Technical Report TR-2006-001, Mathematics and Computer Science Department, University of Mannheim, January 2006
by Stephan Kopf, Manuel Knaus

REIHE INFORMATIK
TR-2006-001

Verbesserung der Qualität von historischen Filmen

Stephan Kopf, Manuel Knaus
University of Mannheim
– Fakultät für Mathematik und Informatik –
Praktische Informatik IV
A5, 6
D-68159 Mannheim, Germany

Verbesserung der Qualität von historischen Filmen

Stephan Kopf, Manuel Knaus
Dept. of Computer Science IV, University of Mannheim, Germany
kopf@informatik.uni-mannheim.de

Zusammenfassung

Die UNESCO hat schon früh erkannt, dass historische Filme einen wichtigen Baustein in der Erhaltung des kulturellen Erbes darstellen. Durch eine Digitalisierung können diese für die Zukunft erhalten werden, ohne dass Filme durch Materialermüdung der Filmrollen bzw. Bänder Schaden nehmen. Viele der historischen Aufnahmen sind durch Abspielen oder Lagerung bereits deutlich beschädigt. Dies macht sich beispielsweise durch helle Linien bemerkbar, die durch Beschädigungen wie Kratzer auf der Filmrolle entstanden sind. Außerdem weisen einige dieser Aufnahmen störende Helligkeitsschwankungen auf.

In diesem Bericht werden Algorithmen zur Erkennung und Behebung solcher Fehler in historischen Schwarzweißfilmen vorgestellt. Dabei handelt es sich um die Erkennung und Beseitigung von horizontalen Störlinien, um die Helligkeits- und Kontrastkorrektur sowohl bei starken Helligkeitsschwankungen als auch bei überdunkelten oder überhellten Sequenzen, sowie um die Entfernung von Verwackelungen bei Kameraeinstellungen.

Die einzelnen Algorithmen werden separat vorgestellt und erläutert. Die Umsetzung in ein lauffähiges C++ Programm und die Einbindung in eine vorhandene Multimediabibliothek wird erklärt. Schließlich wird ein Ausblick auf weitere Einsatzmöglichkeiten der Algorithmen sowie Ansatzpunkte für weitere Forschungsmöglichkeiten in diesem Gebiet gegeben.

1 Einführung

Nachdem die UNESCO erkannt hat, dass es sich bei historischem Filmmaterial um ein schützenswertes Gut handelt, das für die Zukunft erhalten werden muss, ist das digitale Archivieren von Filmen in den letzten Jahren immer stärker zur wissenschaftlichen Disziplin geworden [4]. Auch die EU baut große digitale Archive mit europäischem Kulturgut auf und hat mehrere Projekte am Laufen, die sich mit der Indizierung und Verbesserung der archivierten Filme beschäftigen [5].

Die Herausforderung besteht in der Digitalisierung der Schwarz-Weiß-Filme, die auf Filmrollen- oder Bändern aufgenommen wurden. Durch die jahrelange Lagerung beginnt das Filmmaterial zu zerfallen und muss schnellstmöglich auf digitale, verlustfreie Träger gespeichert werden. Historische Filme haben Schwarz-Weiß-Filmmaterial verwendet. Dieses bestand aus drei Schichten, jedoch nur aus einer Farbschicht. Ein Bild entsteht also aus den Helligkeiten Weiß, Schwarz und ihren Mischungen. Der Schwarz-Weiß-Film war ursprünglich mit Silbernitrat beschichtet. Er bestand aus einer lichtempfindlichen Schicht, die ihrerseits aus einer retuschierbaren Gelatine-Schutzschicht und Emulsionschichten, einem Schichtträger sowie einer Lichthofschutzschicht bestand.

Nach der Digitalisierung von historischen Filmen stellt man leider oft fest, dass die Qualität des Filmmaterials zu wünschen übrig lässt. Im Folgenden werden nun die einzelnen Fehler und Fehlerquellen beschrieben, die zu dieser Qualitätsminderung führen. Helligkeitsschwankungen sowie zu dunkle oder zu helle Sequenzen oder Einzelbilder können für den Betrachter sehr störend sein. Diese Helligkeitsschwankungen können mehrere Ursachen haben:

- Durch Schmutz oder Dreck bei der Lagerung der Filmrollen oder durch den Digitalisierungsprozess kann es zu Störungen in der Helligkeit in einzelnen Sequenzen eines Filmes kommen.
- Zu dunkle oder zu helle Bilder können auch aus Schimmelbefall der Filmrollen bei feuchter Lagerung resultieren.
- Flackern, also die globale Änderung der Helligkeit zwischen Einzelbildern, auch Frames genannt, entstand meist schon bei der Aufnahme von Schwarz-Weiß-Filmen durch die mangelhafte damalige Technik [7].

Oft beobachtet man auch eine horizontale, zum Teil flackernde weiße Linie über weite Strecken eines Films. Diese entsteht durch Kratzer, die beim Transport der Rolle im Abspielgerät, beispielsweise durch ein Malteserkreuzgetriebe, oder schon bei der Entwicklung entstanden sind [7]. Verwackelte Bilder ent-

stehen durch ungenaue mechanische Transportvorrichtungen bei den Abspielgeräten oder bereits bei der Aufnahme [7].

Im Folgenden wird nun erarbeitet, wie typische Fehler von digitalisierten Schwarz-Weiß-Filmen erkannt und behoben werden können. Ob jede dieser Behebungsmöglichkeiten bei einem einzelnen Film auch tatsächlich angewendet werden soll, liegt in der Entscheidung des Betrachters bzw. des Historikers, der den Film für ein Archiv digitalisiert. Da jedoch eine manuelle Kontrolle jedes einzelnen überarbeiteten Films sehr teuer ist, gerade wenn der Optimierungsprozess im Rahmen einer größeren Archivierung stattfindet, sind die entwickelten Algorithmen standardmäßig darauf ausgelegt, generell nach allen Fehlern zu suchen, auch wenn diese in einem einzelnen Film gar nicht vorkommen.

Das entwickelte C++ Programm wird in die *Automatic-Movie-Content-Analysis-Bibliothek* (*MoCA*) des Lehrstuhls für Praktische Informatik IV der Universität Mannheim eingebunden. Diese Bibliothek bietet bereits diverse Möglichkeiten der Manipulation von Einzelbildern, Videosequenzen, Audiotracks sowie Verknüpfungen, wie z.B. automatische Videobearbeitung und Inhaltsanalyse [6]. Diese Bibliothek stellt auch notwendige Methoden für das hier entwickelte Programm zur Verfügung, beispielsweise das Einlesen und Ausgeben eines Videos.

Für jede Art von Fehler, die in diesem Bericht analysiert wird, wird auf dessen Entstehung sowie typische Merkmale eingegangen. Gegebenenfalls werden Verfahren anderer Autoren vorgestellt, die sich mit gleichen oder ähnlichen Fehlern beschäftigt haben. Daraufhin wird der Algorithmus zur Erkennung des Fehlers vorgestellt und erläutert. Schließlich wird der Optimierungsalgorithmus dargestellt und dessen Potenzial und Grenze aufgezeigt. Abbildung 1 zeigt die generelle Vorgehensweise des Programms.

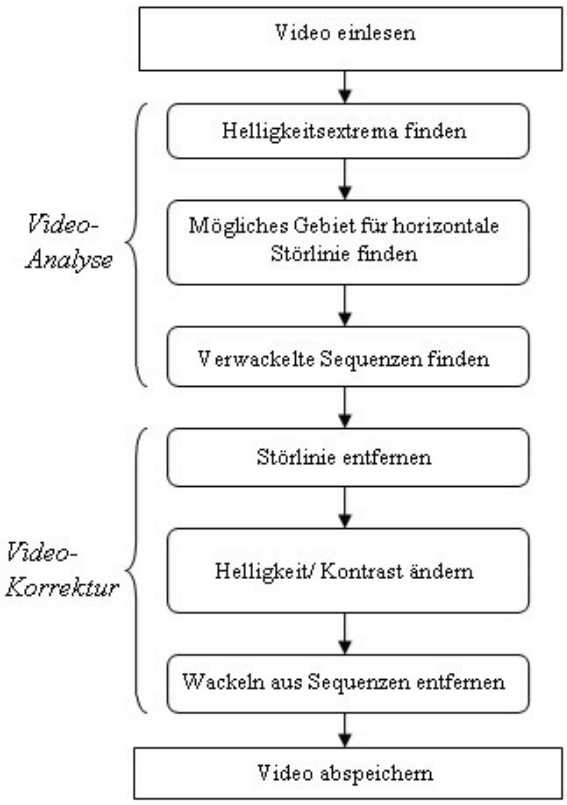

Abbildung 1: Gesamtüberblick über das Programm

2 Helligkeit und Kontrast

Im Folgenden wird nun die Erkennung von Helligkeitsfehlern in Videosequenzen näher betrachtet und es werden Wege zur Korrektur von Helligkeit und Kontrast aufgezeigt. Die Helligkeit eines Frames wird in diesem Bericht nur global betrachtet, d.h. bezüglich eines gesamten Frames. Dabei wird die globale Helligkeit eines Frames mit anderen Frames einer Sequenz verglichen. Lokale Fehler in der Helligkeit, beispielsweise eines einzelnen zu dunklen Objektes innerhalb eines Frames, werden nicht betrachtet. Abbildung 2 verdeutlicht die Anpassung eines zu hellen Bildes.

Neben der Helligkeit wird auch der Kontrast betrachtet. Als Kontrast bezeichnet man den Helligkeitsunterschied eines Bildes. Während das menschliche Auge den in der Natur auftretenden enormen Kontrastumfang relativ problemlos meistert, stößt man bei dem Bemühen, das Gesehene festzuhalten, an Grenzen. Deshalb kann nur etwa ein Zehntel des wirklichen Kontrastumfangs bewältigt werden, so dass zu helle Bereiche abgedunkelt oder zu dunkle Bereiche aufgehellt werden müssen.

Durch eine Verringerung des Kontrastes wird der Unterschied zwischen dem hellsten und dunkelsten Wert eines Frames reduziert, durch eine Vergrößerung entsprechend erhöht. So erfolgt die Korrektur dann auch durch Änderung der Helligkeit bzw. des Kontrastes für jeweils ein Frame.

2.1 Helligkeitsschwankungen

2.1.1 Erkennung von Helligkeitsschwankungen

Da Helligkeitsschwankungen auf der einen Seite nur lokal auftreten können, auf der anderen Seite die Lichtverhältnisse in unterschiedlichen Kameraeinstellungen differieren, ist es notwendig, das gegebene Video in einzelne Kameraeinstellungen zu unterteilen und diese dann separat zu untersuchen. Dies geschieht hier durch die Erkennung von harten Schnitten im Video. Unter einem Schnitt versteht man den Übergang von

Abbildung 2: Beispiel für die Korrektur von Helligkeitsschwankungen

einer Kameraeinstellung zur nächsten, wobei man in Abgrenzung von Blenden von einem *harten Schnitt* spricht. Dazu wird die Klasse `CutDetection` der *MoCA*-Bibliothek verwendet. Die im Folgenden beschriebenen Abläufe finden für jede Kameraeinstellung separat statt.

In Anlehnung an den Erkennungsalgorithmus von Yan und Kankanhalli [8] wird von jedem Frame die durchschnittliche Helligkeit `actLum` berechnet. Die maximale und minimale auftretende durchschnittliche Helligkeit `lumMax` bzw. `lumMin` einer Kameraeinstellung werden zwischengespeichert. Nun wird die Distanz $d = lumMax - lumMin$ berechnet. Die Helligkeit der Kameraeinstellung wird, wenn d größer ist als ein Schwellenwert `DTHRESIMBALANCE=20`, korrigiert. Anderenfalls gilt die Kameraeinstellung als ausbalanciert und es wird keine Änderung durchgeführt.

2.1.2 Korrektur von Helligkeitsschwankungen

Wurde eine Helligkeitsschwankung festgestellt, dann werden die Änderungswerte für Helligkeit und Kontrast, `lumChangeRate` und `contrastChangeRate` berechnet. Abbildung 3 zeigt den Ablauf der Erkennung und Korrektur von Helligkeitsschwankungen schematisch auf.

Yan und Kankanhalli [8] verwenden eine gleichbleibende Änderungsrate der Helligkeit für alle Frames. Dabei ändern sie nur die Helligkeit und lassen die Änderungsrate konstant. Die in diesem Bericht vorgestellten Algorithmen ändern jedoch die Helligkeit und den Kontrast, und zwar für jedes Frame der Kameraeinstellung einzeln in Abhängigkeit des bisherigen durchschnittlichen Helligkeitswertes `actLum` von jedem Frame. Dabei werden noch zwei Schwellenwerte `UPPERBOUND=110` und `LOWERBOUND=90`

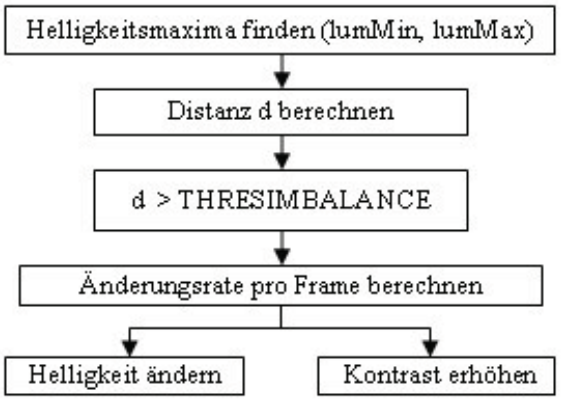

Abbildung 3: Ablauf der Helligkeitskorrektur

benötigt. Frames mit Helligkeitswerten zwischen diesen Schwellenwerten werden nicht geändert. Wenn $actLum < LOWERBOUND$ ist, handelt es sich um ein zu dunkles Frame, es gilt:

$$lumChangeRate = 0.8 \cdot (LOWERBOUND - actLum)$$

Dabei werden die Änderungswerte umso größer, je kleiner die aktuelle Helligkeit `actLum` ist. Theoretisch – bedingt durch die *MoCA*-Klasse `LuminanceFilter` – kann die `lumChangeRate` zwischen -255 und +255 liegen, der maximal mögliche Wert liegt jedoch hier bei +90, da durchschnittlich überhellte Helligkeitswerte wenig Kontrast bieten, verwaschen aussehen und das menschliche Auge nur wenig erkennen kann. Deshalb verkleinert der Faktor 0.8 die Änderungswerte und die übrige notwendige Aufhellung findet mit dem Kontrast statt. Hier gilt:

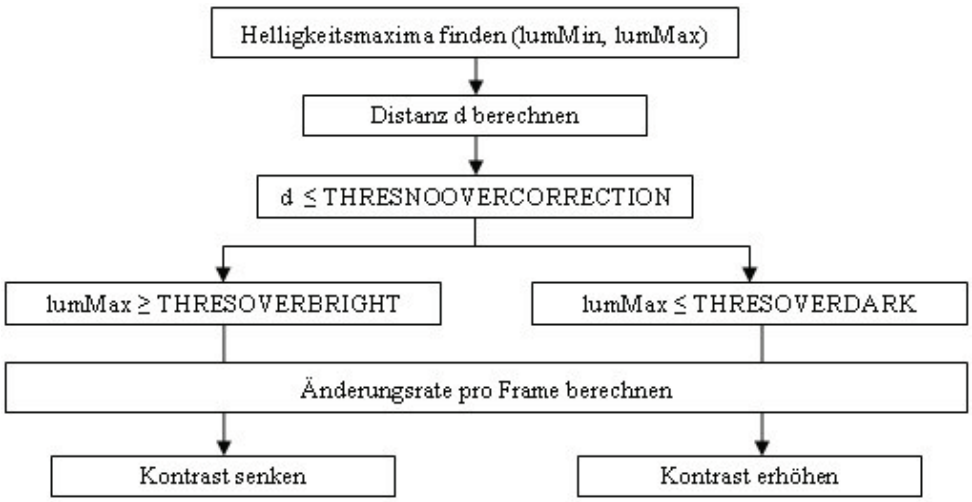

Abbildung 4: Ablauf der Erkennung und Korrektur zu heller oder zu dunkler Sequenzen

$$contrastChangeRate = -(0.5 * lumChangeRate)$$

Dieser Wert wird mit der *MoCA*-Klasse `ContrastFilter` geändert, sein Wertebereich liegt wiederum zwischen -255 und +255, den Maximalwerten des Kontrastes eines Bildes, wobei der negative Wert zu einer Erhöhung des Kontrastes führt. Da die Kontraständerung hier nur zur Ergänzung der Helligkeitsänderung dient, ist der maximal mögliche Wert entsprechend kleiner. Eine weitere Anpassung des Kontrastes wird in den Abschnitten 2.2 und 2.3 beschrieben.

Wenn $actLum > UPPERBOUND$ ist, handelt es sich um ein zu helles Frame, es gilt:

$$lumChangeRate = UPPERBOUND - actLum$$

Hier ergibt sich ein negativer Wert von maximal -145, falls das Bild komplett weiß wäre (`actLum` = 255). Eine Abdunklung um solch einen Betrag ist im Gegensatz zum vorhergehenden Fall durchaus vertretbar. Allerdings wird auch hier die u.U. starke Änderung der durchschnittlichen Helligkeit durch Erhöhung des Kontrastes ergänzt. Frames zwischen den Schwellenwerten gelten als ausbalanciert und werden nicht geändert.

2.2 Überdunkelte Sequenzen

2.2.1 Erkennung zu dunkler Sequenzen

Zu dunkle Sequenzen in Filmen können durch Unterbelichtung, falsche Entwicklung oder Beschädigungen bei der Lagerung der Filmrolle entstehen. Wie schon bei der Erkennung der Helligkeitsschwankungen werden die durch harte Schnitte ermittelten Kameraeinstellungen wieder separat betrachtet. Abbildung 4 gibt einen schematischen Überblick des vorgestellten Erkennungs- und Korrekturprozesses sowohl für zu dunkle als auch für zu helle Sequenzen.

Bei überdunkelten Sequenzen geht man davon aus, dass die größte in der Sequenz vorkommende Helligkeit `lumMax` unter einem Schwellenwert `THRESOVERDARK=70` liegt. Da es allerdings auch Sequenzen geben kann, bei denen sehr dunkle Frames keinen Fehler darstellen, muss noch ein weiteres Kriterium eingeführt werden. Dazu schaut man sich die bereits in 2.1.1 eingeführte Distanz $d = lumMax - lumMin$ über die Sequenz an. Diese muss zusätzlich unter einem Schwellenwert `THRESNOOVERCORECTION=40` liegen. Für eine überdunkelte Sequenz gilt also:

$$(lumMax \leq THRESOVERDARK) \quad and \quad (d \leq THRESNOOVERCORECTION)$$

2.2.2 Änderung zu dunkler Sequenzen

Wird eine Sequenz als überdunkelt erkannt, ist eine Verbesserung notwendig. Eine Erhöhung der Helligkeit jedoch wird wie bereits in 2.1.2 angesprochen vom Betrachter oft als unangenehm empfunden, deshalb findet diese hier nicht statt, es sei denn die Kriterien in 2.1.1 treffen zu und dort wird eine Helligkeitsänderung durchgeführt.

Um zu dunkle Sequenzen für den Betrachter deutlicher zu machen, wird hier eine Kontrasterhöhung durchgeführt, die wiederum für jedes einzelne Frame der Sequenz separat berechnet wird. Diese muss jedoch maßvoll stattfinden, da eine zu starke Erhö-

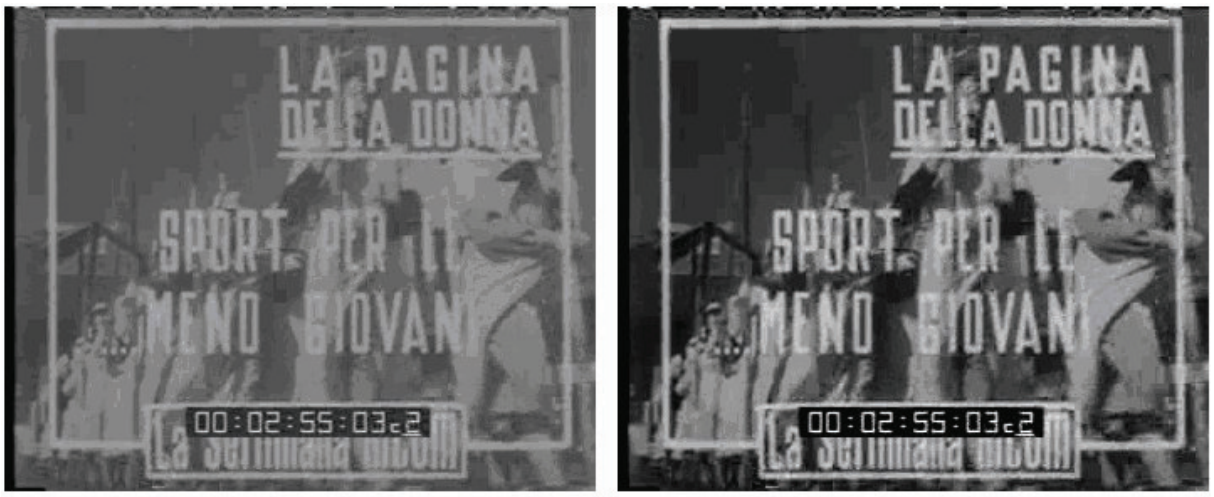

Abbildung 5: Beispiel für Helligkeitskorrektur

hung des Kontrastes zu Unschärfen führt. Die mögliche Kontrasterhöhung liegt deshalb zwischen -1 und -41. Es hat sich gezeigt, dass diese Werte nur zu kaum wahrnehmbaren Unschärfen führen. Für die Kontraständerung gilt demnach:

$$contrastChangeRate = d - 41$$

2.3 Überhellte Sequenzen

2.3.1 Erkennung zu heller Sequenzen

Zu helle Sequenzen können viele Ursachen haben. Sie können durch Überbelichtung, falsche Entwicklung oder Beschädigungen bei der Lagerung der Filmrolle entstehen. Auch hier wird die Erkennung wieder für jede Kameraeinstellung separat durchgeführt.

Die Erkennung der zu hellen Sequenzen erfolgt analog zur Erkennung der zu dunklen Sequenzen. Die größte in der Sequenz vorkommende Helligkeit `lumMax` muss wieder einen Schwellenwert `THRESOVERBRIGHT=110` übersteigen und die Distanz d unter dem Schwellenwert `THRESNOOVERCORECTION=40` liegen. Es gilt:

$$(lumMax \geq THRESOVERBRIGHT) \quad and \\ (d \leq THRESNOOVERCORECTION)$$

2.3.2 Änderung zu heller Sequenzen

Wie schon bei der Änderung der zu dunklen Sequenzen wird hier mit einer Kontraständerung statt einer Helligkeitsänderung gearbeitet. Allerdings wird hier der Kontrast verringert. Da bei zu starker Verringerung das Bild zu verschwommen aussieht, wird auch hier die Änderung begrenzt, sie liegt zwischen 1 und 201. Man sieht, dass die absoluten Änderungswerte im Vergleich zu 2.2.2 durchaus höher ausfallen können. Für die Änderungsrate gilt:

$$contrastChangeRate = 201 - 5 * d$$

Abbildung 5 zeigt den Effekt der Korrektur eines zu hellen Beispielframes. Es ist deutlich erkennbar, dass das Bild an Kontrast und Schärfe gewonnen hat und Einzelheiten besser zu erkennen sind.

3 Horizontale weiße Störlinie

3.1 Erkennung der weißen Linie

Störlinien haben charakteristische Merkmale, mit deren Hilfe ein Erkennungsalgorithmus entwickelt werden kann. Die Pixel einer Linie haben oft eine wesentlich höhere bzw. niedrigere Helligkeit als die Nachbarpixel. Es handelt sich um sehr schmale, gerade Linien ohne Kurven [3]. Horizontale Störlinien beginnen am linken äußeren Rand eines Frames und reichen meist bis über die Hälfte einer Zeile. Es kann eine vertikale Verschiebung einer Linie im Zeitablauf geben, die jedoch 5% vertikale Abweichung nicht überschreitet [7]. Beispiele von Störlinien sind in Abbildung 6 dargestellt.

Wie bereits in Abschnitt 1 erwähnt, entstehen Störlinien durch Kratzer, die beim Transport der Filmrolle im Abspielgerät entstanden sind, oder schon bei der Entwicklung des Filmes. Aufgrund deren Zustandekommens treten sie häufig über einen längeren Zeitraum auf und sind unabhängig von harten Schnitten

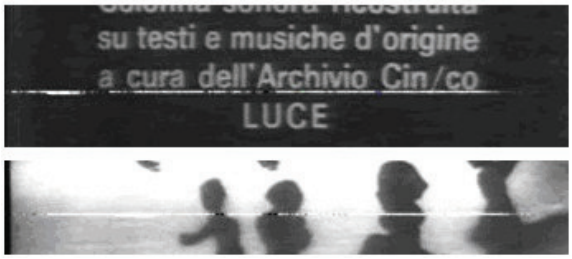

Abbildung 6: Beispiele für weiße Linien

oder anderen inhaltlichen Schnitten, so dass der Erkennungsprozess – im Gegensatz zu dem Verfahren in Abschnitt 2 – über die gesamte zu untersuchende Sequenz stattfindet.

Problematisch bei der Erkennung einer weißen Linie sind ähnliche Objekte, die im Video vorkommen. Dabei kann es sich um Text oder um dünne helle Kanten wie in Abbildung 7 handeln. Insbesondere deshalb, weil die Störlinien oft die charakteristischen sehr hellen oder sehr dunklen Pixel nur sporadisch aufweisen, und viele graue Pixel haben, die nur sehr schwer von anderen Pixel unterschieden werden können. Aus diesem Grund werden die vorgestellten Parameter, insbesondere die Helligkeitsschwellen, sehr niedrig gewählt.

3.1.1 Voruntersuchungen

Um eine effektive Erkennung durchführen zu können, und um Bereiche, die zwar Ähnlichkeiten zu den charakteristischen Merkmalen einer weißen Störlinie aufweisen, jedoch keine enthalten, identifizieren zu können, sind einige Voruntersuchungen notwendig.

Zu Beginn wird nun die Methode `checkPixel` vorgestellt, die einen wichtigen Bestandteil des Erkennungsprozesses darstellt. Bei dieser Methode wird der Wert eines jeden übergebenen Pixels überprüft, und zurückgegeben, ob es charakteristische Merkmale eines Linienpixels besitzt. Falls dieser Pixelwert, ein linker oder rechter Nachbar oder sowohl der Nachbar zwei Positionen links als auch der Nachbar zwei Positionen rechts über einem Schwellenwert `WHITETHRES=230` liegt, ist die erste Bedingung erfüllt (`white=true`). Abbildung 8 veranschaulicht dieses Vorgehen und zeigt die drei Fälle, bei denen die erste Bedingung erfüllt ist. Dabei stellt der dicke schwarze Pfeil das aktuell zu überprüfende Pixel dar, der dünne weiße Pfeil zeigt die hellen Pixel, die über dem Schwellenwert liegen.

Diese Vorgehensweise ist notwendig, da zwar viele Pixel aus einer Linie einen hohen Wert haben, jedoch die unmittelbaren Nachbarn nicht, obwohl auch diese auf der Linie liegen und verändert werden sollen. Anschließend wird überprüft, ob das Pixel Teil einer vertikalen Linie oder einer weißen Fläche ist. Wenn dies der Fall ist, gehört das Pixel nicht zu einer weißen horizontalen Linie. Dazu wird der aktuelle Wert des Pixels mit dem zwei Pixel darüber sowie darunter liegenden Nachbarn verglichen. Unterscheiden sich diese um mehr als `VERTICALDIFFERENCETHRES=20`, so ist dies nicht der Fall (`notvertical=true`). Nur wenn beide Prüfungen erfolgreich sind, gibt die Methode `true` für das geprüfte Pixel zurück.

Ziel des ersten Arbeitsschrittes ist das Auffinden des Bereiches, in dem möglicherweise eine horizontale weiße Linie auftritt. Der Bereich soll ca. 20 Zeilen umfassen, da wie in Abschnitt 1 bereits beschrieben die vertikale Abweichung einer Linie nur gering ist. Die Methode `presearchHorizontalLine` durchläuft das komplette Video. Für jedes einzelne Frame werden alle Pixel durchgegangen und mit der oben vorgestellten Methode `checkPixel` überprüft, ob ein Pixel charakteristische Eigenschaften einer weißen Linie erfüllt. Ist dies der Fall, so wird in dem globalen Feld `yCounter` an der Position der Zeilennummer des geprüften Pixels der Wert um eins erhöht.

Am Schluss wird die Zeile, an dem die höchste Anzahl möglicher Linienpixel vorkommt, also die Stelle von `yCounter` mit dem Maximalwert, als mittlere Zeile für den Bereich gewählt, in dem die Linie auftritt.

3.1.2 Erkennungsprozess

Nun wird das komplette Video nochmals durchlaufen, diesmal werden aber nur diejenigen Pixel untersucht, die sich in dem Bereich befinden, der in 3.1.1 bestimmt wurde. In diesem Bereich liegt – falls in dem Film vorhanden – eine weiße Linie. Nun muss für jedes Frame die genaue Position der Linie gefunden werden, sowie deren horizontale Anfangs- und Endposition.

Die Frames werden nun Linie für Linie durchgegangen. Jedes Pixel wird nun wieder von der Methode `checkPixel` überprüft, und falls `true` zurückgegeben wird, wird ein Linienzähler `lineCounter` um eins erhöht. Wenn es sich um das erste oder letzte mögliche weiße Pixel handelt (`firstOccurrence` bzw. `lastOccurrence`) wird diese Position gespeichert. Da auch sehr dunkle Pixel in einer Linie vorkommen können, sucht die Methode `check4BlackPixel`, die vergleichbar ist mit `checkPixel`, nach diesen, und falls ein schwarzes Pixel gefunden wird, wird `lineCounter` ebenfalls um eins erhöht.

Damit es sich nun um weiße Störlinie handelt, müssen zwei Bedingungen erfüllt sein. `lineCounter` muss größer sein als `LINECOUNTERTHRES=30`, und die Differenz zwischen erster und letzter Position eines weißen Pixels der Linie muss größer sein als `DIFFFIRSTLASTTHRES=50`.

Abbildung 7: Beispiel für natürlich auftretende weiße horizontale Linien

3.2 Beseitigung der weißen Störlinie

Wurde eine Störlinie entdeckt, soll diese beseitigt werden. Da es sich wie schon beschrieben um ein lokales Phänomen handelt, sind genügend Bildinformationen vorhanden, um die fehlerhaften Pixel zu rekonstruieren. Die Höhe einer Störlinie beträgt ein Pixel. Die direkt darüber bzw. darunter liegenden Pixel können dadurch noch leicht verändert sein. Deshalb werden als Referenzwerte die jeweils 2 Pixel darüber bzw. darunter liegenden Pixel verwendet. Der neue Wert eines Pixels auf der Störlinie wird dann mit Hilfe dieser Referenzpixel interpoliert.

Für die Korrektur wird nun die fehlerhafte Zeile eines Frames in drei Bereiche unterteilt, an denen Korrekturen vorgenommen werden. Der erste Bereich startet an Position `firstOccurrence-80` (bzw. Position 0 falls `firstOccurrence < 80`) und geht bis `firstOccurrence`. Dabei handelt es sich um den Anfangsbereich der Zeile. In den meisten Fällen beginnt dieser Bereich bei der Position 0, da wie bereits erwähnt die Störlinien oft ganz links beginnen. Der zweite Bereich geht von `firstOccurrence` bis `lastOccurrence`, er ist also der zentrale Bereich der Störlinie.

Der dritte Bereich geht von `lastOccurrence` bis zu `lastOccurrence+20`. Dabei handelt es sich um den Bereich unmittelbar nach dem vermeintlichen Ende einer Störlinie, in dem allerdings auch noch fehlerhafte Pixel vorkommen. Außerdem wird das Ende einer Störlinie durch den Erkennungsalgorithmus nicht immer fehlerfrei erkannt, so dass dieser *Pufferbereich* zu guten Ergebnissen führt. Der Restbereich der Linie – sofern vorhanden – hat keine Störlinie, deshalb sind auch keine Änderungen notwendig.

Im mittleren Bereich, also in dem Bereich, in dem alle Pixel als fehlerhaft gelten, findet eine lineare Interpolation statt. Für die allgemeine Newton'sche Interpolationsformel $f(x) = f_0 + \frac{f_1 - f_0}{x_1 - x_0}(x - x_0)$ wird dabei für x ein geeigneter Faktor gewählt. Für alle Pixel in diesem Bereich gilt:

$$newValue = 0.5 * (a - b) + b$$

wobei a der Pixelwert zwei Positionen über und b der Pixelwert zwei Positionen unter der aktuellen Pixelposition sind. Wie erwähnt können die direkt darüber liegenden Pixel ebenfalls Fehler enthalten. Deshalb gilt für das darüber liegende Pixel:

$$newVal1Obove = \lfloor 0.3 * (oldVal1Obove - newVal) + newVal \rfloor$$

Man sieht, dass hier sowohl der bisherige Wert des Pixels als auch die Werte der darüber bzw. darunter liegenden Pixel in den neuen Wert eingehen. Dabei wird der alte Wert zu 30% gewichtet. Für die direkt darunter liegenden Pixel `newValue1Below` gilt entsprechendes.

Im ersten Bereich werden die Pixelwerte nun in Abhängigkeit der Entfernung zum erkannten Linienbeginn geändert. Es gilt für den neuen Pixelwert:

$$newValue = [factor * (0.5 * (a + b) - oldValue] + oldValue$$

Dabei ist `factor` der dynamische Gewichtungsfaktor, der wie folgt berechnet wird:

$$factor = 1 - ((firstOccurrence - x)/80)$$

x stellt hier die x-Position des Pixels auf der Linie dar. Je weiter das Pixel also vom erkannten Linienbeginn entfernt ist, desto stärker wird der bisherige Pixelwert gewertet. Die darüber bzw. darunter liegenden Pixel werden wie im mittleren Bereich geändert. Der dritte Bereich umfasst die 20 Pixel nach dem Ende der Linie. Wie auch im ersten Bereich soll die Berechnung in Abhängigkeit vom Abstand dazu erfolgen. Die Formel ist entsprechend, allerdings wird der Gewichtungsfaktor wie folgt berechnet, um der kürzeren Länge des Bereiches Rechnung zu tragen.

$$factor = (1 - (x - lastOccurrence)/20)$$

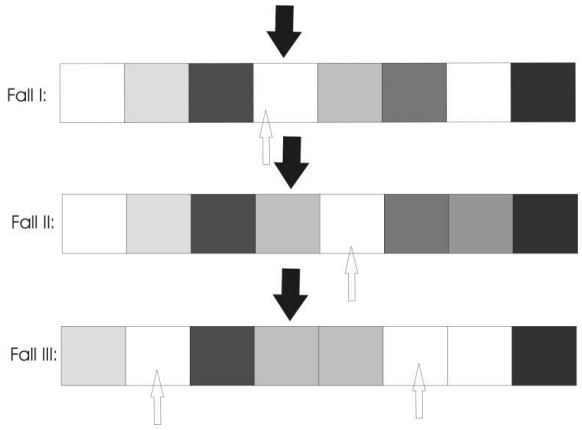

Abbildung 8: Veranschaulichung der Methode checkPixel

Wiederum werden die darüber bzw. darunter liegenden Pixel wie im mittleren Bereich geändert. Die beschriebene Korrektur wird für jedes Frame, das vom Erkennungsalgorithmus aus 3.1.2 ausgewählt wurde, einzeln durchgeführt. Abbildung 9 zeigt das Ergebnis an zwei Beispielen. Die Linien wurden vollständig erkannt und beseitigt.

Tests haben ergeben, dass der Algorithmus bei der Beseitigung der Linien einen hohen Erfolgsfaktor aufweist. Allerdings kann es vorkommen, dass in einzelnen Frames die Linie nicht erkannt wurde, da sie stark unterbrochen ist bzw. zu graue Pixel enthält. Selten kann es auch vorkommen, dass der Erkennungsalgorithmus fälschlicherweise eine Linie entdeckt und eine Korrektur durchgeführt wird. Dies ist jedoch für den Benutzer meist nicht erkennbar, da die Länge dieser vermeintlichen Linie relativ kurz ist und der Interpolationsalgorithmus die Pixelwerte nur minimal verändert.

4 Verwackelte Sequenzen

Wenn sich die meisten Objekte – in einer größeren Anzahl von zusammenhängenden Frames eines Videos – wiederholt in einem kurzen Zeitraum in dieselbe Richtung hinwärts und rückwärts bewegen, dann spricht man von einer verwackelten Sequenz [8]. Verwackelte Sequenzen können unterschiedliche Ursachen haben. Auf der einen Seite können sie schon bei der Aufnahme entstanden sein, dies ist v.a. bei Aufnahmen von Hobbyfilmern der Fall. Sie entstehen durch eine unruhige Kameraführung oder durch einen unbeabsichtigten Schwenk. Auf der anderen Seite können sie durch ungleichmäßigen Filmtransport bei der Aufnahme oder dem Digitalisierungsprozess entstehen.

Czúni et al. haben einen Algorithmus zur automatischen Stabilisierung von Filmen entwickelt. Dabei wird ein Referenzframe nicht durch einen Benutzer, sondern automatisch gesucht. Von diesem Referenzframe aus werden mittels inverser Fourier Transformation Spitzenwerte gesucht und mit Schwellenwerten verglichen. Darauf basierend wird anschließend eine Korrektur durchgeführt. Diese Vorgehensweise liefert gute Ergebnisse bei langen Sequenzen, z.B. wenn der komplette Film verwackelt ist.

Das in diesem Bericht entwickelte Verfahren eignet sich gut für die Entfernung von Verwacklungen in kurzen Kameraeinstellungen. Dies gilt auch bei Verwacklungen mit größeren Verschiebungswerten. Dies ist besonders dann interessant, wenn nur eine Kameraeinstellung eines Filmes verwackelt ist und der Rest des Filmes dieses Problem nicht hat. Wie schon in den vorherigen Abschnitten wird der Prozess zweigeteilt. Zuerst findet eine Erkennung statt und anschließend bei Bedarf eine Korrektur.

4.1 Erkennung von verwackelten Sequenzen

Der Erkennungsprozess wird wiederum pro Kameraeinstellung durchgeführt, also immer zwischen zwei harten Schnitten. Der Erkennungsalgorithmus basiert auf der *MoCA*-Klasse `CameraModelData`, die die Bewegungsvektoren zwischen jeweils zwei aufeinander folgenden Frames berechnet und zur Verfügung stellt. Eine exakte Berechnung dieser Vektoren ist für die korrekte Erkennung und ganz speziell für die korrekte Behebung sehr wichtig. Im Verlauf der Entwicklung dieses Algorithmus wurde festgestellt, dass die Klasse `CameraModelData` bisher ungenaue Vektoren liefert. Deshalb musste das zugrunde liegende Berechnungsverfahren in der Klasse in einen *full search* Suchalgorithmus abgeändert werden.

Zu Beginn werden wieder Voruntersuchungen durchgeführt, und zwar jeweils eine Untersuchung für Bewegungen in x-Richtung, also horizontale Bewegungen und für Bewegungen in y-Richtung, also vertikale Bewegungen. Dazu werden die Bewegungen zwischen allen Frames der Kameraeinstellung addiert. Um jedoch fehlerhafte Bewegungsvektoren sowie sehr große Bewegungen herauszufiltern, die die Berechnungen stark verfälschen würden, werden grundsätzlich alle Werte größer `THRESBIGMOVEMENT=32` für die Berechnungen nicht verwendet.

Wenn eine Kameraeinstellung verwackelt ist, finden abwechselnde Bewegungen nach links und rechts bzw. oben und unten statt. Die Summe der Bewegungsvektoren würde in diesem Fall einen kleinen Wert geben, da sich positive und negative Werte gegenseitig aufheben. Um einen Schwellenwert für diese Summe festzulegen, muss jedoch der Wert angepasst werden, um auch die Länge einer Kameraeinstellung zu berücksichtigen. Dazu wird die Summe durch die Zahl

Abbildung 9: Beispiele zur Entfernung einer Störlinie

der Frames der Kameraeinstellung geteilt und der Betrag davon genommen. Ist dieser Wert nun kleiner `THRESNORMALMOVING=1,1`, so liegt keine größere Bewegung in eine Richtung, wie z.B. ein Kameraschwenk, in der Kameraeinstellung vor.

Dies bedeutet jedoch noch nicht, dass es sich um eine verwackelte Kameraeinstellung handeln muss. Es könnte sich auch um eine Kameraeinstellung handeln, in der keine Kamerabewegung vorhanden ist. Dazu werden nochmals die Bewegungen zwischen jeweils 2 Frames addiert, allerdings nur für jeweils 8 Frames und im Gegensatz zur vorigen Summe betragsweise. Zu große Werte werden wieder ignoriert und die Summe entsprechend kompensiert. Nach den 8 Frames wird die berechnete Summe mit dem Schwellenwert `THRESNOMOVEMENT=0,65` verglichen. Ist die Summe kleiner als dieser Wert, gibt es in den untersuchten Frames keine nennenswerte Bewegung. Hier geht man nun davon aus, dass in einer verwackelten Kameraeinstellung in 8 zusammenhängenden Frames eine Bewegung vorhanden sein müsste. In diesem Fall wird die Untersuchung abgebrochen und für die beobachtete Richtung zurückgegeben, dass dort kein Wackeln vorgefunden wurde. Anderenfalls wird die Untersuchung bis zum Ende der Kameraeinstellung fortgesetzt. Ist nun weder eine gewünschte Bewegung noch überhaupt keine Bewegung in der Kameraeinstellung vorhanden, wird die Kameraeinstellung als verwackelt erkannt.

Für eine spätere Korrektur ist es wichtig, ein Referenzframe zu bestimmen, anhand dessen eine Ausrichtung der Frames stattfinden kann, die korrigiert werden sollen. Dabei könnte man im Idealfall das Frame in der Mitte der tatsächlich verwackelten Frames wählen, so dass möglichst kleine Korrekturfaktoren benötigt werden. Da das Referenzframe jedoch automatisch und nicht von einem Benutzer gewählt werden soll, ist dies sehr schwierig zu finden.

Bei der Entwicklung des Erkennungsalgorithmus wurden Versuche getätigt, Frames mit wenig Bewegung als Referenzframes für den Korrekturprozess zu wählen. Dies führte jedoch in einigen Fällen zu schlechten Ergebnissen. Eine weitere Möglichkeit wäre es, das Frame zu Beginn einer Kameraeinstellung als Referenz zu wählen. Da es sich bei Wechseln von Kameraeinstellungen aber auch um Überblendungen handeln kann und die ersten Frames noch starke Abweichungen der Bewegung enthalten können, bietet sich auch diese Option nicht an.

Die Methode, die schließlich gewählt wurde, bestimmt das Frame in der mittleren Position der Szene als Referenzframe. Dies funktioniert dann besonders gut, wenn man davon ausgeht, dass die komplette Kameraeinstellung durchgehend verwackelt ist, und die Länge einer Kameraeinstellung überschaubar ist. Der Voruntersuchungsalgorithmus berechnet schließlich diese Referenzposition für jede als verwackelt erkannte Kameraeinstellung und liefert diese an die Korrekturmethode weiter.

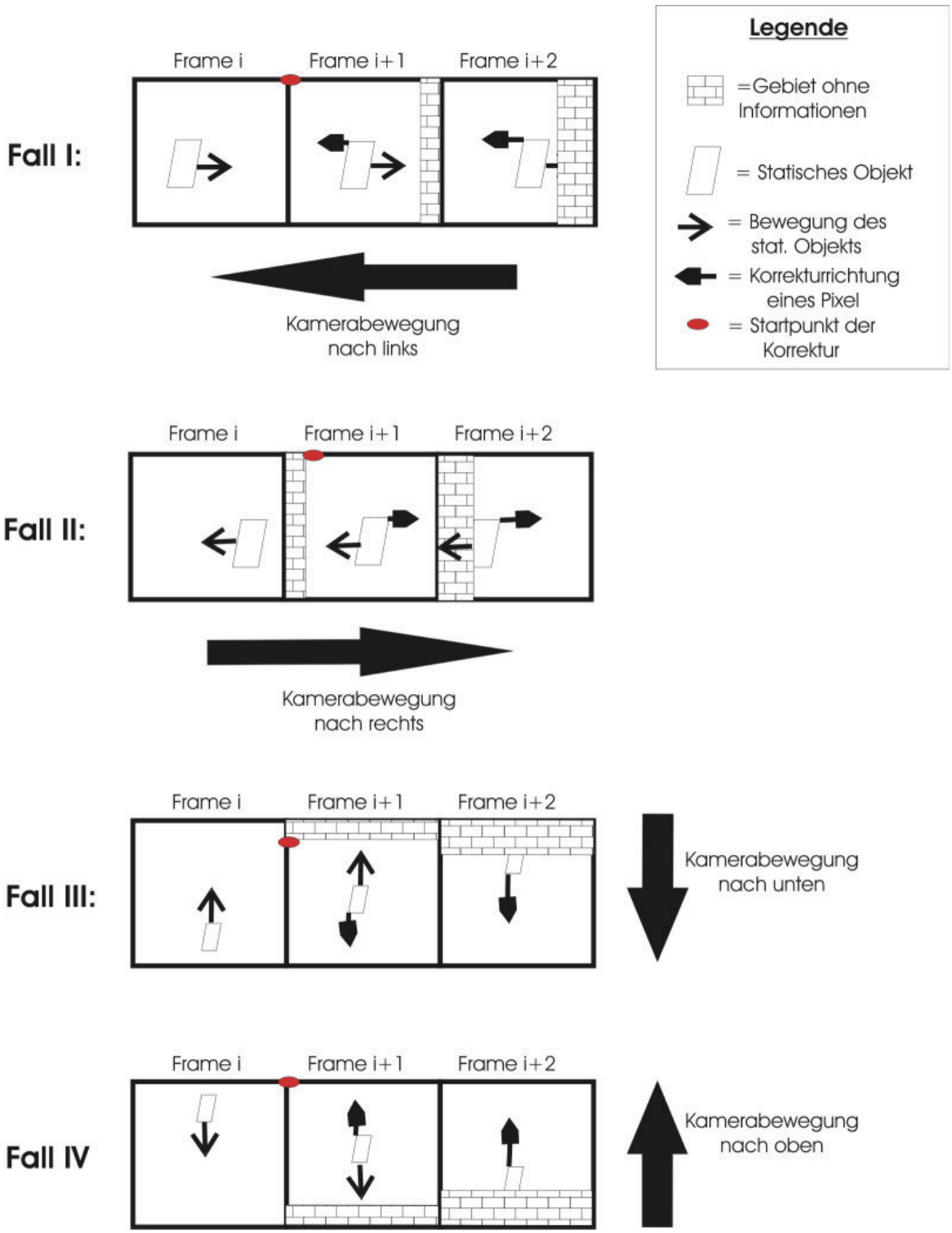

Abbildung 10: Die 4 Fälle der Kamerabewegungen und deren Korrekturen

Abbildung 11: Abfolge von Frames mit sich änderndem Rand

4.2 Stabilisierung von verwackelten Sequenzen

Jede verwackelte Kameraeinstellung soll nun entsprechend korrigiert werden, um das Wackeln zu entfernen. Dazu muss für jedes einzelne Frame sowohl in x- als auch in y-Richtung ein Verschiebungsvektor berechnet werden. Wie schon in 4.1 erwähnt, soll dieser Verschiebungsvektor relativ zum Referenzframe berechnet werden.

Für alle Frames, die vor der Referenzposition liegen, werden die Korrekturfaktoren berechnet, indem alle Bewegungsvektoren, von dem Referenzframe bis zum aktuell zu berechnenden Frame, aufaddiert werden und das Ergebnis ggf. negiert wird. Dabei gelten dieselben Einschränkungen wie in 4.1. Es entsteht ein Verschiebungswert relativ zum Referenzframe. Entsprechendes findet für alle Frames nach der Referenzposition statt.

Nach diesem Arbeitsschritt werden Korrekturfaktoren für die x- und y-Richtung der Kameraeinstellung berechnet. Diese werden abgerundet, da das Verschieben von Pixeln nur in ganzzahligen Werten möglich ist. Der Korrektur sind auch andere Grenzen gesetzt. So machen Verschiebungswerte von mehr als `THRESCORRECTIONIMPOSSIBLE=90` keinen Sinn, worauf im Folgenden noch näher eingegangen wird. Deshalb werden alle größeren Werte auf diesen Wert verkleinert. Werte dieser Größenordnung stellen allerdings auch kein Verwackeln dar. Sie könnten beispielsweise durch unkontrollierte Kameraschwenks entstehen. Zur Korrektur dieser Art von Fehler sind jedoch andere Algorithmen notwendig.

Bei der Verschiebung der Pixel tritt nun das Problem auf, dass ein Bildbereich entsteht, für den keine Bildinformationen vorhanden sind. Abbildung 10 zeigt die vier unterschiedlichen Fälle eines Verschiebungsprozesses, auf die auch im Folgenden noch näher eingegangen wird. Dabei werden die Kamerabewegungen sowie die Objektbewegungen dargestellt und die Richtung der Pixelkorrektur gezeigt.

Für den Bereich *mit Ziegeln* sind keine Informationen vorhanden. In der Abbildung würden dort Informationen von außerhalb des Frames geholt. Versuche, diese Informationen aus vorherigen bzw. nachfolgenden Frames zu gewinnen sind fehlgeschlagen, da eine natürliche bzw. gewünschte Bewegung von Objekten oder Personen in einem Film stattfinden kann, d.h. die Bildinformationen in den Bereichen ändern sich ebenfalls. Um diese auszugleichen, müsste eine Objekterkennung stattfinden und die Bewegungsvektoren dieser Objekte entsprechend zurückverfolgt werden.

In diesem Bericht wird der Bereich ohne Informationen als schwarzer Rand dargestellt, der ein Frame wie einen Rahmen umgibt. Die Größe dieses Randes wird dynamisch berechnet. Diese Berechnung wird am Ende dieses Abschnitts näher erläutert.

Zunächst wird auf die Korrektur eingegangen. Dazu werden die vier Fälle aus Abbildung 10 einzeln besprochen. In Fall I wird ein Frame i+1 neu berechnet. Dazu werden, am linken oberen Ende beginnend, die Pixelwerte des ursprünglichen Frames i+1 in x-Richtung um den berechneten Korrekturfaktor nach links verschoben. Das neue Frame i+1 wird um den Korrekturfaktor verschoben. Die Pixelwerte des *geziegelten* Bereiches, der die Breite des Korrekturfaktors hat, bleiben beim neuen Frame vorerst ohne Informationen. Dieser Bereich wird am Ende von dem schwarzen Rand überdeckt.

Entsprechend wird beim Fall II nicht links oben, sondern erst nach dem Randbereich begonnen. In Abbildung 10 ist dieser Anfangpunkt als ovaler Punkt markiert. Die Pixelwerte werden hier um den Korrekturfaktor in x-Richtung nach links verschoben. Bei Fall III werden beim ovalen Punkt beginnend die Pixelwerte in y-Richtung um den Korrekturfaktor nach unten verschoben. Der oben entstehende Bereich ohne Informationen wird wiederum später durch einen schwarzen Rand überdeckt.

In Fall IV schließlich wird wieder oben links begonnen die Pixelwerte des ursprünglichen Frames i+1 für das neue Frame i+1 um den Korrekturfaktor nach oben zu verschieben. Der Bereich ohne Informationen entsteht am Frameende unten.

Das neu entstandene Frame i+1 ist nun in x- und y-Richtung verschoben worden. Das Objekt in Abbildung 10 befindet sich nun an derselben Position wie bei Frame i. Schließlich muss noch der schwarze Rand über das *neue* Frame gelegt werden. Versuche, den Rand dynamisch bei jedem einzelnen Frame zu berechnen führten zu sehr schlechten Ergebnissen. Wie in Abbildung 11 dargestellt, erscheint ein sich schnell ändernder schwarzer Rand für einen Betrachter sehr störend und macht den Effekt, das verwackelte Bild zu stabilisieren, zunichte.

Eine weitere Möglichkeit besteht darin, einen fixen Rand entweder über den gesamten Film oder jeweils

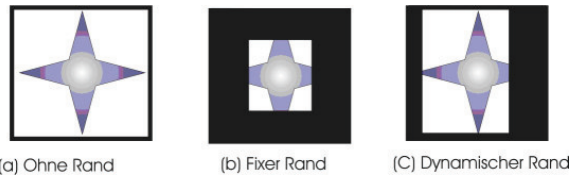

Abbildung 12: Frames mit unterschiedlichen Rändern

über eine gemeinsame Kameraeinstellung, wie in Abbildung 12 (b) dargestellt, zu legen. Die Größe dieses Randes könnte entweder fix sein oder den Wert der größten vorkommenden Korrektur betragen. Kommen in einem Video jedoch große Werte vor, wenn auch nur vereinzelt, würde ein großer Teil des Videos *weggeschnitten*. Es entsteht also ein großer Verlust, der in vielen Fällen nicht hinnehmbar ist. Hier wird ein dynamischer Rand gewählt, der für alle vier Seiten jeweils den maximal notwendigen Wert für eine Kameraeinstellung berechnet, und dann diesen Rand für eine Kameraeinstellung fix lässt. Dies ist in Abbildung 12 (c) demonstriert.

Sind keine oder nur sehr geringe Korrekturen notwendig ist kein oder ein sehr kleiner Rand in der Kameraeinstellung vorhanden. Findet ein Wackeln nur in x-Richtung statt, so ist oben und unten kein Rand vorhanden. Dies stellt einen annehmbaren Kompromiss zwischen den Alternativen dar.

Es bleibt anzumerken, dass der Algorithmus nicht für jeden Film gut geeignet ist. Wenn man weiß, dass in einem Film keine oder nur sehr sporadische bzw. ungleichmäßig verwackelte Stellen auftreten, sollte man den Algorithmus nicht anwenden. Denn neben einer schlechten Performance, die aus der Berechnung der Bewegungsvektoren resultiert, sind auch kleine schwarze Ränder störend, wenn durch sie kein Vorteil entsteht.

Wenn jedoch relativ konstante Verwacklungen durch Projektoren oder durch einen ungeübten Kameramann entstehen, sollte man den Algorithmus anwenden. Ebenso ist es vorteilhaft, wenn die einzelnen Kameraeinstellungen nicht zu lange sind, da auch kleine Fehler bei der Berechnung der Verschiebungen bei großem Abstand zum Referenzframe problematisch sind.

5 Ergebnis und Ausblick

5.1 Experimentelle Ergebnisse

Innerhalb dieses Berichtes wurde gezeigt, dass es auch mit vermeintlich einfachen Algorithmen möglich ist, effektiv gute Ergebnisse zu erzielen. Die getesteten Videos, hauptsächlich digitalisierte historische Schwarz-Weiß-Filme des *ECHO*-Projektes, konnten mit dem entwickelten Programm deutlich verbessert werden. Im Rahmen der Entwicklung wurden ca. 20 Sequenzen zwischen 30 und 120 Sekunden verwendet, die markante Fehler enthalten.

Helligkeitsänderungen wurden – in unterschiedlichem Umfang – bei allen Sequenzen durchgeführt. Die Helligkeit und der Kontrast machten die Bilder deutlicher erkennbar. Allerdings wäre es hier wünschenswert, auch die Helligkeit einzelner Objekte verändern zu können, anstatt jeweils nur ein gesamtes Frame.

Bei der Erkennung von horizontalen Störlinien waren die Erfahrungen unterschiedlich. Wiesen die Linien die beschriebenen markanten Eigenschaften auf, so wurden sie gut erkannt und konnten weitestgehend behoben werden. Die Qualität wurde dadurch erheblich verbessert. Wichen die Linien – v.a. in Einzelframes – stark davon ab, so konnten sie streckenweise nicht erkannt werden. Des Weiteren ist der Algorithmus bisher nur in der Lage, eine Störlinie bzw. Störlinien in einer Region zu finden. Für lange Filme bzw. Filme mit einer größeren Anzahl solcher Fehler müsste der Algorithmus entsprechend angepasst werden.

Wie schon in Abschnitt 4 erwähnt, sollte der Algorithmus zur Behebung von Verwacklungen nicht bei jedem Film angewandt werden. Ein Blick in die Archive zeigt, dass Filme aus professionellen Quellen, wie z.B. historische Nachrichtensendungen, selten verwackelt sind. Problematischer ist dies bei historischen Amateurvideos, die mit schlechten Kameras aufgezeichnet wurden sowie bei Verwacklungen durch den Digitalisierungsprozess. Gerade wenn nur in vereinzelten Kameraeinstellungen Verwacklungen auftreten, diese aber sehr störend für den Betrachter sind, ist eine deutliche Verbesserung zu beobachten.

Bei den getesteten Videos hat der Erkennungsalgorithmus zu guten Ergebnissen geführt. Zur Simulation wurden auch vier Sequenzen mit künstlichem Wackeln in unterschiedlichen Richtungen und in unterschiedlicher Intensität erstellt. Diese Verwacklungen wurden in allen Fällen korrekt erkannt. Die künstlich erstellten Sequenzen wurden fehlerfrei stabilisiert, bei den übrigen Filmen kam es zu deutlichen Verbesserungen.

Die Laufzeit des Algorithmus ist in starkem Maße von den eingebundenen *MoCA*-Klassen abhängig. Diese erstellen bei erstmaliger Verwendung Indexdateien, in denen Videodaten gespeichert werden. Speziell bei der Berechnung der Bewegungsvektoren benötigt dies sehr viel Zeit. Deshalb ist es sinnvoll, den Algorithmus zur Erkennung und Behebung von Verwacklungen nur bei Bedarf einzusetzen.

Auch die in diesem Bericht vorgestellten Algorithmen laufen bei durchschnittlicher Rechenleistung nicht in Echtzeit ab, sondern im Bereich 1:4. Eine Annäherung an Echtzeit ist jedoch bei diesem Aufgabengebiet nicht notwendig, da ohnehin der Digitalisierungsprozess einen hohen Zeitaufwand benötigt und

der Optimierungsprozess nur einmalig stattfindet.

5.2 Zukünftige Ansatzpunkte

Die Suchalgorithmen, die in diesem Bericht entwickelt wurden, basieren weitestgehend auf statischen Schwellenwerten, anhand derer Vergleiche durchgeführt werden. Diese Schwellenwerte wurden durch Beispielvideos manuell verfeinert. Es ist jedoch vorstellbar, solche Schwellenwerte durch *KI*-Verfahren wie beispielsweise *Künstliche Neuronale Netze* (*KNN*) berechnen zu lassen oder dynamisch zu gestalten. Bei *KNN* werden die Algorithmen mit Beispieleingaben trainiert und bestimmen die Schwellenwerte eigenständig.

Neben den betrachteten Fehlern gibt es noch weitere, die bei digitalisierten historischen Filmen auftreten können. Dunkle oder helle Flecken treten durch den Verfall des Originalmaterials in einzelnen Frames auf. Hierbei sind oft keine Informationen mehr über die Originalpixel vorhanden. Diese Fehler können durch lineare Interpolation oder durch Interpolation basierend auf Min-Max-Funktionen repariert werden, wie sie von Armstrong, Kokaram und Rayner [1] vorgeschlagen werden. Dabei werden die Ausgleichswerte für die Interpolation mit numerischen Verfahren genauer bestimmt, um verschwommene Kanten zu vermeiden.

Weit verbreitet sind auch vertikale Störlinien. Bretschneider, Kao und Bones [2] stellen dazu einen Algorithmus vor, der mit Hilfe von diskreter Wavelet-Transformation arbeitet. Es ist jedoch auch leicht möglich, den in diesem Bericht vorgestellten Algorithmus zu Erkennung von horizontalen Störlinien entsprechend zu modifizieren. Verbreitet sind auch Fehler, die durch Schmutz und Dreck bei unsachgemäßer Lagerung, durch Verschmutzungen während des Digitalisierungsprozesses oder beispielsweise durch ein Haar auf der Filmrolle entstehen. Diese Fehler treten oft nur lokal auf, meist nur bei einem einzelnen Frame. Deshalb nennt man sie *1-Frame-Fehler*.

Schallauer et al. [7] haben ein Erkennungsverfahren entwickelt, bei dem diese Fehler durch Analyse der Helligkeit von Bewegungstrajektorien über mehrere Frames automatisch entdeckt werden können. Eine Korrektur findet dann mittels Interpolation oder den benachbarten Frames statt. Das beschriebene Verfahren eignet sich sowohl bei kleinen Fehlern in einem einzelnen Frame als auch bei großflächigen Beschädigungen und würde eine gute Ergänzung der entwickelten Methoden darstellen.

5.3 Ausblick

Abschließend lässt sich anmerken, dass dieses interessante und wichtige Forschungsgebiet sicherlich auch in den nächsten Jahren an Aktualität nicht verlieren wird. Aufgrund der rapiden Verschlechterung des historischen Filmmaterials wird der Prozess der Digitalisierung und damit auch der Überarbeitung und Fehlerkorrektur an Bedeutung gewinnen. Insbesondere dann, wenn sich internationale Organisationen auf Speicherformate und Speichermedien verständigt haben, die Standard für die nächsten Jahrzehnte sein werden.

Die Forschung auf diesem Sektor wird auch deshalb weitergehen, weil die Ergebnisse sowohl für Videonachbereitung als auch für Echtzeit-Fehlerkorrektur von Videos, beispielsweise von Hobbyfilmern oder Überwachungskameras, verwendbar sind.

Literatur

[1] S. Armstrong, A. C. Kokaram, and P. J. W. Rayner. Non-linear interpolation of missing image data using Min-Max functions. In *Proceedings of IEEE International Conference on Nonlinear Signal and Image Processing (NSIP)*, September 1997.

[2] T. Bretschneider, O. Kao, and P. J. Bones. Removal of vertical scratches in digitised historical film sequences using wavelet decomposition. In *Proceedings of Image and Vision Computing*, pages 38–43, 2000.

[3] M. E. Diaz, E. Decencière, and J. Serra. A model-based method for line scratches detection and removal in degraded motion picture sequences. Technical Report 187, Centre de Morphologie Mathématique, Fontainebleau, 1999.

[4] R. Edmondson. Audiovisual archiving: Philosophy and principles. In *United Nations Educational, Scientific and Cultural Organization*, 2004.

[5] EU Commission. Digital cultural heritage. In *European Commission, Directorate-General Information Society and Media Unit Learning*, 2005.

[6] S. Pfeiffer, R. Lienhart, G. Kühne, and W. Effelsberg. The MoCA project – movie content analysis research at the University of Mannheim. In *Informatik '98: Informatik zwischen Bild und Sprache, 28. Jahrestagung der Gesellschaft für Informatik*, pages 329–338, September 1998.

[7] P. Schallauer, A. Pinz, and W. Haas. Automatic restoration for 35mm film. In *Journal of Computer Vision Research*, volume 1(3). MIT press, 1999.

[8] W.-Q. Yan and M. S. Kankanhalli. Detection and removal of lighting and shaking artifacts in home videos. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 107–116. ACM Press, 2002.