80	120	150
70	(3,3, 1,7) 100 <i>Neuer Pixel</i>	140	170
90	130	160	190
110	150	180	200

(0,3) (1,3) (2,3) (3,3)

Berechnung:

1. 16 Pixel sammeln (4×4)
2. Kubische Polynome:
 $f(x) = ax^3 + bx^2 + cx + d$
3. X-Interpolation
4. Y-Interpolation

Ergebnis: 142.7

Lanczos Mechanismus

50	80	120	150
70	(1,8, 1,7) 100 <i>Neuer Pixel</i>	140	170
90	130	160	190
110	150	180	200

(0,3) (1,3) (2,3) (3,3)

Berechnung:

1. Lanczos-Kernel:
 $L(x) = \text{sinc}(x) \times \text{sinc}(x/2)$
2. Gewichte für 4×4 Pixel
3. Gewichtete Summe
4. Normalisierung

Ergebnis: 140.1

Sehr glatt, hochwertige Qualität^a

Schärfste Details, komplexer Algorithmus^a

^a Bild mit Hilfe von KI erstellt

^a Bild mit Hilfe von KI erstellt

Detektierbarkeit

Schwieriger zu detektieren - subtilere periodische Muster

Das "Rezept": Kirchner's Fast Detection (2008)

Warum Kirchner's Methode?

- 40x schneller als Popescu & Farid [4]
- Keine komplexe EM-Iteration notwendig
- Feste Filter-Koeffizienten
- Vergleichbare Genauigkeit

Grundprinzip:

- ① **Input:** Verdächtiges Bild
- ② **Prozess:** Suche nach periodischen Interpolations-Artefakten
- ③ **Output:** Klassifikation als "Resampled" oder "Original"

Schritt 1: Linear Prediction Filter

Filter-Koeffizienten (fest):

$$\alpha^* = \begin{bmatrix} -0.25 & 0.50 & -0.25 \\ 0.50 & 0 & 0.50 \\ -0.25 & 0.50 & -0.25 \end{bmatrix}$$

Prediction Error:

$$e(i,j) = p(i,j) - \sum_{k,l} \alpha_{k,l}^* \cdot p(i+k, j+l)$$

Was passiert hier?

- Jeder Pixel wird durch Nachbarn vorhergesagt

Warum feste Koeffizienten?

- Interpolation erzeugt periodische Artefakte *unabhängig* von den exakten Koeffizienten
- Massive Beschleunigung(im vergleich zu EM)

Schritt 1: Linear Prediction Filter - How it works¹

NATURAL IMAGE ("Random" Werte)

142	89	187	103	156	91
178	125	84	172	96	138
161	112	189	77	145	168
94	159	123	188	81	147
175	106	134	162	98	181
119	153	87	174	129	196

Kirchner Filter



Prediction Errors

-43.2	+67.8	+15.1	-29.7
+52.4	+81.9	-48.5	+37.2
+24.6	-38.1	+72.3	-59.8
-41.7	+19.5	+43.2	-35.4

- Vorhersagefehler wird für jedes Pixel berechnet
- Periodische Muster erzeugen starke Korrelationen

RESAMPLED IMAGE (Linear interpolation)

100	110	120	130	140	150
105	115	125	135	145	155
110	120	130	140	150	160
115	125	135	145	155	165
120	130	140	150	160	170
125	135	145	155	165	175

Kirchner Filter



Prediction Errors

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

Filter wird auf jedes Pixel angewendet²

¹ Stark vereinfacht, um Konzept zu verdeutlichen

² Bild wurde selbst erstellt (war schneller als die KI-Bilder zuvor...)

Schritt 1: Linear Prediction Filter - Visualisierung



Original Prediction Error



Resampled Prediction Error

Schritt 1: Linear Prediction Filter - Visualisierung



Original Prediction Error



Resampled Prediction Error

Schritt 2: P-Map Generation

Contrast Function:

$$p(i,j) = \lambda \cdot \exp\left(-\frac{|\mathbf{e}(i,j)|^\tau}{\sigma}\right)$$

Parameter:

- $\lambda = 1$ (Normalisierung)
- $\sigma = 1$ (Schwellwert)
- $\tau = 2$ (Kontrast-Exponent)

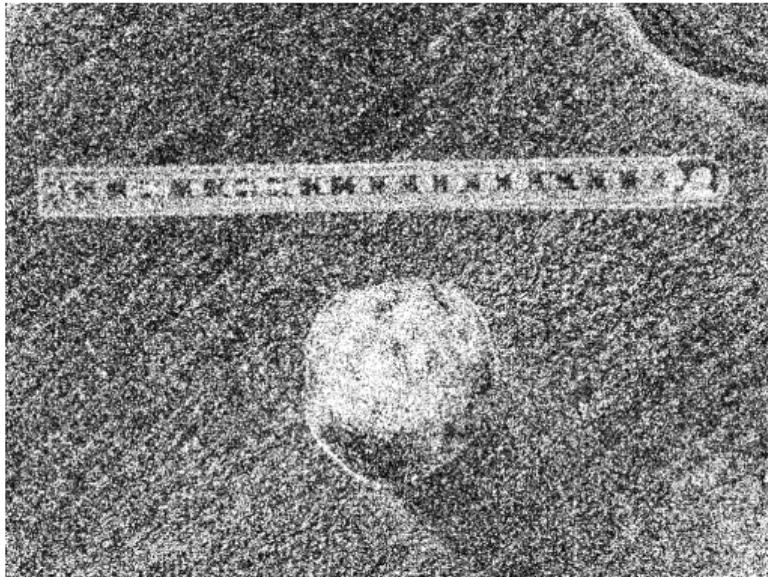
Was macht die P-Map?

- Kleine Errors → Hohe P-Map Werte
- Große Errors → Niedrige P-Map Werte
- Verstärkt periodische Muster

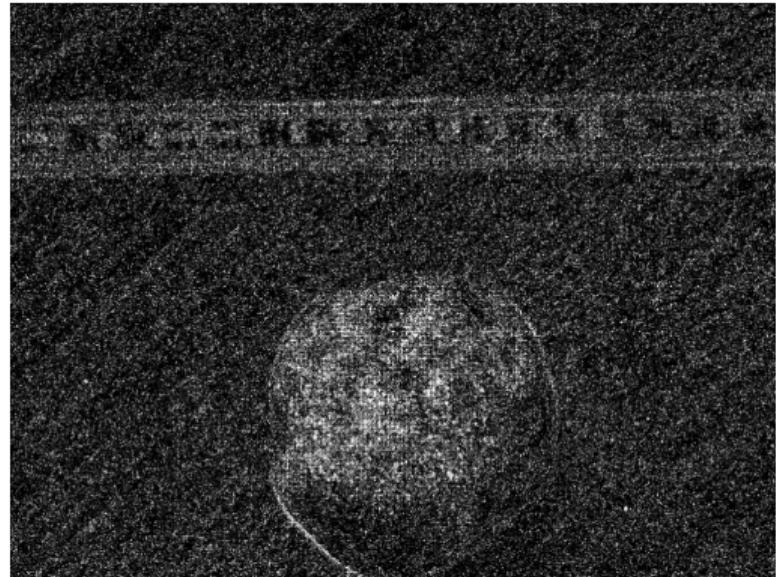
Ergebnis:

- Interpolierte Bereiche zeigen periodische P-Map
- Echte Bereiche zeigen zufällige P-Map

Schritt 2: P-Map Generation - Visualisierung

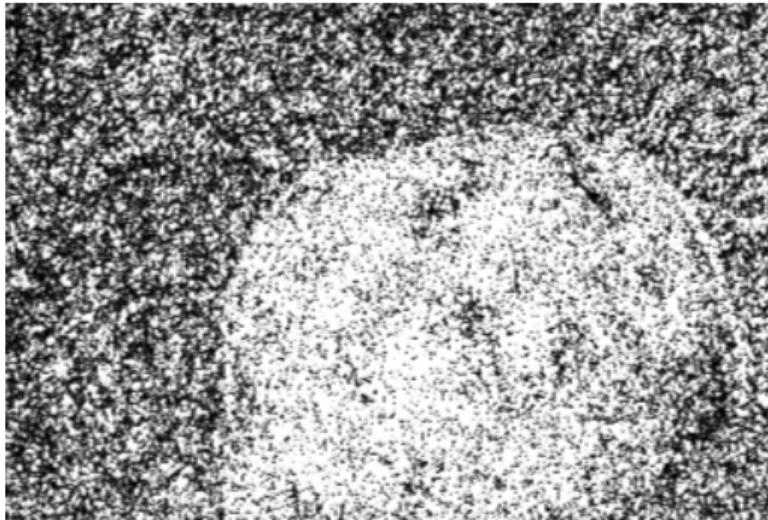


Original P-Map (Zufällig)

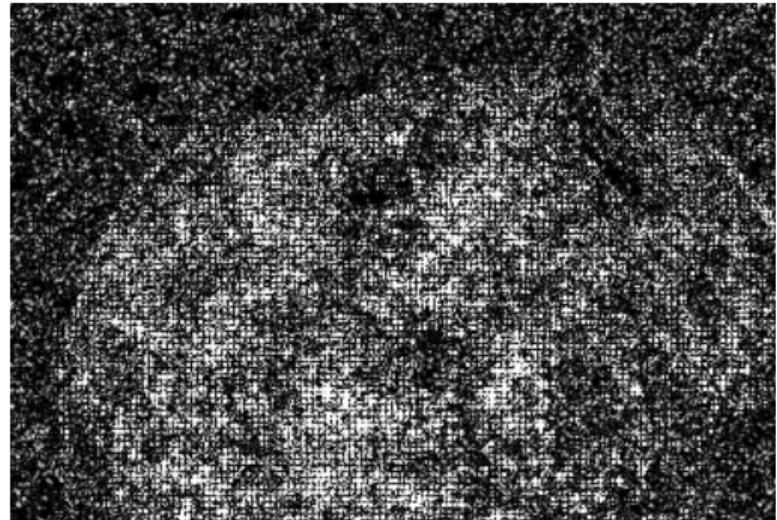


Resampled P-Map (Periodisch)

Schritt 2: P-Map Generation - Visualisierung



Original P-Map (Zufällig)



Resampled P-Map (Periodisch)

Schritt 3: Spektralanalyse & Peak Detection

Fourier Transform:

$$P_f = \text{DFT}(p)$$

Cumulative Periodogram:

$$C(f) = \frac{\sum_{0 < f' \leq f} |P(f')|^2}{\sum_{0 < f' \leq b} |P(f')|^2}$$

Decision Criterion:

$$\delta' = \max_f |\nabla C(f)|$$

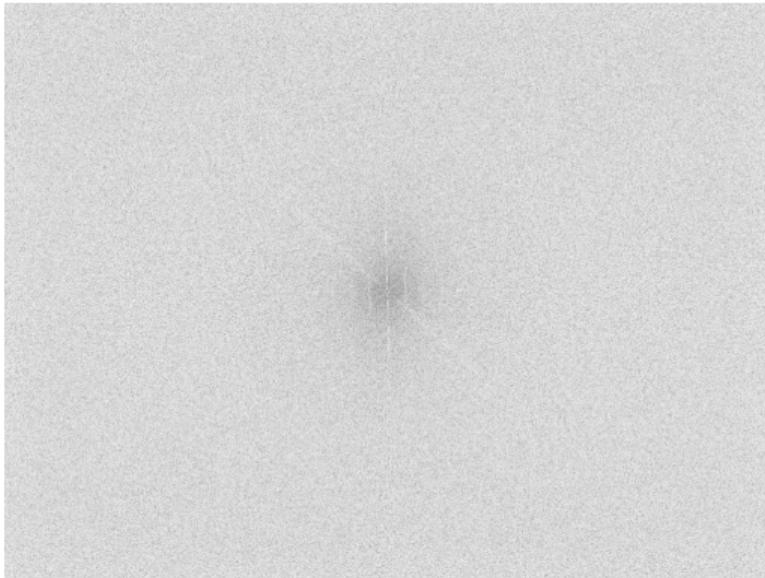
Peak Detection Logic:

- Periodische P-Map → Scharfe Peaks im Spektrum
- Zufällige P-Map → Gleichmäßiges Spektrum

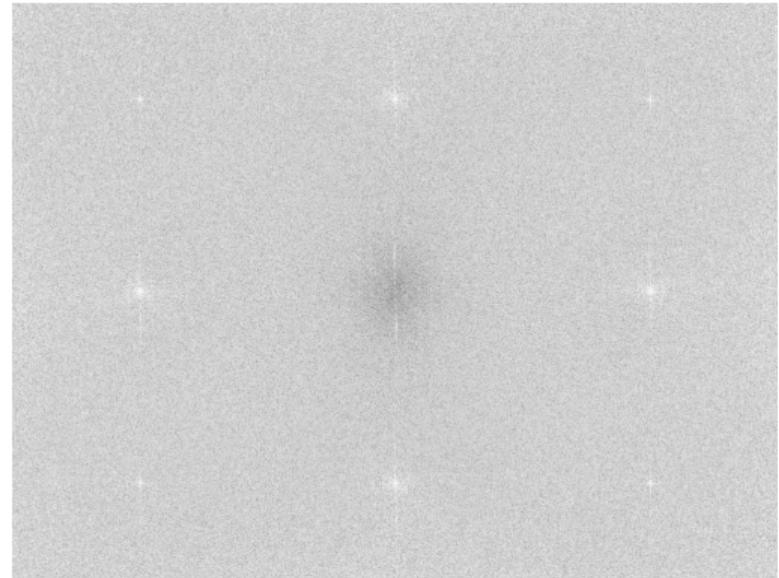
Automatische Detektion:

- Maximum des Gradienten im Cumulative Periodogram
- Schwellwert δ'_T für Klassifikation
- $\delta' > \delta'_T$ → Resampled
- $\delta' \leq \delta'_T$ → Original

Schritt 3: Spektralanalyse - Visualisierung

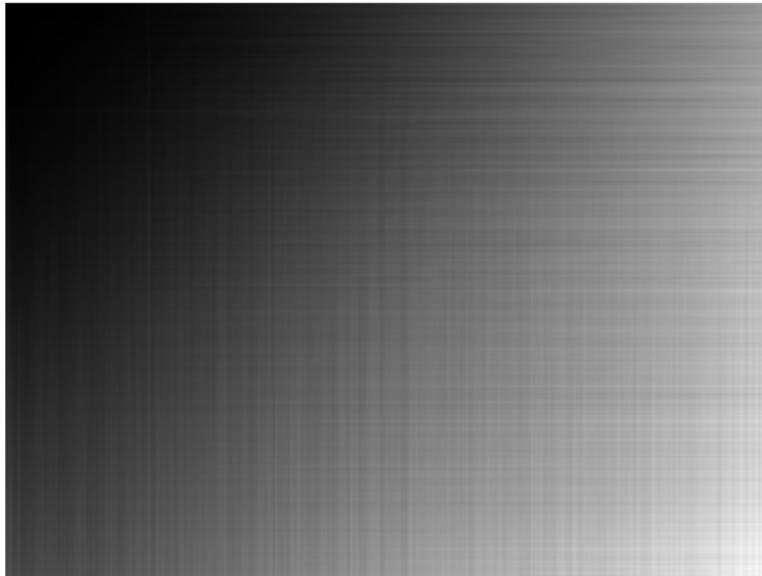


Original Spektrum (Zufällig)

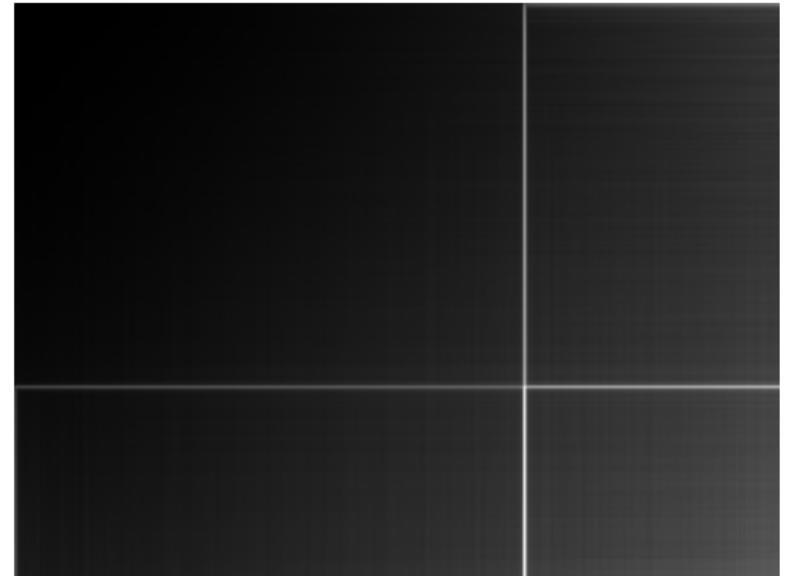


Resampled Spektrum (Periodisch)

Schritt 3: Decision Criterion - Visualisierung

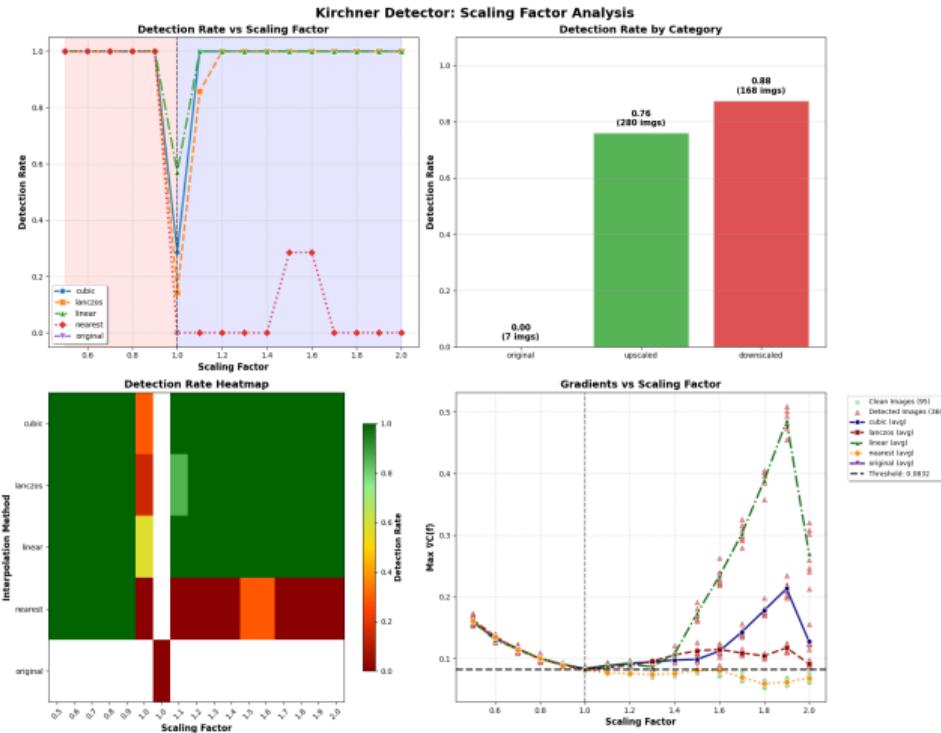


Original Gradienten Map("Zufällig")



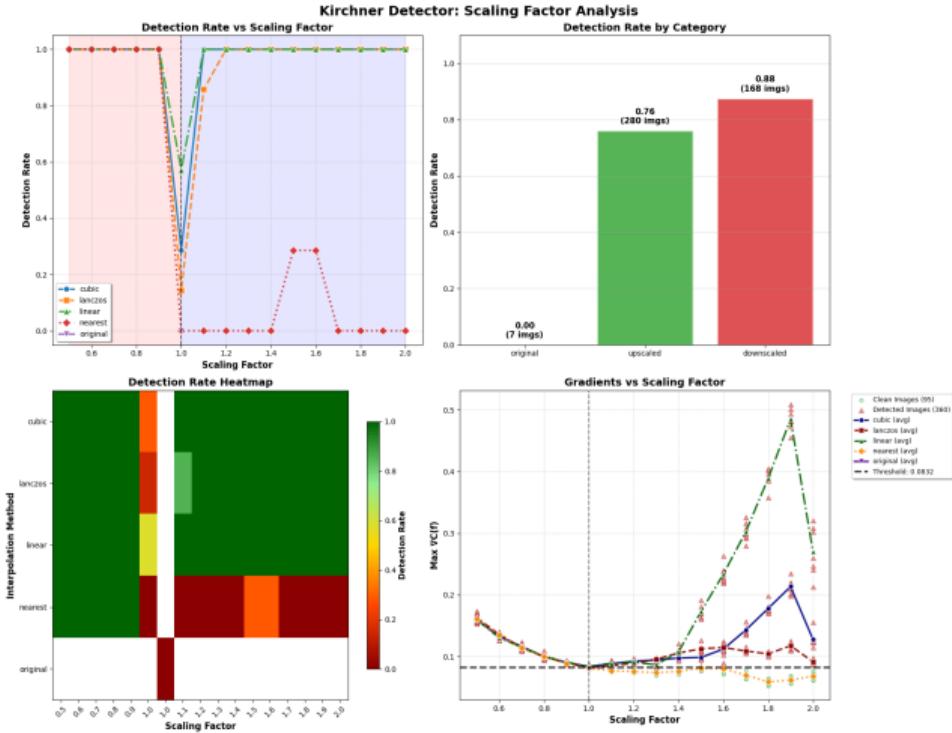
Resampled Gradienten Map("Periodisch")

Fallauflösung: Analyseergebnisse



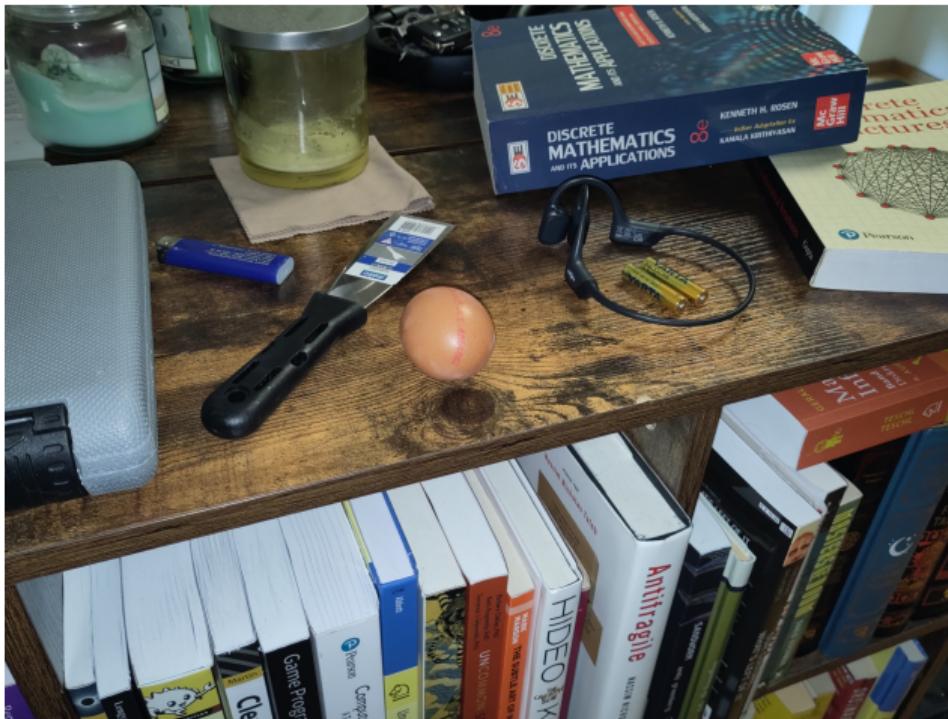
- 7 Originalbilder
- 16 Scalierungs faktoren
- 4 Interpolationsmethoden
- In Summe 455 Bilder
- Detection Rate von > 75%

Fallauflösung: Analyseergebnisse(Ausgewählte Daten)



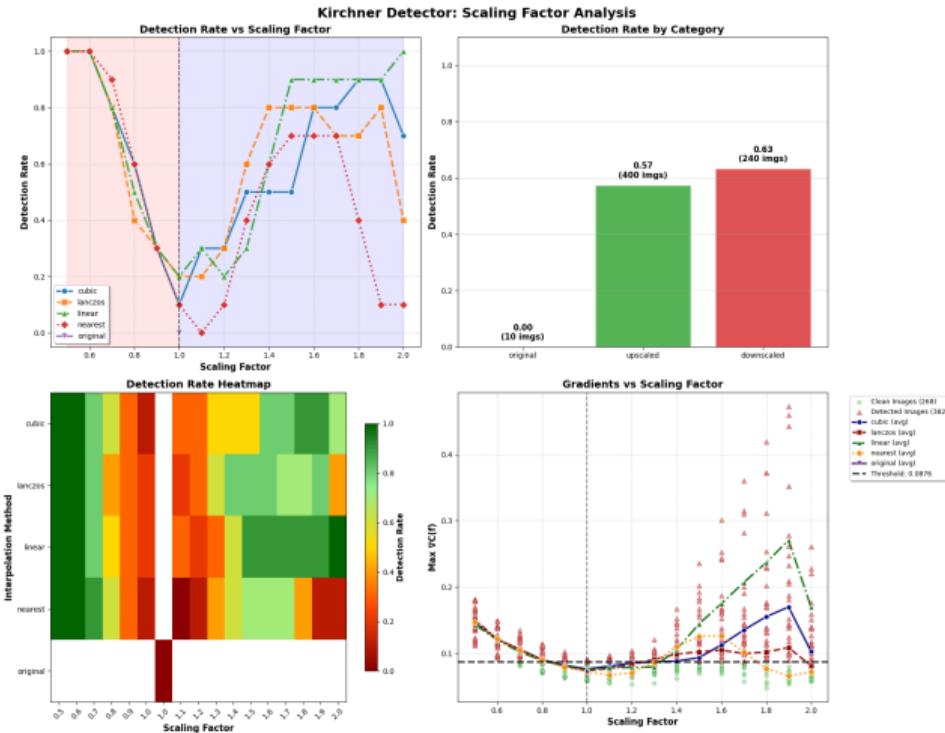
- 7 Originalbilder
- 16 Scalierungsfaktoren
- 4 Interpolationsmethoden
- In Summe 455 Bilder
- Detection Rate von > 75%
- Sehr **Inhomogene Bilder**

Fallauflösung: Analyseergebnisse(Ausgewählte Daten)



Hinweis: Ja, das ist keine Kartoffel,...

Fallauflösung: Analyseergebnisse("Realistische" Daten)



- 11 Originalbilder
- 16 Scalierungsfaktoren
- 4 Interpolationsmethoden
- In Summe 650 Bilder
- Detection Rate von > 55%

Fallauflösung: Analyseergebnisse("Realistische" Daten)



Grenzen der Methode & Grenzen unsere Implementierung

Methode

- **Sehr Abhängig vom Bildinhalt**
(Homogene Bildbereiche=False Positives)
- **"Fixed" Threshold** für δ'_T kann zu Fehlklassifikationen führen
- **Interpolationsmethoden** können die Detektion beeinflussen
- **Anti-Forensik-Methoden** können die Detektion umgehen [3]

Unsere Implementierung

- **Rotation** ist durch Spectrum/Gradienten Map erkennbar, aber Threshold funktioniert nicht
- **Lokale Änderungen** scheinen nicht sonderlich auf
- Eigenen Bilder wurden **"nur" mittels Smart-Phone Camera** aufgenommen(.tif wurden auch getestet [1])

Was hat gut funktioniert? & Lessons Learned

Was hat gut funktioniert?

- Testing Framework, um verschiedene Operationen automatisch zu testen
- Gute Visualisierung der Ergebnisse

Lessons Learned

- Methode liefert ein **starkes Indiz**, jedoch keinen abschließenden Beweis
- Rechenintensive Algorithmen wie das vollständige Gradientenverfahren sind nicht immer besser(Sobel operators [5])
- **Qualität der Testdaten** ist entscheidend für zuverlässige Ergebnisse und Methodenverständnis

Quellen

- [1] *Columbia Uncompressed Image Splicing Detection Evaluation Dataset*. URL: <https://www.ee.columbia.edu/ln/dvmm/downloads/authsplcuncmp/>.
- [2] Matthias Kirchner. "Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue". In: *Proceedings of the 10th ACM workshop on Multimedia and security*. MM&Sec '08. New York, NY, USA: Association for Computing Machinery, Sept. 2008, pp. 11–20. ISBN: 978-1-60558-058-6. DOI: 10.1145/1411328.1411333. URL: <https://doi.org/10.1145/1411328.1411333> (visited on 06/01/2025).
- [3] Matthias Kirchner and Rainer Bohme. "Hiding Traces of Resampling in Digital Images". In: *IEEE Transactions on Information Forensics and Security* 3.4 (Dec. 2008), pp. 582–592. ISSN: 1556-6013. DOI: 10.1109/TIFS.2008.2008214. URL: <http://ieeexplore.ieee.org/document/4668368/> (visited on 06/04/2025).
- [4] A.C. Popescu and H. Farid. "Exposing digital forgeries by detecting traces of resampling". In: *IEEE Transactions on Signal Processing* 53.2 (Feb. 2005), pp. 758–767. ISSN: 1941-0476. DOI: 10.1109/TSP.2004.839932. URL: <https://ieeexplore.ieee.org/document/1381775> (visited on 06/01/2025).
- [5] *Sobel operator*. en. Page Version ID: 1295912455. June 2025. URL: https://en.wikipedia.org/w/index.php?title=Sobel_operator&oldid=1295912455 (visited on 06/26/2025).

Vielen Dank für Ihre Aufmerksamkeit!

Fragen?

Kirchner's Fast Resampling Detection

InterpoLIE-tion - Catching lies through interpolation analysis

Dominik Barbist, Lukas Egger