

Automatisierte Aufbereitung archäologischer Grabungsfotos mittels Computer Vision

Simon Metzger

Masterarbeit

zur Erlangung des akademischen Grades Master of Arts im
Studiengang Digitale Methodik der Geistes- und
Kulturwissenschaften

Johannes-Gutenberg-Universität Mainz und Hochschule Mainz

Zusammenfassung

In der Archäologie, aber auch in anderen grabenden Wissenschaften werden Funde und Befunde fotografiert und mittels einer daneben platzierten Tafel mit Metadaten versehen. Selbst wenn diese Fotografie schon digital erfolgt ist, fehlt es oft an weiteren Metadaten. Praktisch wäre daher, die Metadaten von den Tafeln zu entnehmen und den Fotos zuzuordnen. Im Rahmen des Projektes erfolgt das in zwei Schritten: Der Erkennung der Tafel, um die Textregion zu finden und die Texterkennung selbst. Beides soll in einer Python-Pipeline abgehandelt werden. Für Ersteres wurden CNNs erprobt, was nicht funktionierte, und schließlich auf klassische Computervision zurückgegriffen. Die Erkennung basiert auf findcontours, was zunächst eine Binarisierung notwendig macht. Dafür gibt es zwei verschiedene Ansätze, die sich bewährt haben: Der adaptive und der iterative. Bei ersterem werden Pixel im Kontext des sie umgebenden Bildausschnittes binarisiert, bei zweiterem werden verschiedene Grenzwerte für die Binarisierung genutzt. Die so erhaltenen Konturen werden über den Flächeninhalt mit dem sie umgebenden kleinstmöglichen Rechteck verglichen. Nähern sich die Flächen ausreichend einander an, gilt auch die Kontur als Rechteck und somit als mögliche Tafel. Es folgt das Ausschneiden der gefundenen Bildregionen in zwei möglichen Verfahren. Das erste entnimmt den Bereich des kleinstmöglichen Rechtecks aus dem Bild und eignet sich nur bedingt, um die Rotation der Tafeln auszugleichen. Das zweite dient dazu, die tatsächlichen Eckpunkte der Tafeln zu ermitteln und anhand dieser eine Projektion auf ein Rechteck vorzunehmen. Anschließend werden die so erzeugten Bildausschnitte mittels OCR auf ihre Beschriftung untersucht. Dabei müssen die Problematiken der Kreidebeschriftung ausgeglichen werden. Die Ergebnisse werden anschließen evaluiert und ausgegeben.

Inhaltsverzeichnis

1 Einleitung	2
1.1 Grabung Kapitol	3
2 Material und Methoden	5
2.1 Fotos	5
2.2 Software	7
2.3 Tafeldetektion	7
2.3.1 CNN-basierter Ansatz	7
2.3.2 Kontur-basierter Ansatz	8
2.4 Cropverfahren	15
2.4.1 Simple Crop	16
2.4.2 Hough ¹ Crop	18
2.5 Texterkennung	21
2.5.1 Preprocessing	22
2.5.2 OCR	23
2.6 Evaluation	24
3 Ergebnisse	26
3.1 Gesamtergebnis	26
3.2 Tafelerkennung	26
3.3 Texterkennung	29
3.4 Weitere Tafeln	30
4 Diskussion	31
4.1 Tafelerkennung	31
4.2 Texterkennung	32
4.3 Empfehlung	36
5 Fazit	38

¹Gesprochen: [hʌf]

1 Einleitung

In der Archäologie, wie auch in anderen, „grabenden“ Wissenschaften, werden Fotografien zur Dokumentation von Grabungen und Befunden eingesetzt. Diese werden durch die Positionierung von beschrifteten Tafeln im Bild mit Metadaten angereichert. Sowohl für die Digitalisierung von Altbeständen als auch für die Verwaltung von bereits digital aufgenommenen Fotos ist es notwendig, diese Metadaten auszulesen und in Datenbanken oder in die Metadaten der Bilddateien selbst einzupflegen.

Diese Aufgabe stellt sich auch im Kapitol-Projekt am Deutschen Archäologischen Institut in Rom. Gegenstand des Projektes sind Grabungen auf dem Gebiet des historischen römischen Staatsheiligtums unter Einbezug der in der Vergangenheit durch bereits durchgeführten Grabungen. Es liegen zahlreiche Fotos vor, die Metadaten in Form von Schrifttafeln enthalten. In der Auseinandersetzung mit diesem Material stellt sich die Frage, ob der Aufwand durch eine automatisierte Erkennung der Tafeln und deren Beschriftung reduziert werden kann. Erforderlich wären dazu zwei Schritte: Die Lokalisation der Tafeln und die Erkennung des Textes darauf.

Die Detektion bestimmter Formen und Objekten gehört zu den klassischen Anwendungen der Computer Vision. Die Verwendung der Konturen zur Erkennung eines Objektes ist erprobt und entsprechende Funktionen sind fester Bestandteil der Computer Vision-Bibliothek OpenCV. Grundlegend für diese Anwendungsgebiete sind Kanten- und Linienerkennung wie die von Canny [Canny, 1986] bzw. Hough [Hough, 1962] so wie Template-Matching-Verfahren [OpenCV-Documentation, 2021e] zur Identifikation von Bildausschnitten in einem größeren Gesamtbild. Die praktischen Anwendungsbereiche sind vielfältig, so z.B. in der Robotik und Mustererkennung [Adamek and O'Connor, 2003] oder als Orientierungshilfe für autonome Roboter [Shaw and Barnes, 2006]. Auch wenn für komplexere Aufgaben inzwischen vor allem auf Neuronale Netzwerke zurückgegriffen wird [O’Shea and Nash, 2015], werden auch Objekte wie AR-Marker und QR-Codes auch heute noch mit diesen Methoden erfasst [Siltanen, 2012].

Auch an und mit der Optical Character Recognition (OCR) wird seit der Entwicklung digitaler Computer gearbeitet. Die ersten kommerziellen Produkte wa-

ren bereits in den 50er Jahren verfügbar [Eikvil, 1993] und werden seitdem stetig verbessert. Der letzte große Schritt war die Einführung von Neuronalen Netzwerken auch in diesem Bereich. Die im Rahmen dieser Arbeit verwendete Tesseract-OCR-Engine vollzog diesen Schritt 2018 [Tesseract Documentation, 2021b]. In der OCR wird zwischen offline und online Texterkennung unterschieden, wobei online das Tracking des Schreibprozesses meint und sich offline-Texterkennung mit bereits geschriebenen Text befasst [Feldmann, 2001]. Im Bereich der offline-Erkennung wird unterschieden, ob der Text als Scan oder vergleichbares Ausgangsmaterial in einer kontrollierten Umgebung vorliegt, oder ob es sich um Text in einer natürlichen Umgebung (*natural scenes*) handelt. Bei letzterem muss der Text erst gefunden und Faktoren, wie der Winkel zur Kamera oder die Lichtverhältnisse, ausgeglichen werden [Agnes Forsberg and Melvin Lundqvist, 2020] [Ye et al., 2007] [Chen et al., 2004]. Diese Faktoren erschweren die Texterkennung deutlich, weshalb hier mit mehr Aufbereitung und schlechteren Ergebnissen gerechnet werden muss, als in natürlicher Umgebung. Eine weitere große Herausforderung im Bereich offline-OCR ist die Erkennung von Handschriften. Handschriften verschiedener Personen unterscheiden sich stark voneinander [Hallale and Salunke, 2013], aber auch die Handschrift einer Person ist unregelmäßig und voller Variation [Feldmann, 2001]. Druckbuchstaben können dabei von herkömmlichen *Engines* deutlich besser erkannt werden als Schreibschrift. In den Geisteswissenschaften wird dieses Thema vor allem in Bezug auf historische Quellen interessant. Programme wie Transkribus [READ-COOP, 2021] bieten die Möglichkeit, Texte zu Transkribieren und auf dieser Basis ein Neuronales Netzwerk auf spezifische (Hand-)Schriften zu trainieren. Entscheidend für den Erfolg dieses Vorgehens ist das Vorliegen ausreichender Datenmengen; Eine Voraussetzung, die in diesem Projekt nicht erfüllt ist.

1.1 Grabung Kapitol

Seit den späten 90er Jahren führte die *Sovraintendenza ai beni culturali del Comune di Roma* Grabungen im Gebiet des antiken römischen Staatsheiligtums auf dem Kapitolinischen Hügel in Rom durch [Danti et al., 2014]. In den Jahren 2011-2014 wurde verstärkt auch auf dem Gebiet um das ehemalige preußischen Hos-

pital gegraben. Seit 2018 wird das Projekt vom Deutschen Archäologischen Institut und den *Musei Capitolini* weitergeführt [Dally, 2021]. Dabei sollen Fragen zur „kulturgeschichtlichen Entwicklung des Kapitolhügels in Antike, Mittelalter und Neuzeit“ [Dally, 2021] beantwortet werden. Im Rahmen des Projektes sollen digitale Methoden zur Dokumentation und Erkenntnisgewinn eingesetzt werden. Das bedeutet einerseits die Erfassung aktueller Befunde mittels Bodenradar, Fotogrammetrie oder 3D-Modellen [Dally, 2021]. Andererseits sollen historische Quellen und Dokumentationen von Altgrabungen digitalisiert und zugänglich gemacht werden. Historische Karten, Dokumente und Beschreibungen sowie die bei bisherigen Grabungen gemachten Funde und Befunde könnten dabei helfen, das Bild des Kapitols zu vervollständigen – durch die reine Erschließung, aber auch durch die Verarbeitung mit modernen Mitteln der Wissenschaft und Technik. Dazu gehören auch die Grabungsfotos, die Gegenstand dieser Arbeit sind.

2 Material und Methoden

In diesem Kapitel werden das zu Grunde liegende Material – die Grabungsfotos – und die darauf angewendete Software vorgestellt. Außerdem wird die Verarbeitung der Fotos im Detail beschrieben. Diese gliedert sich in zwei Teilbereiche: Die Tafeldetektion und die Texterkennung (OCR). Die Tafeldetektion wiederum ist in die eigentliche Erkennung sowie das Ausschneiden der Tafeln aus dem Gesamtbild unterteilt. Für beide Arbeitsschritte gibt es je zwei Varianten.

2.1 Fotos

Der Gegenstand dieser Arbeit sind 1.424 Fotos, die während der Grabungen auf dem Kapitol in Rom in den Jahren 2011-2014 entstanden sind. Die Fotos haben eine Auflösung von 3264 x 2448 Pixeln und liegen im JPG-Format vor. Teilweise sind die exif-Daten (*Exchangeable Image File Format*) vorhanden, die von der Kamera erzeugt wurden, also Informationen, die nach den Standards für Metadaten in Bilddateien[CIPA Standardization Committee, 2020] angelegt wurden. Sie beinhalten das Datum, das Kameramodell (Nikon Coolpix L19), die Belichtungsdauer und den ISO-Wert für die Lichtempfindlichkeit des Fotosensor. Weitere Metadaten – Datum, Ort, das Kürzel für die Grabung sowie weitere Anmerkungen – befinden sich auf Tafeln, die auf einigen der Fotos zu sehen sind. Das Datum auf der Tafel und das in den exif-Daten weichen teilweise um mehrere Tage voneinander ab. Allerdings sind beide Daten von Bedeutung, um den Grabungsverlauf zu rekonstruieren. Bei den Tafeln, die auf den Grabungsfotos zu sehen sind, handelt es sich um Schiefertafeln mit einem Holzrahmen, die mit Kreide beschriftet wurden (Abbildung 1). Die Detektion der Tafeln beruht auf folgenden Faktoren: (1) Die Tafeln haben grundsätzlich eine rechteckige Form. (2) Durch die Breite des Rahmens können bis zu zwei Rechtecke erkannt werden, ein inneres und ein äußeres. (3) Der Rahmen ist hell und die Schreibfläche dunkel, es besteht ein starker Kontrast. Die im Beispielbild gezeigte Tafel stellt ein Idealbild dar: Die Tafel nimmt einen relativ großen Teil des Originalbildes ein. Sie ist frontal vor der Kamera positioniert. Die Beleuchtung ist gut und indirekt. Keines der weiteren Bildelemente verdeckt die Tafel. Diese Beschreibung impliziert schon die



Abbildung 1: Beispiel eines Fotos der verwendeten Tafel. GOT bezeichnet die Kampagne, darunter folgt das Datum. US (*unità stratigrafica*) bezeichnet die stratigrafische Einheit.

Problemfelder, die bei der Detektion beachtet werden müssen:

1. Die Tafel ist unter Umständen rotiert (Abbildung 2).
2. Die Distanz der Tafel zur Kamera und damit ihre Größe im Bild kann stark variieren.
3. Der Rahmen der Tafel kann teilweise verdeckt oder anderweitig durch Gegenstände überlagert sein (Abbildung 2).
4. Ein geringer Kontrast des Hintergrunds zum Tafelrahmen kann die Detektion erschweren.
5. Unregelmäßigkeiten im Rahmen, die auf grobe Verarbeitung oder Abnutzung zurückzuführen sind, können die Detektion erschweren.
6. Die Beleuchtung kann zu Problemen führen. Grundsätzlich sind alle Fotos hell und gut ausgeleuchtet, direktes Licht kann sich aber, bedingt durch Spiegelungen, negativ auf die Kontraste auswirken.
7. Weitere Gegenstände, die den Spezifika der Tafeln entsprechen, können im Bild vorhanden sein.



Abbildung 2: Problematische Tafeln: Rotation und teilweise verdeckter Rahmen.

2.2 Software

Der Programmcode ist in Python (Version 3.8) geschrieben. Grund für diese Wahl war, dass sowohl OpenCV als auch verschiedene OCR-Engines über Python-Anbindungen verfügen. Verwendet werden die *packages* Numpy (1.19.5) und vor allem OpenCV (4.4.0.44). Die Texterkennung arbeitet mit PyTesseract (0.3.7). Für die Evaluation wurde die Bibliothek Difflib (3.8) eingebunden.

2.3 Tafeldetektion

Der folgende Abschnitt befasst sich mit der ersten Teilaufgabe dieser Arbeit: Der Detektion der Tafeln auf den Grabungsfotos. Hier werden mehrere Ansätze vorgestellt, die im Laufe der Auseinandersetzung mit dem Thema erprobt wurden: die Tafeldetektion mittels CNNs und mittels klassischer Computer Vision über die Detektion von Konturen.

2.3.1 CNN-basierter Ansatz

Der auf Convolutional Neural Networks (CNN) basierende Ansatz bestand darin, bereits bestehende, trainierte Datensätze für die Bilderkennung auf die Grabungsfotos anzuwenden.

CNNs werden auf großen Datenbanken mit bekannten Klassifizierungen trainiert, um unbekannte Bilder auf dieser Grundlage klassifizieren zu können. So basiert

das COCO-Dataset² auf über 330.000 Bildern mit 1,5 Millionen Objekten [Dataset, 2021] darauf. Das Training wird, je nach Datenmenge, auf Grafikkarten oder Großrechnern durchgeführt [Schürholz and Spitzner, 2019].

Da die Tafeln Objekten wie Büchern, Verpackungen von Frühstücksflocken oder anderen rechteckigen, beschriebenen Objekten ähneln, die im trainierten Datensatz enthalten sind, sollte hier geprüft werden, ob die Tafeln regelmäßig und zuverlässig als eines dieser Objekte erkannt werden. Zum Einsatz kamen auf dem Datensatz von COCO trainierte Modelle mit Gewichten von COCO und YOLO (You Only Look Once [Redmon, 2021] [Redmon et al., 2015]).

2.3.2 Kontur-basierter Ansatz

Beim Kontur-basierten Ansatz wurden die Umrisse der Objekte auf dem Foto erfasst. Aus diesen sollten alle Objekte mit annähernd rechteckiger Form ausgewählt werden. Bei diesen, so die Annahme, handelt es sich um mögliche Tafeln.

`Cv2.Contours` basiert auf einem Algorithmus, der Punkte gleicher Farbe und Intensität umrandet [Suzuki and Abe, 1985]. Das Resultat ist eine Liste von Punkten, die ein geschlossenes Polygon ergeben. Die gefundenen Konturen können mit einer Hierarchie versehen werden: Konturen, die sich innerhalb anderer Konturen befinden, gelten als deren „Kinder“. Das Verfahren ist darauf ausgelegt, auf binäre Bilder angewendet zu werden. Die Implementierung in OpenCV unterscheidet in Nullwerte und Nicht-Nullwerte [OpenCV-Documentation, 2021f]. Nicht-binäre Bilder werden dadurch automatisch binarisiert. Für die Erzielung optimaler Ergebnisse kommt dem Binarisierungsverfahren eine große Bedeutung zu.

Auf Basis dieses Algorithmus lässt sich Detektionsverfahren aufbauen, das Objekte, die sich vom Hintergrund der Bilder abheben, zu erkennen. Die Herausforderung besteht, nach diesem Ansatz, in zwei Punkten: Erstens muss die Binarisierung so erfolgen, dass eine möglichst saubere Trennung von Vordergrund (Objekten) und Hintergrund (vor allem Erde) der Bilder stattfindet und zweitens müssen aus den gefundenen Vordergrundobjekten diejenigen ausgewählt werden, die als Tafeln in Frage kommen. Bei der Binarisierung haben sich zwei Wege als praktikabel herausgestellt, die im Folgenden beide präsentiert werden sollen. Diese

²Common Objects in Context [Dataset, 2021].

Ansätze werden im Folgenden als adaptiver und als iterativer Ansatz bezeichnet. Im Flowchart sind die Abläufe der Rechtecksdetektion dargestellt (Abbildung 3).

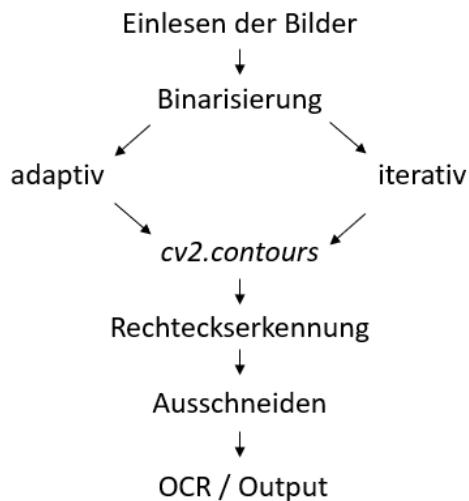


Abbildung 3: Flowchart Rechteckserkennung: Die Binarisierung erfolgt durch einen der beiden möglichen Ansätze. Auf dieser Basis wird die eine Konturerkennung durchgeführt. Aus diesen Konturen werden die Rechtecke ausgewählt und aus dem Gesamtbild ausgeschnitten. Es folgt die weitere Verarbeitung.

Adaptive Binarisierung

Das einfachste Verfahren der Binarisierung eines Bildes besteht darin, einen Schwellwert festzulegen. Dieser muss auf der Spanne der Farbwerte, also zwischen 0 und 255 liegen. Farbwerte unterhalb dieses Schwellwertes werden zu Nullen, Farbwerte darüber zu Einsen. Das Ergebnis ist ein Schwarz-Weiß-Bild. Bei komplexen Szenen, wie den Grabungsfotos, ist dieses Verfahren jedoch zu einfach. So kann ein Foto beispielsweise stark unterschiedliche Beleuchtung, wie direktes Sonnenlicht und Schatten, enthalten. Eine Differenzierung innerhalb dieser Zonen ist so nicht möglich (Abbildung 4).

Daher wurde für den ersten Ansatz ein adaptiven Thresholds [OpenCV-Documentation, 2021d] gewählt: Statt global, über das gesamte Bild, einen Schwellwert festzulegen, können lokale Schwellwerte errechnet werden. Die Größe des lokalen Ausschnittes sowie das exakte Verfahren können dabei frei gewählt werden. In diesem Fall wurde

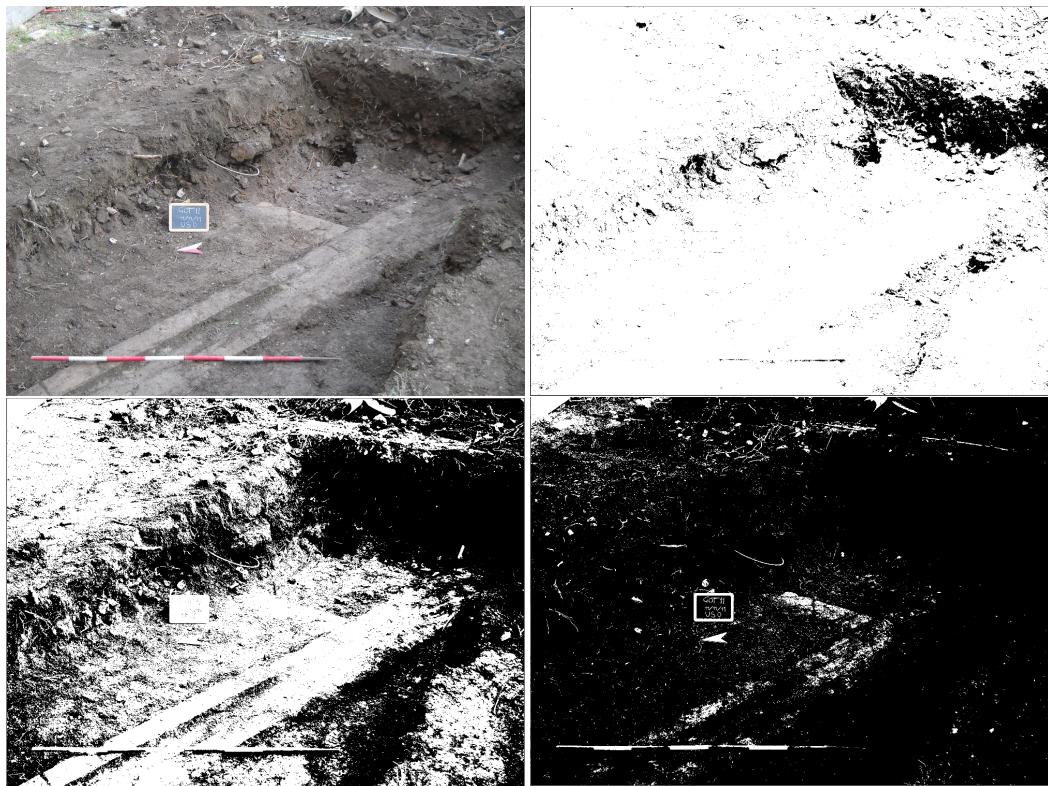


Abbildung 4: Grabungsfoto im Original (o.l.), mit niedrigem (60, o.r.), mittlerem (125, u.l.) und hohem (185, u.r.) Schwellwert.

für die Binarisierung ein Kernel von 11×11 Pixeln angewendet: Für jeden Pixel im Bild wurde durch die Kreuzkorrelation mit dem Gaußfenster die gewichtete Summe der Pixel innerhalb des Kernels berechnet, um diese als lokalen Schwellwert zu verwenden. Die Beleuchtung oder Farbunterschiede innerhalb des Bildes können so ausgeglichen werden (Abbildung 5).

Vor der Binarisierung wird das Bild in ein Graustufenbild umgewandelt und, unter Beibehaltung der Seitenverhältnisse, auf eine Größe von 1000×750 Pixeln verkleinert. Das verbessert die Genauigkeit der Detektion und verringert die zu verarbeitende Datenmenge, wodurch eine Beschleunigung des Prozesses zu erwarten ist.

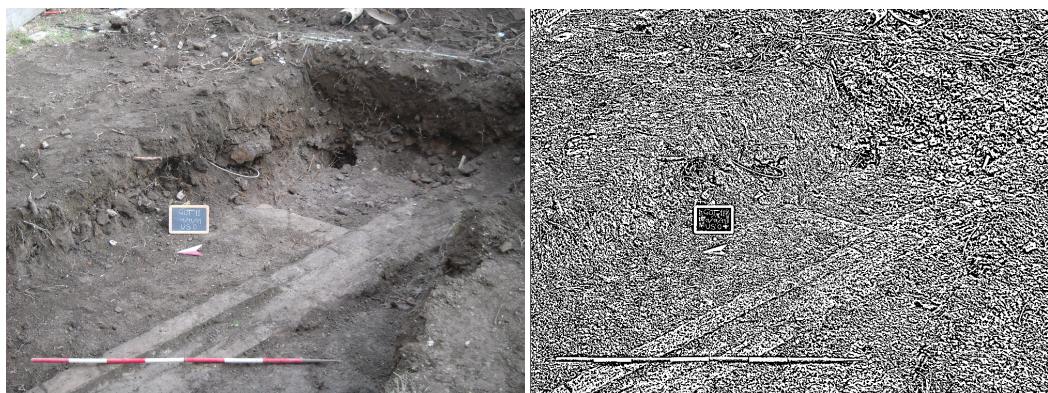


Abbildung 5: Grabungsfoto im Original sowie mit adaptivem Threshold.

Iterative Binarisierung

Die Grundidee der iterativen Binarisierung besteht darin, das Bild mit einem globalen Schwellwert zu binarisieren. Die Problematik davon besteht darin, dass bei großen Beleuchtungsunterschieden oder sehr hellen oder dunklen Objekten auf dem Bild ganze Bereiche durch den Schwellwert von der weiteren Bearbeitung ausgeschlossen werden. Der iterative Ansatz sieht daher vor, den Schwellwert von 20 auf 200 in Fünferschritten zu erhöhen. Dadurch entstehen pro Foto 37 binäre Bilder, auf die die Konturenerkennung angewendet werden kann. Auf jedem dieser Bilder können mehrere mögliche Tafeln erkannt werden³. Der nächste Schritt besteht also darin, aus diesen möglichen Tafeln die auszuwählen, die am wahrscheinlichsten tatsächlich eine ist. Dazu wird eine Grundannahme getroffen: In dem Bereich, in dem auf den meisten der 37 Bilder eine Tafel vermutet wird, befindet sich tatsächlich eine Tafel. Alle anderen werden als Falsch-Positive betrachtet. Diese Annahme beruht auf zwei Faktoren: Erstens hat sich gezeigt, dass Objekte, die keine Tafeln sind, aber als solche erkannt werden können – z.B. Fenster, Türen oder Plakate – nur unter wenigen Schwellwertes als solche eingeordnet werden. Das liegt unter anderem daran, dass die Tafeln einen hellen Holzrahmen und eine dunkle Innenfläche haben, was zu einem starken Kontrast führt, der auf

³Die eigentliche Tafelerkennung wird erst im folgenden Abschnitt beschrieben. Da die dort gewonnenen Informationen nicht, wie beim adaptiven Ansatz, an das Hauptprogramm übergeben, sondern innerhalb der Funktion des iterativen Ansatzes weiter verarbeitet werden, ist hier ein Vorgriff nötig.

vielen Stufen des Schwellwertes erhalten bleibt⁴. Außerdem werden durch eben diesen Rahmen in mittleren Schwellwert-Bereichen die Tafeln oft zweimal erkannt: Einmal an der Außenkante und einmal an der Innenkante des Rahmens. Dadurch häuft sich die Detektion möglicher Tafeln in diesem Bereich.

Basierend auf dieser Annahme wird für alle möglichen Tafeln immer paarweise die intersection over union berechnet. Dieser Algorithmus basiert auf dem Jaccard-Koeffizienten zur Berechnung der Ähnlichkeit zweier Mengen [Jaccard, 1902]. Der Koeffizienten wird errechnet, indem die Schnittmenge durch die Vereinigungsmenge geteilt wird. Das Ergebnis liegt zwischen 0 und 1. Je mehr es sich der 1 annähert, desto ähnlicher sind die Mengen. Diese Berechnung lässt sich auch auf die Rechtecke, mit denen die möglichen Tafeln verortet werden, anwenden (Abbildung 6).

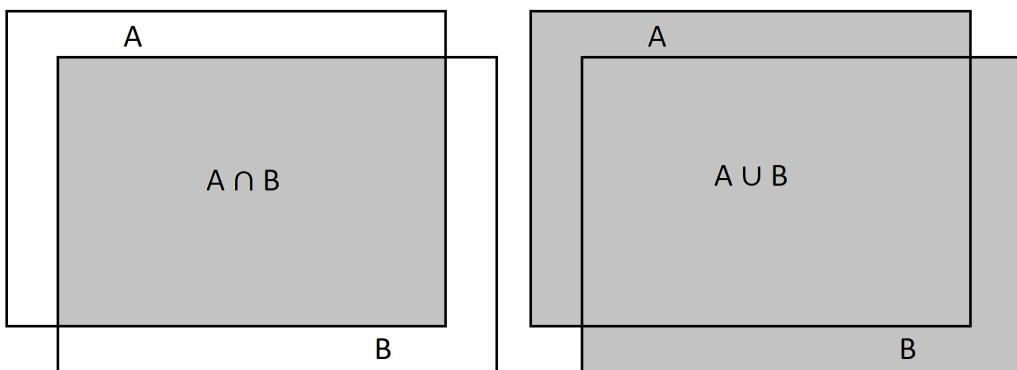


Abbildung 6: Schnittmenge und Vereinigungsmenge von Rechtecken. Der Jaccard-Koeffizient wird durch die Division der beiden Flächen bestimmt.

Zwei Rechtecke galten im entwickelten Algorithmus dann als hinreichend ähnlich, wenn der Jaccard-Koeffizient über 0.9 lag. Jedes Mal, wenn ein Rechteck im Vergleich mit einem anderen diesen Wert erzielt, wird für das Rechteck ein Zähler erhöht. Das Rechteck, welches am Ende der Vergleiche den höchsten Wert in diesem Zähler erzielt, wird als tatsächliche Tafel angenommen. Eine Ausnahme ergibt sich, wenn dieser Wert unter 6 liegt. In diesem Falle wird von Falsch-Positiven und somit einem Foto ohne Tafel ausgegangen.

⁴Die Tafeln verfügen damit über eine Eigenschaft, die auch beim Einsatz von AR-Markern genutzt wird: Starke hell-dunkle Kontraste beschleunigen und vereinfachen die Detektion [p. 45]ar-marker

Rechteckserkennung

Ist die Binarisierung der Bilder nach einer der beiden vorgestellten Methoden erfolgt, können mittels `cv2.findContours` die Konturen der Objekte darauf gefunden werden (Abbildung ??). Die Hierarchie der Konturen wird dabei außer Acht gelassen, da sich daraus keine verlässlichen Informationen gewinnen lassen. Als nächstes müssen unter diesen Konturen die ausgewählt werden, die als Tafel



Abbildung 7: Grabungsfoto im Original sowie nach der Anwendung der Konturerkennung.

in Frage kommen. Dazu wird die Tatsache genutzt, dass Tafeln auf den Fotos zumindest annähernd als Rechtecke abgebildet werden. Durch ihre Lage zur Kamera können sie zu einem gewissen Grad davon abweichen, grundsätzlich bleibt diese Form aber erhalten. Andere Rechtecke kommen zwar vor, wie Plakate, Fenster, Türen, Ziegel und Fliesen, unterscheiden sich jedoch meist in der tatsächlichen Form oder können im Zweifelsfall bei der späteren Texterkennung ausgeschlossen werden. Dementsprechend geht es in der weiteren Tafelerkennung darum, Rechtecke zu finden und diese durch verschiedene weitere Kriterien so zu sortieren, dass alle Tafeln und möglichst nur Tafeln erkannt werden. Das wird in mehreren Schritten erreicht: Zunächst wird die Fläche der Konturen berechnet. Genutzt wird hierfür die OpenCV-Funktion `cv2.contourArea`. Ist die Fläche zu klein, wird die Kontur aussortiert. Als Schwellwert wird hier die Kantenlänge der längsten Kante des Bildes genommen, damit bei höherer Auflösung weiterhin korrekt sortiert werden kann(Abbildung 8). Eine Größenbegrenzung ist sinnvoll, da die Tafeln in der Regeln eher prominent im Bild zu sehen sind. Sollte tatsächlich eine

Tafel durch dieses Kriterium aussortiert werden, wäre die spätere Texterkennung ohnehin so erschwert, dass die Tafel in der weiteren Verarbeitung keinen Mehrwert hätte.

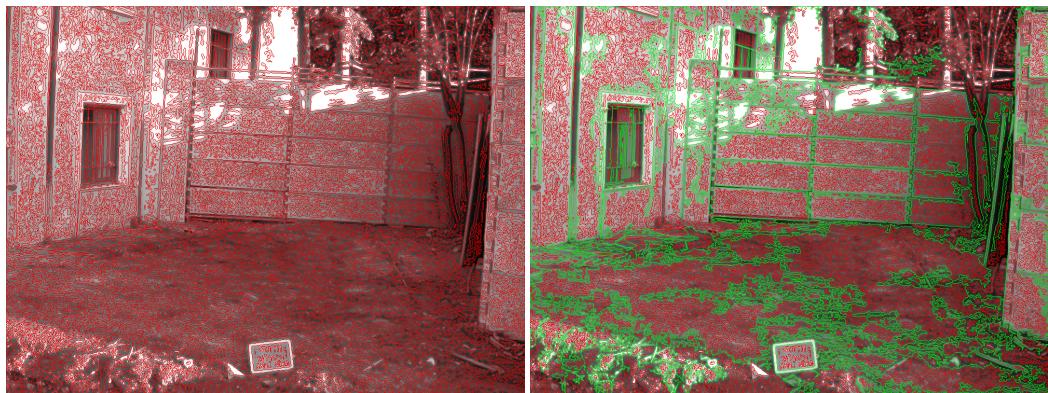


Abbildung 8: Konturen vor und nach der Selektion nach Fläche.

Der nächste Schritt besteht darin, durch `cv2.minAreaRect` das kleinstmögliche Rechteck um die Kontur zum bilden (Abbildung 9).

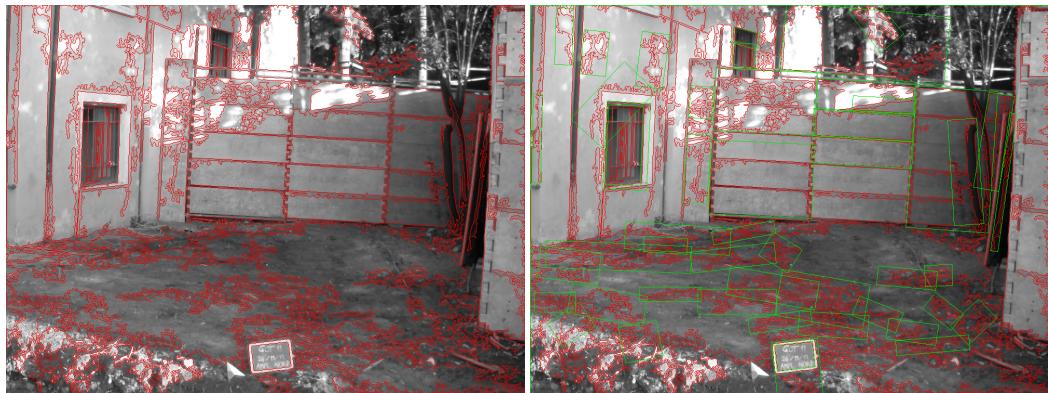


Abbildung 9: Konturen ohne und mit Rechtecken.

Anschließend werden die bereits berechneten Flächen der Konturen mit denen der kleinstmöglichen Rechtecke verglichen. Die Annahme: Nähert sich der Flächeninhalt der Kontur dem des Rechtecks an, handelt es sich bei der Kontur selbst wahrscheinlich um ein Rechteck. Dabei hat sich als Schwellwert bewährt, dass die Fläche der Kontur 85% der Fläche des Rechtecks betragen sollte. Ebenfalls ausgeschlossen wird ein Rechteck, wenn es die Fläche des gesamten Bildes

ausmacht, da teilweise, wie auch bei diesem Beispiel, der Rand des Bildes als Kontur erkannt werden kann (Abbildung 10).



Abbildung 10: Rechtecke aller Konturen und jene derer mit annähernd rechteckiger Fläche.

Als letztes Kriterium wird das Seitenverhältnis herangezogen. Im Programm besteht die Möglichkeit, vor der Untersuchung aller Bilder ein Template zu hinterlegen, auf dem eine Tafel gut und eindeutig zu sehen ist. Aus diesem Template wird das Seitenverhältnis der Tafel berechnet. Findet dieser Schritt nicht statt, wird ein maximales Seitenverhältnis von 2:1 angenommen, da Tafeln selten in länglichen Formaten zu finden sind⁵. Mit einer deutlichen, Toleranz von 30% in jede Richtung wird das Seitenverhältnis der verbliebenen Rechtecke mit dem des Templates verglichen. Sind die Werte sich ähnlich genug, wird das Rechteck als Tafel interpretiert und gilt als das Ergebnis des Algorithmus (Abbildung 11).

2.4 Cropverfahren

Durch die vorhergehende Rechtecksdetektion sind die Positionen der Tafeln bekannt. Der nächste Schritt besteht darin, die Tafeln aus dem Gesamtbild auszuschneiden, damit diese Ausschnitte für die Texterkennung genutzt werden können. Auch die perspektivischen Verzerrungen der Tafeln müssen ausgeglichen werden. Für diesen Ausgleich müssen die Eckpunkte der Tafeln bestimmt und das durch

⁵Sollte das Format einer Tafel doch einmal von dieser Annahme abweichen, lassen sich hier problemlos Anpassungen vornehmen.



Abbildung 11: Die bisher verbliebenen Rechtecke und die mit dem richtigen Seitenverhältnis.

sie definierte Viereck auf die Fläche eines Rechtecks übertragen // Auch hier haben sich wieder zwei Verfahren ergeben, von denen eines effizient arbeitet, während das zweite stärker auf das Entzerren der Bilder abzielt (Abbildung 12).

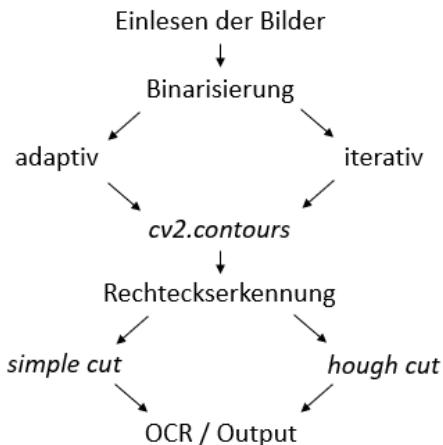


Abbildung 12: Vollständiges Flowchart, welche Prozesse die Fotos durchlaufen.

2.4.1 Simple Crop

Ein Weg, die Tafeln aus den Bildern auszuschneiden, besteht darin, die kleinstmöglichen Rechtecke aus dem Detektionsverfahren als Basis für den Ausschnitt zu

nutzen. Basierte die Detektion auf dem adaptiven Ansatz, musste zunächst das Rechteck auf die ursprüngliche Bildgröße skaliert werden, da für die weitere Verarbeitung die Originalbilder mit der höheren Auflösung verwendet wurden. Zu diesem Zweck wurde das Verhältnis aus der Größe des Originalbildes und der des skalierten Bildes gebildet. Multipliziert man dieses Verhältnis mit den vier Kennzahlen der Rechtecke (X- und Y-Koordinate des Mittelpunktes sowie Breite und Höhe) wird das Rechteck passend skaliert. Als letzter Schritt wurden aus diesen Werten, unter Einbezug der Rotation des Rechtecks, die Eckpunkte berechnet. Neben den so erhaltenen Ausgangskoordinaten mussten aber auch Zielkoordinaten erzeugt werden, auf die das Rechteck projiziert wird. Diese wurden ebenfalls dem Detektionsrechteck entnommen, indem Höhe und Breite als Maße für das Projektionsziel genutzt wurden.

Für die Transformation des Ausgangsrechtecks auf das Zielrechteck wurde mittels `cv2.getPerspectiveTransform` die Transformationsmatrix erzeugt ([OpenCV-Documentation, 2021b]). Die eigentliche Transformation erfolgte dann durch `cv2.warpPerspective` (Abbildung 11).

Der so erzeugte Tafelausschnitt wurde anschließend auf 1000 Pixel skaliert, was in der Regel eine Vergrößerung darstellt, um eine einheitliche Größe zu gewährleisten und zu kleine Ausschnitte zu vermeiden. Außerdem wurde das Bild um 90° rotiert, wenn die Höhe die Breite übertraf, um nur Ausschnitte im Querformat zu erhalten.

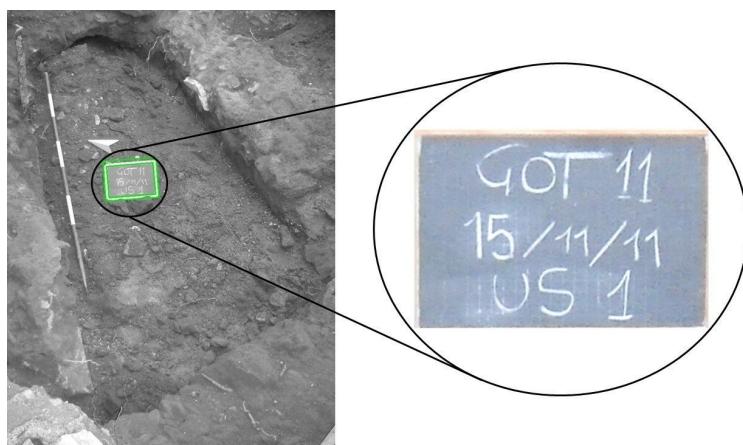


Abbildung 13: Eine Tafel vor und nach dem Ausschneiden.

2.4.2 Hough⁶ Crop

Durch unterschiedlichen Perspektiven, aus denen die Fotos aufgenommen wurden, ist die beschriftete Seite der Tafeln nicht immer frontal der Kamera zugewendet. Simple Crop ist nicht geeignet, um die daraus resultierende Verzerrung auszugleichen (Abbildung 14).



Abbildung 14: Diskrepanz zwischen den Eckpunkten der gefundenen Rechtecke und den Eckpunkten der Tafel.

Daher wurde ein zweiter Ansatz entwickelt, um dieses Problem zu beheben. Ziel war es hier, statt der Eckpunkte der gefundenen Rechtecke die Eckpunkte der Tafel für das Cropverfahren zu nutzen. Diese mussten also zunächst gefunden werden. Das kann erreicht werden, indem die Seiten der Tafeln durch Kanten detektion ermittelt und im Anschluss deren Schnittpunkte berechnet werden [Ye et al., 2007] [Shaw and Barnes, 2006].

Zunächst musste dazu ein passender Bildausschnitt gewählt werden. Auf den Tafeln befinden sich viele Objekte, die lange gerade Kanten aufweisen. Nach der Detektion zu entscheiden, welche davon Teil einer Tafel sind und welche nicht ist komplex und würde die gesamte Problematik der Detektion neu aufwerfen. Gleichzeitig konnten auch die detektierten Rechtecke nicht verwendet werden, da hier Teile des Rahmens abgeschnitten sein können. Daher wurden die Rechtecke um den Faktor 1,3 vergrößert und mittels simple crop ausgeschnitten (Abbildung 15).

Diese Bildausschnitte wurden mit mehreren Filtern bearbeitet, um das meist stark vorhandene Rauschen zu reduzieren. Es folgte die Umwandlung in ein Graustufenbild, das normalisiert wurde. Beim Prozess der Normalisierung werden Farbs-

⁶Gesprochen: [hʌf]



Abbildung 15: Bildausschnitt für Hough Crop, basierend auf dem vergrößerten Tafelrechteck.

kalen, in diesem Fall die Graustufen, auf ihre maximalen Werte gestreckt. Der niedrigste wird auf 0, der höchste auf 255 gesetzt und alle Werte dazwischen entsprechend angepasst (Abbildung 16).

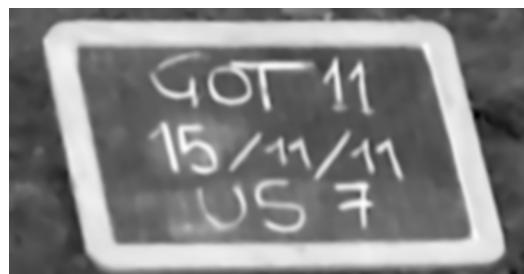


Abbildung 16: Graustufenbild, gefiltert und normalisiert.

Auf die normalisierten Bilder wurde die *Canny Edge Detection* (Kantenerkennung) angewendet [Canny, 1986]. Dieser Algorithmus berechnet die Gradienten der einzelnen Pixel, wodurch Kantenverläufe im Bild erkennbar werden. Auf diese wird eine *non-maximum suppression* angewendet, d.h., die Gradienten der Pixel werden mit ihren Nachbarn verglichen und jeweils nur das Maximum weiterverwendet. Die Kantenstärke wird somit auf die Breite von einem Pixel reduziert. Aus den so erhaltenen Gradienten werden mittels zweier Schwellwerte durch Hysterese die gewünschten Kanten bestimmt. Das bedeutet: Enthält ein Pixel einer Kante den zweiten, höheren Schwellwert, gilt sie als eine der gesuchten Kanten. Von dort ausgehend werden alle Pixel der Kante, die den niedrigeren Schwellwert übertreffen, ebenfalls Teil dieser Kante (Abbildung 17).

Dieses Gradientenbild wurde anschließend mittels Hough-Transformation [Hough, 1962]

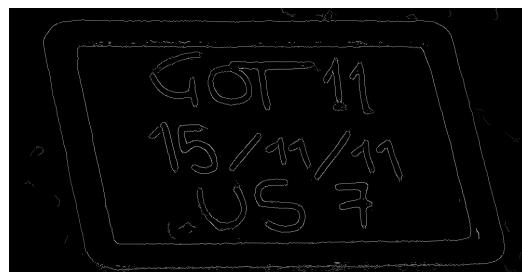


Abbildung 17: Kantendetektion mit dem Canny-Algorithmus.

einer Linienerkennung unterzogen. Die vorher detektierten Kanten, die einen beliebigen Verlauf nehmen können, wurden jetzt darauf geprüft, ob sie eine gerade Linie bilden. Dazu wird mit der hesseschen Normalform jede mögliche Gerade durch jeden Kantenpunkt berechnet. Jeder Kantenpixel, durch den die Geraden verlaufen, erhält einen *upvote*, es wird also ein Zähler inkrementiert. Durch die Bereiche mit den meisten *Upvotes* laufen die gesuchten Linien auf dem Bild. Ist eine Linie gefunden, ist sie wahrscheinlich Teil des Rahmens der Tafel und somit für die Erkennung der Eckpunkte relevant (18).

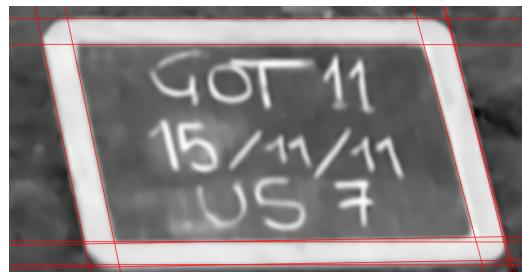


Abbildung 18: Linienerkennung mit dem Hough-Algorithmus.

Im nächsten Schritt wurden die Geraden mit dem Bildrahmen geschnitten. Die Schnittpunkte wurden als neue Endpunkte des Liniensegments für die weiteren Berechnungen genutzt, um etwaige Schnittpunkte außerhalb des Bildes ausschließen zu können. Die Endpunkte wurden, entsprechend ihrer Position im Bild, den vier Ecken zugeteilt. Anschließend wurden die Schnittpunkte der Linien untereinander berechnet, aber nur, wenn sie (1) nicht in die gleiche Richtung und (2) nicht diagonal durchs Bild verliefen. Beide Kriterien konnte durch die Einordnung der Eckpunkte bestimmt werden. Damit war sichergestellt, dass nur Linien miteinan-

der geschnitten wurden, die annähernd senkrecht zueinander verlaufen, wie es bei den Tafelrändern der Fall ist. Schnittpunkte am Rand wurden ausgeschlossen, um die Detektion des Bildrandes oder Störobjekte zu ausschließen.

Die Gruppe der Schnittpunkte wurde in die vier Ecken aufgeteilt. In jeder Ecke wurde dann der Punkt bestimmt, der am weitesten Richtung Bildmitte liegt. Dieser wurde als der wahrscheinlichste Eckpunkt angenommen (Abbildung 19). Das wei-

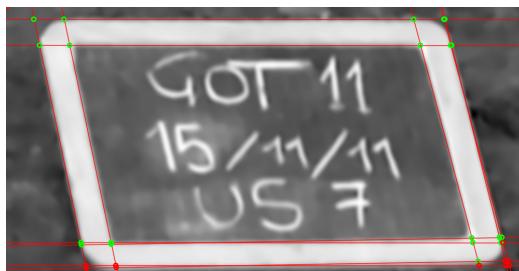


Abbildung 19: Die Schnittpunkte der Linien nach der Berechnung. Die roten Punkte liegen zu nahe am Rand und wurden daher ausgeschlossen.

tere Verfahren entsprach dem des `simple crop`-Ansatzes: Mit den Eckpunkten wurde eine Transformationsmatrix berechnet, durch die der Bildausschnitt innerhalb der vier Eckpunkte auf eine Rechtecksform übertragen werden konnte (Abbildung 20).

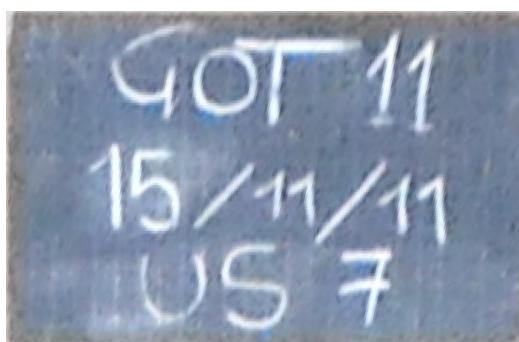


Abbildung 20: Die Tafel, in ein Rechteck transformiert.

2.5 Texterkennung

Im folgenden Abschnitt wird der zweite Aspekt der Verarbeitung von Grabungsfotos vorgestellt: Die Texterkennung oder *optical character recognition*. Da die

Vorbereitung der Bilder für das Gesamtergebnis von großer Bedeutung ist, wird zunächst das *Preprocessing* behandelt. Im Anschluss wird die eigentliche Texterkennung mit Tesseract vorgestellt.

Folgende Probleme müssen bei der Texterkennung adressiert werden: (1) Komplexität der Szene⁷, (2) Beleuchtungsverhältnisse, (3) Rotation des Textes relativ zur Kamera, (4) Unschärfe, (5) Größe und Format der Textfelder, (6) Neigungswinkel des Textes relativ zur Kamera, (7) Schriftart, (8) Mehrsprachigkeit, (9) perspektivische Verzerrungen [Hamad and Kaya, 2016]. In diesem Falle wurde allen Aspekten, die der Textdetektion gelten (1, 5), geringeres Gewicht beigemessen, da die Tafeln bereits als die Grenzen des beschriebenen Areals betrachtet werden konnten. Die Neigung und die Schiefe der Buchstaben relativ zur Kamera (3, 6, 9) wurden bereits beim Cropverfahren entsprechend der Möglichkeiten ausgeglichen. Beleuchtung, Unschärfe und Rauschen waren Teil des Preprocessings (2, 4), die sprachbezogenen Punkte (7, 8) wurden im Rahmen der eigentlichen Texterkennung adressiert.

2.5.1 Preprocessing

Das Preprocessing für die Texterkennung folgte gängigen Verfahren⁸ [Shah and Gokani, 2014] [Hallale and Salunke, 2013]. Zunächst mussten die Bilder in Graustufen umgewandelt werden. Nacheinander wurden zwei Filter angewendet, `cv2.GaussianBlur` [OpenCV-Documentation, 2021c] und `cv2.fastNlMeansDenoising` [OpenCV-Documentation, 2021a]. Ersterer dient als Weichzeichner, letzterer reduziert das Rauschen, indem Pixel verschoben werden, die dem Farbwert nach nicht in ihre Umgebung passen. Damit sollten die Störeffekte durch verwischte Kreide auf der Tafel reduziert werden. Das Bild wurde invertiert, sodass die eigentliche weiße Kreide schwarz dargestellt und der Hintergrund hell wurde, was den Empfehlungen für Tesseract entspricht [Tesseract Documentation, 2021a]. Der letzte Schritt bestand in der Normalisierung der Bilder. Alle diese Schritte wurden auf das Grundbild sowie jeweils auf die daraus extrahierten 3 Farbkanäle angewendet [Chen et al., 2004].

⁷Gemeint ist hier, wie komplex die Szene ist, wenn das Foto in einer natürlichen Umgebung aufgenommen wurde. Text auf einer das Bild füllenden Hauswand zu finden ist weniger anspruchsvoll als die Detektion von Text in einer Straßenszene mit Häusern, Autos und Pflanzen.

⁸Weitere übliche Schritte im Preprocessing, wie das Thresholding, unterblieben hier. Die Gründe dafür werden im Kapitel 4.2 diskutiert.

2.5.2 OCR

Pro gefundener Tafel waren jetzt vier Bilder vorhanden, die der Texterkennung unterzogen werden konnten. Für jedes Bild wurde diese zweifach ausgeführt: Einmal mit `image_to_string` und einmal mit `image_to_boxes`. Erstere liest den Text Zeilenweise ein und versucht dabei, ganze Wörter zu erkennen, letzteres liest jeden Buchstaben einzeln, wodurch die Ergebnisse variieren. `Image_to_string` wurde zusätzlich auf das um 180° rotierte Bild angewendet, um die Orientierung zu ermitteln: Die Ergebnisse der Texterkennung beider Orientierungen wurden, nach der Bereinigung von Leerzeichen und Zeilenumbrüchen, anhand der Länge miteinander verglichen. Die Version, bei der mehr Text gefunden wurde, wurde als korrekt orientiert angenommen⁹. Diese Annahme folgt daraus, dass die Texterkennung unter erschwerten Bedingungen stattfindet und Buchstaben oft nicht erkannt werden. Werden also mehr Buchstaben erkannt, kann davon ausgegangen werden, dass die Erkennung insgesamt besser war. Eine solche Funktion ist zwar in Tesseract bereits grundsätzlich implementiert, da die Texterkennung hier aber ohnehin unter erschwerten Bedingungen und ohne erkennbare Wörter stattfand, wurde hier ein eigener Ansatz verwendet¹⁰. Die Ergebnisse der beiden Pytesseract-Funktionen aus den je vier Bildern wurden im Anschluss untereinander verglichen. Der längste gefundene Text wurde als Ergebnis übernommen. Beide Befehle wurden mit folgender Konfiguration ausgeführt: (1) Es wurde eine Whitelist verwendet, die auf das Schema der Tafeln zugeschnitten wurde. Gelistet wurden die Ziffern 0-9, die Buchstaben G,O,T,U und S sowie das Sonderzeichen /. (2) Als Sprache wurde ein eigenes Wörterbuch angegeben, das entsprechend der Tafeln nur wenige Worte enthält (US, GOT). (3) Der *OCR Engine Mode* (oem) wurde auf ein Neurales Netzwerk mit *Long Short-term Memory* [Hochreiter and Schmidhuber, 1997] festgelegt. (4) Für die *Page Segmentation* (psm) wurde ein vollautomatischer Modus, ohne weitere Vorgaben, gewählt. Dabei handelt sich sich jeweils um die Standardeinstellungen.

⁹Alternative wäre hier die Verwendung der *confidence* möglich gewesen, also der Wert, mit dem angegeben wird, für wie wahrscheinlich ein Neuronales Netz die angegebene Kategorisierung angibt, also wie sicher es bei diesem Ergebnis ist. Dieser Wert kann nicht bei allen Tesseract-Funktionen erhoben werden und wurde daher hier ausgeschlossen.

¹⁰U.a. liefert die Funktion `image_to_osd` von Pytesseract die Orientierung des gefundenen Textes; für dessen Anwendung befindet sich auf den Tafeln jedoch zu wenig Text.

2.6 Evaluation

Die Ergebnisse sowohl bei Tafelerkennung und -ausschnitt als auch bei der Texterkennung wurden durch statistische Analyse des erkannten Textes evaluiert. Dazu wurden die Texte auf einer Auswahl von 79 Tafeln transkribiert und in einer Liste abgelegt. Dabei handelt es sich um alle Tafeln aus dem Datensatz der Grabung 2011. In diesem Teildatensatz ist eine ausreichend große Vielfalt von Fotos mit und ohne Tafeln enthalten. Diese Liste wurde dann als Referenz (Grundwahrheit) für die Auswertung der erkannten Texte eingelesen. Als Maße für die Genauigkeit der Erkennung wurden die Maße *F-Score*, *Recall* und *Precision* verwendet [Klinke, 2017] [Ye et al., 2007]. *Recall* gibt dabei an, wie viele der Tafeln tatsächlich erkannt wurden:

$$\text{Recall} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalseNegatives}} \quad (1)$$

Die *Precision* gibt an, wie viele der erkannten Tafeln tatsächlich Tafeln sind:

$$\text{Precision} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalsePositives}} \quad (2)$$

Der F-Score erzeugt einen gewichteten, harmonischen Mittelwert aus diesen beiden Maßen, um die Qualität der Tafelerkennung in einem Wert zusammen zu fassen:

$$F = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (3)$$

Für die Evaluation der Texterkennung wurde das `ratio` der Funktion `sequence_matcher` verwendet, das ein Maß für die Ähnlichkeit zweier Textsequenzen berechnet. Es berechnet sich aus [Python-Documentation, 2021]:

$$\text{ratio} = 2 \times \frac{\text{Matches}}{\text{Length of both Sequences}} \quad (4)$$

Durch den stärkeren Einbezug der Länge der Sequenzen schneidet ein Text mit zu vielen erkannten Zeichen in der Tendenz schlechter ab als bei der Berechnung des *F-Scores*, was hier angemessen erschien. Die Option, hier dennoch den *F-Score* zu verwenden kann optional verwendet werden.

3 Ergebnisse

In diesem Kapitel wird zunächst das Gesamtergebnis präsentiert. Im Anschluss wird auf Besonderheiten bei Tafel- und Texterkennung eingegangen. Basis der Auswertung ist dabei ein Testdatensatz aus 292 Bildern, worunter sich 79 Bildern mit Tafeln befinden¹¹. Schließlich werden Versuche mit den Datensätzen anderer Projekte präsentiert.

3.1 Gesamtergebnis

Für das Gesamtergebnis wurden die Tafelausschnitte in den vier Varianten gemeinsam der Texterkennung unterzogen (Vgl. collection.csv). Der durchschnittliche *F-Score* (*Gesamt*) betrug 0.49. Die Durchschnittswerte der zeilenweisen (*Zeilen*) und buchstabenweisen (*Buchstaben*) Texterkennung lagen bei 0.51 bzw. 0.47. Die durch das Programm als bestes Ergebnis ausgewählten Zeichenketten (*Endauswahl*) erzielten einen *F-Score* von 0.61. Das theoretische Optimum (*Optimum*), also der Durchschnitt der tatsächlichen besten Ergebnisse pro Bild, lag bei 0.7 (Tabelle 1). 19 der insgesamt 412 Tafelausschnitte konnten nicht ausgelesen werden. Daraus resultierte eine nicht ausgelesene Tafel (letztlich eine Falsch-Negative), in den übrigen Fällen konnten die Tafeln über andere Bildausschnitte gelesen werden.

3.2 Tafelerkennung

Bei der Tafelerkennung zeigten die beiden Kontur-basierten Ansätze gute Ergebnisse, die nur in Nuancen voneinander abwichen. So erkannte der adaptive Ansatz aus 292 Bildern 133 mit Tafeln. Dass diese Zahl über der der 79 Tafeln lag, ist darin begründet, dass der äußere und der innere Tafelrand als Tafel identifiziert werden können, es also bis zu zwei richtige Erkennungen pro Bild geben kann¹². Zudem wurden 8 Falsch-Positive erkannt, Falsch-Negative gab es keine. Der *Re-*

¹¹Fotos, auf denen die Tafeln nur teilweise zu sehen sind, wurden hier als Fotos ohne Tafeln gewertet.

¹²Der Einfluss dieses Umstandes auf die Kennwerte ist so gering, dass er vernachlässigbar ist.

Verfahren	Buchstaben	Zeilen	Gesamt	Endauswahl	Optimum
Gesamt	0.47	0.51	0.49	0.61	0.7
Adaptive Hough	0.43	0.47	0.45	0.49	0.56
Adaptive Simple	0.52	0.57	0.54	0.61	0.64
Iterative Hough	0.44	0.47	0.45	0.47	0.52
Iterative Simple	0.51	0.56	0.54	0.57	0.6

Tabelle 1: Mittelwerte der Ergebnisse bei der Anwendung aller Verfahren gleichzeitig sowie der einzelnen Kombinationen aus Detektions- und Ausschnittverfahren. Angegeben sind die F-Werte für das **Buchstaben**verfahren (image to box), das **Zeilen**verfahren (image to text), der **Gesamtdurchschnitt** beider Verfahren, die durch den Algorithmus vorgenommene **Endauswahl** sowie das theoretische **Optimum**, also die tatsächliche beste Auswahl.

call lag damit, wie gewünscht, bei 1. Die *Precision* betrug 0.926, der *F-Score* damit 0.961. Beim iterativen Ansatz wurden 85 Tafeln erkannt. Da dieser Ansatz nur das wahrscheinlichste Ergebnis pro Bild ausgibt und hier keine Falsch-Negativen auftraten, lag die Zahl der Falsch-Positiven bei 6. Der *Recall* war also auch hier 1, die *Precision* 0.929. Der *F-Score* betrug 0.963. Die relativ hohe Zahl der Falsch-Positiven resultierte aus vier Fotos, auf denen Plakate abgebildet waren. Da Plakate alle Kriterien der Tafeln erfüllen – rechteckig, passendes Seitenverhältnis, mit Text beschrieben – konnte der Algorithmus weder in diesem Schritt noch später, bei der Texterkennung, diese Bilder aussortieren. Eine manuelle Entfernung der Bilder aus dem Datensatz, die sich auch in der Praxis als erster Schritt empfehlen würde, steigerte den *F-Score* auf 0.992 beim adaptiven und auf 0.987 beim iterativen Ansatz. Die Unterschiede zwischen beiden Ansätzen sind also gering, wobei die Anfälligkeit für Falsch-Positive im iterativen Ansatz etwas niedriger ist.

Weniger erfolgreich verlief die Detektion der Tafeln mit CNNs. Das Grabungsareal entspricht nicht den üblichen in einer Trainingsdatenbank enthaltenen Szenarien. Einzelne Objekte auf dem Gelände, so z.B. Rohrleitungen, der Messstab und der Nordungspfeil, wurden oft als Objekte identifiziert(Abb.21), allerdings mit wechselnder Kategorisierung. Die Tafeln selbst wurden nur in Einzelfällen erkannt, entweder als Schere oder als Sportball (Abb.22. Eine Systematik war nicht festzustellen. Dieser Ansatz muss also als gescheitert betrachtet werden.

Bei der Auswertung der Texterkennung nach Detektionsverfahren ergaben sich Unterschiede: Beim adaptiven Verfahren (ausgewertet mit beiden Schnitt-



Abbildung 21: Rohrleitung, Messstab und Nordungspfeil werden erkannt und als Sandwich, Tennisschläger und Surfbrett kategorisiert.

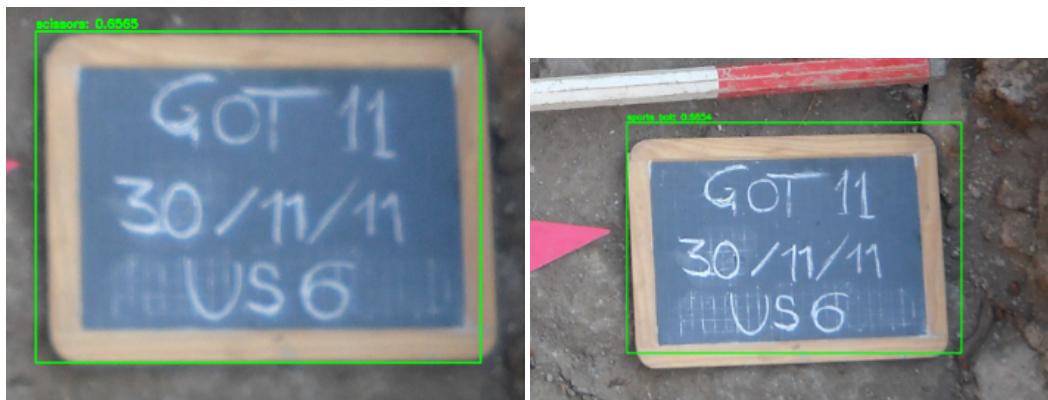


Abbildung 22: Erkennung und Kategorisierung der Tafeln als Schere oder Sportball.

verfahren) lag das *Ratio* der *Endauswahl* bei 0.59, das theoretische Optimum bei 0.67 (Vgl. adaptive only.csv). Das iterative Verfahren kam auf ein *Ratio* von 0.56 bei einem Optimum von 0.65 (Vgl. iterative only.csv). Beim nächsten Schritt, dem Schnittverfahren, variierten die Ergebnisse deutlicher. So erzielte die *Endauswahl* des Hough Crop-Verfahrens (iterativer und adaptiver Ansatz parallel) ein *Ratio* von 0.49, das theoretische Optimum betrug 0.59 (Vgl. hough only.csv). Im Gegensatz dazu lag die *Endauswahl* des Simple Crop bei 0.62 und somit noch über der Gesamtauswertung. Das theoretische Optimum lag mit 0.67 etwas unter der Gesamtauswertung (Vgl. simple only.csv). Als Einzelverfahren war die Kombination Adaptive Simple Crop am besten, mit einem *Gesamt*-Durchschnitt von 0.61 und einem theoretischen Optimum von 0.64 (Vgl. crop adaptive sim-

Verfahren	Buchstaben	Zeilen	Gesamt	Endauswahl	Optimum
Gesamt	0.47	0.51	0.49	0.61	0.7
Adaptive (Hough + Simple)	0.47	0.51	0.49	0.59	0.67
Iterative (Hough + Simple)	0.48	0.51	0.5	0.56	0.65
Hough (Adaptive + Iterative)	0.43	0.47	0.45	0.49	0.59
Simple (Adaptive + Iterative)	0.52	0.56	0.54	0.62	0.67

Tabelle 2: Mittelwerte der Ergebnisse bei der Anwendung aller Verfahren gleichzeitig sowie der beiden Detektionsverfahren und der beiden Schnittverfahren untereinander. Angegeben sind die *Ratios* für das **Buchstabenverfahren** (image to box), das **Zeilenverfahren** (image to text), der **Gesamtdurchschnitt** beider Verfahren, die durch den Algorithmus vorgenommene **Endauswahl** sowie das theoretische **Optimum**, also die tatsächliche beste Auswahl.

ple.csv).

3.3 Texterkennung

Im Bereich der Texterkennung ließ sich ein Unterschied zwischen `Image to Box`, also der Buchstabenerkennung, und `Image to Text`, der Zeilenerkennung, feststellen. So war das mittlere *Ratio* bei der Buchstabenerkennung um 0.03 bis 0.05 niedriger als bei der Zeilenerkennung (Vgl. `collection.csv` u.a.). Im Einzelfall variierten die Ergebnisse jedoch stark, sodass die Buchstabenerkennung deutlich bessere Ergebnisse als die Zeilenerkennung liefern kann, was sich positiv auf den *bestguess* auswirkt. Wie bereits vorgestellt wurde für die Texterkennung ein Wörterbuch und eine Whitelist verwendet. In der Auswertung zeigte sich, dass vor allem die Whitelist das Ergebnis deutlich verbessert. Die Anwendung der Texterkennung ohne Wörterbuch, aber mit Whitelist, auf die mit `Adaptive Simple Crop` erzeugten Tafelausschnitte erzielte die gleichen Ergebnisse wie der Durchlauf mit Wörterbuch, also einen Durchschnitt von 0.61 bei der *Endauswahl* (Vgl. `no dic.csv`). Wurden sowohl Whitelist als auch Wörterbuch entfernt, sank das durchschnittliche *Ratio* auf 0.26, wobei die Buchstabenerkennung 0.25 erzielte, die Zeilenerkennung nur 0.01 (Vgl. `no config no dic.csv`). Schaltete man nur das Wörterbuch dazu, verbesserte sich der Wert der Zeilenerkennung auf 0.26 (Vgl. `dic on`).

ly.csv). Festzustellen war auch, dass die *Endauswahl* nicht zu Ergebnissen unterhalb es Durchschnitts führte, andererseits aber für deutliche Verbesserungen sorgen konnte. Deutlich wurde das in der Gesamtauswertung aller Ansätze, bei der die Differenz zwischen dem durchschnittlichen *Ratio* der beiden Texterkennungsverfahren (0.49) und dem durchschnittlichen *Ratio* der *Endauswahl*(0.61) 0.12 Punkte betrug. Allerdings bestand auch ein Unterschied von 0.9 zum theoretischen Optimum (0.7) (Vgl. collection.csv).

3.4 Weitere Tafeln

Bei der Anwendung des Algorithmus auf die projektfremden Tafeln der Gruppe Terrestrische Ökohydrologie der FSU Jena sowie der späteren Grabungen am Kapitol durch das Deutsche Archäologische Institut ergaben sich folgende Ergebnisse: Bei den Tafeln der Ökohydrologie Jena wurde eine Stichprobe aus 40 Fotos mit Tafeln verwendet. Von diesen wurden 9 korrekt erkannt. Auf 16 Fotos wurde fälschlicherweise ein beiliegendes Maßband erkannt¹³. Der *Recall* beträgt hier 0.225, die *Precision* 0.36. Der *F-Score* liegt bei 0.277. Diese Ergebnisse wurden mit dem iterativen Verfahren erzielt. Mittels adaptivem Ansatz konnten keine Tafeln ermittelt werden. Die Texterkennung verlief – abgesehen von Zahlen auf dem Maßband – ergebnislos. Aus den Fotografien der Grabungen des DAI wurden 35 Bilder mit Tafeln darauf ausgewählt. Mit dem adaptiven Verfahren wurden 26 Tafeln erkannt, dazu gab es eine Falsch-Positive. Daraus ergab sich ein *Recall* von 0.743, eine *Precision* von 0.963 und ein *F-Score* von 0.838. Das iterative Verfahren erkannte 25 Tafeln ohne Falsch-Positive. Der *Recall* betrug hier 0.714, die *Precision* 1 und der *F-Score* 0.833. Für die Texterkennung wurde die Whitelist modifiziert, sodass alle Ziffern und alle Großbuchstaben enthalten waren, wie es der Beschriftung der Tafeln entspricht. Das *Ratio* der Zeilenerkennung lag bei 0.86, das der Buchstabenerkennung bei 0.9. Das theoretische Optimum lag bei 0.91. Die *Endauswahl* lag mit 0.83 deutlich unter diesen Werten. Ohne Whitelist konnten lediglich Werte von 0.04 bei der Zeilenerkennung, bzw. 0.64 bei der Buchstabenerkennung erzielt werden.

¹³Hier ist technisch gesehen kein unerwünschtes Ergebnis erzielt worden: Auch das Maßband enthält rechteckige Felder mit passendem Seitenverhältnis und Zahlen darauf.

4 Diskussion

Durch die Ergebnisse konnte gezeigt werden, dass die hier gewählten Ansätze grundsätzlich funktionieren. Die Tafelerkennung ist relativ robust in der Lage, Tafeln zu erkennen, die bestimmte Kriterien erfüllen und auch die Texterkennung ist grundsätzlich trotz der niedrigen Qualität des Ausgangsmaterials möglich.

4.1 Tafelerkennung

Probleme bei der Tafeldetektion entstehen dann, wenn die Tafel nicht klar als Rechteck erkannt werden kann. Durch den doppelten, breiten Rahmen der Projekttafeln ist dieser Fall nie gegeben. In den wenigen Fällen, in denen der Rahmen teilweise durch ein Objekt verdeckt ist, bleibt der innere Rand davon unberührt und ermöglicht so die Erkennung der Schieferfläche im Kontrast zum Rahmen. Auch der umgekehrte Fall, dass ein Objekt im Hintergrund dem Tafelrahmen so ähnlich ist, dass es als Teil der gleichen Kontur erkannt wird, kann so kompensiert werden (Abb. 23). Die Falsch-Negative der Grabungen des Deut-

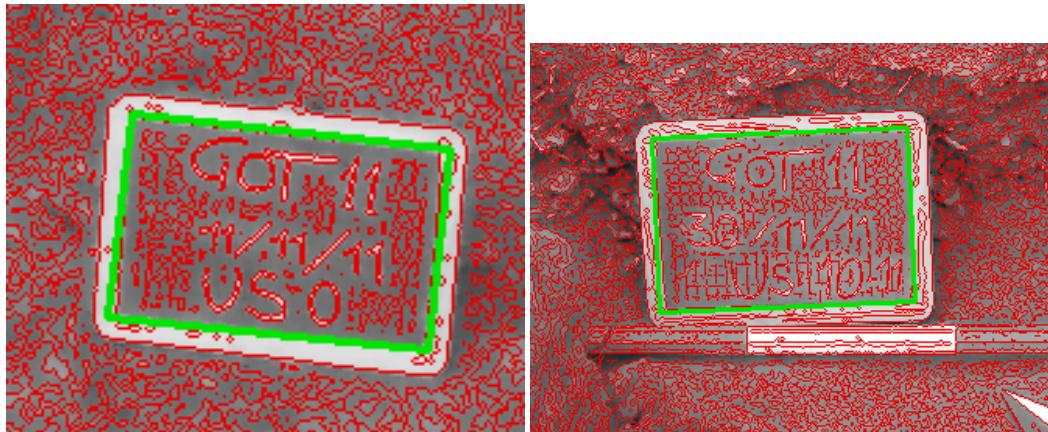


Abbildung 23: Ein Stein im Hintergrund wurde aufgrund der Farbe als Teil des Tafelrahmens erkannt (links) und ein Messstab überlagert den Tafelrand (rechts). Beide Tafeln wurden aufgrund des inneren Rahmens korrekt erkannt.

schen Archäologischen Instituts sind alle auf das Verdecken von Teilen des Rahmens zurückzuführen. Ähnlich verhält es sich bei den Fotos der Ökohydrologie. Hier ist die Tafel komplett rahmenlos, was die Unterscheidung vom Hin-

tergrund zusätzlich erschwert (Abb. 24). Um das Problem der überlagerten Um-



Abbildung 24: Helle Farbe und Lichtspiegelung erschweren die Unterscheidung vom Hintergrund (links) und Objekte überlagern den Tafelrand (rechts). Beide Tafeln wurden nicht erkannt, wie an der fehlenden grünen Umrandung zu sehen ist.

randung zu lösen könnte der Anteil, den die Fläche der Kontur an der Fläche des umgebenden minimalen Rechtecks hat, herabgesetzt werden. Dadurch könnte natürlich auch der Anteil der Falsch-Positiven steigen. Außerdem könnten Maßnahmen wie eine Konvexe Hülle oder *Contour Approximation* angewendet werden [OpenCV-Documentation, 2021f], die die Rekonstruktion eines Rechteckes zulassen, wenn Teile davon überlagert sind. Der fehlende Kontrast zum Hintergrund könnte mit dem Untersuchen weiterer Farbkanäle ausgeglichen werden. Eine Anpassung ist in jedem Falle nötig, aber auch möglich.

4.2 Texterkennung

Eine größere Herausforderung stellt die Schrift dar. Die Detektion findet in einer natürlichen Umgebung statt. Normalerweise müssen Textfelder in solchen Fällen erst aufwändig gesucht werden [Ye et al., 2007], was in diesem Fall durch die Eingrenzung auf die Tafeln gelöst wird. Die übrigen Verhältnisse bleiben jedoch erhalten: Wechselnde Lichtverhältnisse, Rotationen der beschriebenen Fläche um mehrere Achsen verschlechtern die Ergebnisse [Chen et al., 2004] [Tesseract Documentation, 2021a]. Grenzwerte und das Hough-Verfahren sollten hier die Umstände verbessern, was jedoch nur bedingt zum Erfolg führte. Das lässt sich daran erkennen, dass die Er-

gebnisse des Hough-Verfahrens im Schnitt schlechter sind als die des Simple Crop-Verfahrens. Ursache ist hier, dass der Tafelrand durch Abnutzungerscheinungen, seine gerundete Form und Unschärfe durch die niedrige Auflösung, über keine klare Kante verfügt und daher pro Kante nicht eine, sondern oft mehrere Linien erkannt werden, die beinahe, aber nicht ganz, parallel verlaufen (Vgl. Abb. 18). Das verschiebt die erkannten Eckpunkte minimal, führt zu einer Verzerrung der Tafel bei der Projektion und somit auch des Schrift. Das Ziel einer Vereinheitlichung des Schriftbildes konnte so also nicht erreicht werden. Eine zusätzliche Erschwernis entsteht dadurch, dass, je nach Lichteinfall, ein Gitterlinienmuster auf der Tafel, dass der Orientierung beim Schreiben dienen soll, sichtbar wird und die Buchstaben überlagert, was unter kontrollierten Bedingungen entweder vermieden oder zumindest leichter aus dem Bild entfernt werden könnte¹⁴. Der Versuch der Schärfung des Bildes durch eine Unscharfmaskierung¹⁵ brachte keine Verbesserung der Ergebnisse, da dadurch das Rauschen in den Bildern verstärkt und die unerwünschten Gitterlinien betont wurden. Aus dem gleichen Grund erwies sich auch die Anwendung eines bilateralen Filters [Tomasi and Manduchi, 1998] nicht als hilfreich. Generell ist Handschrift problematisch: Da sich Handschriften immer voneinander unterscheiden, müssen für eine gute Erkennung Neuronale Netzwerke auf eine Handschrift spezialisiert werden [Hallale and Salunke, 2013]. Dazu lag in diesem Fall deutlich zu wenig Trainingsmaterial vor. Auch die Überlagerung von Buchstaben, die bei gedruckten Schriften in der Regel nicht vorkommt, erschwert die Erkennung zusätzlich. Dass unter diesen Umständen relativ gute Ergebnisse erzielt werden konnten, liegt vermutlich in der sauberer Handschrift in Druckbuchstaben, die auf den Tafeln des Testdatensatzes zu sehen ist. Theoretisch gibt es hier also Verbesserungspotential, durch die nötige Datenmenge für das Training eines Modells erscheint das jedoch nicht realistisch. Es stellt sich aber durchaus die Frage, ob ein Modell, das über große Mengen von Grabungsfotos aus mehreren Projekten trainiert wurde, nicht bessere Ergebnisse lie-

¹⁴Wären die Gitterlinien auf allen Tafeln zu sehen, hätten sie als Orientierung für die Entzerrung der Tafel dienen können. So sind sie jedoch als Störung und nicht als Hilfsmittel zu betrachten.

¹⁵Bei der Unscharfmaskierung wird das Originalbild mit einer unscharfen Kopie verglichen. Überschreitet die Differenz der beiden Bilder an einem Pixel einen Grenzwert, wird an dieser Stelle das unscharfe Bild vom scharfen Bild subtrahiert. So entsteht ein Schärfungseffekt [GIMP Documentation, 2021].

fern könnte. Immerhin bestünde somit eine Spezialisierung auf Handschrift im allgemeinen sowie auf die Erzeugnisse von Schreibwerkzeugen wie Kreide oder Filzstift. Dieser Ansatz, auf einer breiten Datenbasis zu trainieren und für mehrere Schriften gute Ergebnisse zu erzielen, die dann automatisiert verbessert werden können (sogenannte *mixed models*), statt sich auf eine einzelne Schrift zu spezialisieren, wurde von Springmann et al. für historische Druckerzeugnisse erfolgreich angewendet [Springmann et al., 2016]. Das führt zum dritten großen Problem der Schrift: Die Kreide. Im Feld wurde die Tafel nicht immer gründlich gereinigt. Verwischte Kreidespuren um die Schrift herum sowie das ungleichmäßige Bild der Schrift selbst machen eine gängige Maßnahme – Binarisierung vor der Texterkennung [Hamad and Kaya, 2016] [Hallale and Salunke, 2013] – unmöglich¹⁶. Teile der Buchstaben würden durch einen festen Grenzwert entfallen, im Gegenzug besteht die Gefahr großer weißer Flächen in den verwischten Arealen. Der adaptive Grenzwert dagegen, wie ihn Shah und Gokani vorschlagen [Shah and Gokani, 2014], führt zu vielen Störungen und Artefakten im verwischten Bereich. Diese Faktoren wirken sich negativ auf die Texterkennung auf und ergänzen sich gegenseitig derart, dass eine Maßnahme gegen ein Problem ein anderes verstärkt (Abb.25). Ein ebenfalls zu Verbesserung der Texterkennung verwendetes Verfahren, die Erosion¹⁷ bis hin zur Skelettisierung¹⁸ [Hamad and Kaya, 2016] [Hallale and Salunke, 2013] der Schrift, scheiterte ebenfalls an dem durch die Kreide sehr unregelmäßigen Schriftbild. Im Verlauf der Texterkennung hat sich vor allem die Whitelist als wirkungsvolles Werkzeug erwiesen. Je weniger Zeichen ein Datensatz enthält, desto präziser kann die Whitelist ausgestaltet werden und desto besser werden die Ergebnisse [Feldmann, 2001, p. 107]. In der praktischen Durchführung der Texterkennung wurden Zeichen, die auf zwei der Tafeln vorkommen, nicht in die Whitelist aufgenommen¹⁹. Das hat den Hintergrund, dass die vor allem für die Sortierung der Tafeln wichtigen Informationen, also Kampagne, Datum und stra-

¹⁶Die Binarisierung des Inputs ist Teil der Tesseract-Pipeline und wird daher ohnehin ausgeführt [Agnes Forsberg and Melvin Lundqvist, 2020]. Eine vorherige, an das Dokument angepasste Binarisierung wird jedoch von empfohlen. Entsprechende Versuche verschlechterten das Ergebnis jedoch deutlich.

¹⁷Reduktion der Breite einer Linie.

¹⁸Reduktion der Linien auf die Breite von einem Pixel.

¹⁹„GIARDINO EX OSP TEUT“, Garten des ehemaligen deutschen Hospitals. Die Whitelist enthielt daraus nur die Buchstaben G, O, S, T und U.



Abbildung 25: Auf diesem Bild sind die Probleme durch die Kreide deutlich sichtbar: (1) Unterschiedliche Schriftbreiten, (2) Lücken und verwischte Bereiche bei O und T, (3) verwischte Kreide im unteren und linken Bildbereich, (4) Gitterlinien, die durch die verwischte Kreide besonders hervorgehoben werden.

tigrafische Einheit, darin ohnehin nicht enthalten waren, die Bedingungen für die Extraktion dieser Informationen aus den anderen Tafel aber deutlich schlechter geworden wären. Komplett ohne Whitelist zu arbeiten hat sich in diesem Projekt nicht empfohlen, da eine Vielzahl von Störobjekten im Hintergrundrauschen als Sonderzeichen interpretiert wurden. Der Einfluss des Wörterbuchs war, wie gezeigt werden konnte, insgesamt eher gering. Ein positiver Einfluss zeigte sich bei der Zeilenerkennung. Hier wirkt sich eine größere Vielfalt auf den Tafeln vor allem auf die Komplexität der Erstellung des Wörterbuchs aus. Die Voreinstellungen von Tesseract zur *Page Segmentation*, also zur Erkennung des Layouts und der Anordnung der Schrift sowie der Identifikation einzelner Textfelder, wurden übernommen, da es in einem empirischen Vergleich aller 13 verschiedenen Segmentierungsverfahren keine Abweichungen gab. Die Bestimmung des besten Ergebnisses bei der Texterkennung funktionierte bei den hier verwendeten Tafeln gut, aber nicht optimal. Das ist darin erkennbar, dass der Durchschnittswert der als bestes Ergebnis betrachteten Texte höher liegt als der Gesamtdurchschnitt und oft nahe dem theoretischen Optimum liegt. Wie bereits beschrieben liegt der in solchen Fällen verwendete Wert der *confidence* nicht immer vor und kann bei Tesseract nur für die Zeilenerkennung verwendet werden, da er nur für Wörter und nicht für Buchstaben erhoben wird. Eine Vergleichbarkeit von Zeilen- und Buchstabenerkennung wäre somit weggefallen. Damit wäre auch der Mehrwert der parallelen

Anwendung beider Verfahren hinfällig gewesen. Im eigentlichen Anwendungsfall, den Fotografien der Altgrabungen auf dem Kapitol, erfüllt das Kriterium also seinen Zweck. Bei der Anwendung auf die Tafeln der neueren Grabungen konnten anhand dessen keine Aussagen über die Qualität der Texterkennung getroffen werden, da immer alle Zeichen erkannt wurden und somit die Zeichenzahl keine Aussagekraft mehr hatte. Das Grunddilemma der Vergleichbarkeit der Methoden blieb hier aber bestehen, da die Buchstabenerkennung ohne Wörterbuch bereits hervorragende Ergebnisse liefern konnte, während die Zeilenerkennung die richtige Anzahl an Zeichen, aber nur selten das Zeichen selbst erkennen konnte. Die Evaluation schließlich funktionierte wie gewünscht. Da die hier verwendeten Kennwerte die Reihenfolge der Buchstaben nicht berücksichtigen, wäre eine Ersetzung oder Ergänzung des *Ratios* durch die Levenshtein-Distanz möglich. Diese wird durch die Zahl der nicht erkannten (zu ergänzenden), falsch erkannten (auszutauschenden) und fälschlicherweise erkannten (zu löschen) Zeichen im erkannten Text ermittelt [Levenshtein, 1966]. Schließlich ist zu erwähnen, dass auch Tesseract selbst austauschbar ist. Unter schwierigen Bedingungen und bei Handschriften schneidet die Engine zwar gleich gut oder besser ab als die Konkurrenz [Agnes Forsberg and Melvin Lundqvist, 2020]. Es gibt jedoch eine Reihe von Engines, die speziell für das Erkennen von historischen Schriften entwickelt wurden und flexibler sind, wenn keine Wörterbücher vorliegen. Zu nennen wären hier vor allem OCropus²⁰, Kraken²¹ und Calamari OCR²². Das auf einzeilige, alphanumerische Codes spezialisierte SwiftOCR²³ könnte hier auch gute Ergebnisse erzielen, ist jedoch auf ein Betriebssystem beschränkt.

4.3 Empfehlung

Aus diesen Ergebnissen lassen sich klare Empfehlungen für Tafeln ableiten, die im Rahmen solcher Projekte verwendet und im Anschluss automatisiert ausgelesen werden sollen: (1) Die Tafel sollte intakt sein. (2) Sie sollte über einen breiten Rahmen verfügen. (3) Der Rahmen, die beschriftete Fläche und die Buchstaben soll-

²⁰<https://github.com/ocropus/>

²¹<https://github.com/mittagessen/kraken>

²²<https://github.com/Calamari-OCR>

²³<https://github.com/NMAC427/SwiftOCR>

ten jeweils den maximalen Kontrast zueinander haben. (4) Die Buchstaben sollten gestanzt oder gedruckt sein. (5) Die Anordnung der Buchstaben sollte möglichst geordnet erfolgen, wobei Zeilenverläufe exakt eingehalten werden. (6) Die Tafeln sollten nicht durch Gegenstände verdeckt werden.

5 Fazit

In dieser Arbeit konnte gezeigt werden, dass das grundsätzliche Prinzip, Tafeln auf Grabungsfotos mit Methoden der Computer Vision zu erkennen, funktioniert. Weiter konnte gezeigt werden, dass das Auslesen durch eine OCR-Engine, in diesem Fall Tesseract, ebenfalls möglich ist. Problematisch erwiesen sich Faktoren wie die Lage der Tafel, die Beleuchtung und die von Hand ausgeführte Beschriftung der Tafeln mit Kreide. Bei der Texterkennung erwies sich letzteres als Hauptursache für schlechte Ergebnisse. Für eine Automatisierung des Prozesses, die Metadaten aus den Tafeln zu extrahieren, sind die Ergebnisse nicht ausreichend. Durch die Selektion der Fotos mit Tafeln sowie das Bereitstellen des erkannten Textes, der im Anschluss manuell korrigiert wird, kann der Arbeitsaufwand jedoch deutlich reduziert werden. Zudem konnten Empfehlungen abgegeben werden, wie Fotografien zur Metadatenerfassung mittels Tafeln in Zukunft durchgeführt werden kann, um bessere Ergebnisse zu erzielen. In der Zukunft könnte der Detektionsprozess optimiert werden. So hat die Erkennung von Tafeln, deren Rand teilweise verdeckt ist, Verbesserungspotential. Weitere Untersuchungen zu anderen Tafel und Datensätzen wären lohnenswert.

Literatur

- [Adamek and O'Connor, 2003] Adamek, T. and O'Connor, N. (2003). Efficient contour-based shape representation and matching. pages 138–143.
- [Agnes Forsberg and Melvin Lundqvist, 2020] Agnes Forsberg and Melvin Lundqvist (2020). A comparison of ocr methods on natural images in different image domains.
- [Canny, 1986] Canny, J. (1986). A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-8:679 – 698.
- [Chen et al., 2004] Chen, X., Yang, J., Zhang, J., and Waibel, A. (2004). Automatic detection and recognition of signs from natural scenes. *IEEE Transactions on Image Processing*, 13(1):87–99.
- [CIPA Standardization Committee, 2020] CIPA Standardization Committee (2020). Exif 2.32 metadata for xmp.
- [Dally, 2021] Dally, O. (2021). Kapitol.
- [Danti et al., 2014] Danti, A., Marra, F., Meneghini, R., Presicce, C. P., Ungaro, L., Vitti, M., Milella, M., Bernardini, C., Giuffré, M. T., Vigliarolo, P., Massera, M., and Rivaroli, L. (2014). Regione viii. *Bullettino della Commissione Archeologica Comunale di Roma*, 115:244–289.
- [Dataset, 2021] Dataset, C. (2021). Coco.
- [Eikvil, 1993] Eikvil, L. (1993). Ocr - optical character recognition.
- [Feldmann, 2001] Feldmann, B. (2001). Ocr von handschriften. *Fundus - Forum für Geschichte und ihre Quellen*, 1:107–141.
- [GIMP Documentation, 2021] GIMP Documentation (2021). Sharpen (unsharp mask).

- [Hallale and Salunke, 2013] Hallale, S. B. and Salunke, G. D. (2013). Offline handwritten digit recognition using neural network. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 2:4373–4377.
- [Hamad and Kaya, 2016] Hamad, K. and Kaya, M. (2016). A detailed analysis of optical character recognition technology. *International Journal of Applied Mathematics, Electronics and Computers*, 4:244–249.
- [Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.
- [Hough, 1962] Hough, P. V. C. (1962). Method and Means for Recognizing Complex Patterns. *U.S. Patent*, (3,069,654).
- [Jaccard, 1902] Jaccard, P. (1902). Tois de distribution florale dans la zone alpine. *Bulletin de la Société Vaudoise des Sciences Naturelles*, (38):72.
- [Klinke, 2017] Klinke, H. (2017). Information retrieval. In Jannidis, F., Kohle, H., and Rehbein, M., editors, *Digital Humanities. Eine Einführung*, pages 268–278. J.B. Metzler Verlag, Stuttgart.
- [Levenshtein, 1966] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710. Doklady Akademii Nauk SSSR, V163 No4 845-848 1965.
- [OpenCV-Documentation, 2021a] OpenCV-Documentation (2021a). Denoising.
- [OpenCV-Documentation, 2021b] OpenCV-Documentation (2021b). Geometric image transformations.
- [OpenCV-Documentation, 2021c] OpenCV-Documentation (2021c). Image filtering.
- [OpenCV-Documentation, 2021d] OpenCV-Documentation (2021d). Miscellaneous image transformations.

- [OpenCV-Documentation, 2021e] OpenCV-Documentation (2021e). Object detection.
- [OpenCV-Documentation, 2021f] OpenCV-Documentation (2021f). Structural analysis and shape descriptors.
- [O’Shea and Nash, 2015] O’Shea, K. and Nash, R. (2015). An introduction to convolutional neural networks.
- [Python-Documentation, 2021] Python-Documentation (2021). difflib — helpers for computing deltas.
- [READ-COOP, 2021] READ-COOP (2021). Transkribus.
- [Redmon et al., 2015] Redmon, J., Divvala, S. K., Girshick, R. B., and Farhadi, A. (2015). You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640.
- [Redmon, 2021] Redmon, J. C. (2021). Yolo.
- [Schürholz and Spitzner, 2019] Schürholz, M. and Spitzner, E.-C. (2019). Hardware für ki. In Wittpah, V., editor, *Künstliche Intelligenz*, pages 36–47. Springer, Berlin.
- [Shah and Gokani, 2014] Shah, J. and Gokani, V. (2014). A simple and effective optical character recognition system for digits recognition using the pixel-contour features and mathematical parameters. *International Journal of Computer Science and Information Technologies*, 5:6827–6830.
- [Shaw and Barnes, 2006] Shaw, D. and Barnes, N. (2006). Perspective rectangle detection.
- [Siltanen, 2012] Siltanen, S. (2012). *Theory and applications of marker based augmented reality*. PhD thesis.
- [Springmann et al., 2016] Springmann, U., Fink, F., and Schulz, K. (2016). Automatic quality evaluation and (semi-) automatic improvement of mixed models for ocr on historical documents.

- [Suzuki and Abe, 1985] Suzuki, S. and Abe, K. (1985). Topological Structural Analysis of Digitized Binary Images by Border Following. *COMPUTER VISION, GRAPHICS, AND IMAGE PROCESSING*, (30):32–46.
- [Tesseract Documentation, 2021a] Tesseract Documentation (2021a). Improving the quality of the output.
- [Tesseract Documentation, 2021b] Tesseract Documentation (2021b). Release notes.
- [Tomasi and Manduchi, 1998] Tomasi, C. and Manduchi, R. (1998). Bilateral filtering for gray and color images. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pages 839–846.
- [Ye et al., 2007] Ye, Q., Jiao, J., Huang, J., and Yu, H. (2007). Text detection and restoration in natural scene images. *Journal Of Visual Communiucation and Image Representation*, 18:504–513.