Verbesserung des Multi-Dodgings mittels bikubischer Interpolation

Martin Drauschke

martin.drauschke@uni-bonn.de

TR-IGG-P-2008-07

15. August 2008

Technischer Report Nr. 7, 2008

Professur für Photogrammetrie
Institut für Geodäsie und Geoinformation
Universität Bonn

Erhältlich unter
http://www.ipb.uni-bonn.de/~martin/

Verbesserung des Multi-Dodgings mittels bikubischer Interpolation

Martin Drauschke

15. August 2008

1 Aufgabenstellung

Digitalisierte 16-Bit-Luftbilder sollen automatisch verbessert werden. Dazu haben wir in (1) und (2) den Multi-Dodging-Ansatz vorgeschlagen. In diesem Verfahren wird ein Bild in sich nicht überlappende Ausschnitte (Patches) zerlegt. Dann wird in jedem dieser Bildausschnitte eine Histogrammverebnung durchgeführt. Da dieses Vorgehen die Patchgrenzen im verbesserten Bild hinterlässt, wurde abschließend zwischen den Patches bilinear interpoliert. In dieser Arbeit wird untersucht, ob die Verwendung einer bikubischen Interpolation an Stelle der bilinearen zu besseren Ergebnissen führt.

2 Bedienung des Programms

Der Aufruf des Programms `multidodging.exe` unterscheidet sich nur unwesentlich von seinem Vorgänger `dodging.exe`. Auch hier wird ein Grauwertbild im `tiff`-Format mit 16 Bits pro Pixel erwartet. Der Parameter `-i` wurde erweitert und kann jetzt in Verbindung mit den Werten 0, 1 und 2 verwendet werden. Der Übersichtlichkeit halber listen wir an dieser Stelle aber noch einmal alle Parameter auf und erklären ihre Bedeutung.

Ein möglicher Programmaufruf könnte beispielsweise wie folgt lauten:

```
multidodging.exe a.tif -p 500 800 -i 1 -s 0.5 -b 8 -o a2.tif -v
```

Der Dateiname des zu bearbeitetenden Bildes, hier `a.tif`, muss dem Programm als Argument übergeben werden, auch die Position des 1. Arguments nach dem Programmnamen ist dabei vorgeschrieben. Die weiteren Parameter sind optional, ihre Reihenfolge ist nicht weiter festgelegt. Die Zuordnung der Werte erfolgt durch die Schlüssel `-b`, `-i`, `-o` `-p` bzw. `-s`.

Im einzelnen haben die Parameter folgende Bedeutung.

Das Eingabebild Das Eingabebild ist ein Grauwertbild und liegt im `tiff`-Format vor. Es ist dabei egal, ob die Datei in `tiles` oder in `strips` abgespeichert wurde. Die radiometrische Auflösung des Bildes beträgt 16 Bits. Bilder mit einer

anderen geometrischen Auflösung werden zwar nicht zurückgewiesen, aber das Verfahren arbeitet dann auch nicht unbedingt so gut.

Der Parameter `-p` Das Programm `multidodging` berechnet lokale Histogrammverebnungen, d.h. Histogrammverebnungen in sich nicht überlappenden Bildausschnitten. Daher hat der Parameter `-p` für die Qualität des Ausgabebilds einen sehr großen Einfluss. Hinter dem Bezeichner `-p` müssen zwei positive, ganze Zahlen folgen. Die erste gibt die Höhe des Patches an, d.h. die Anzahl der Zeilen. Die zweite beschreibt mit der Anzahl der Spalten die Breite des Patches. Die Einteilung des Bildes in Patches erfolgt von der oberen linken Ecke, die Patches am rechten und unteren Bildrand können entsprechend kleiner sein als die anderen.

Die `default`-Werte sind 1000 und 1500 für Höhe und Breite eines Patches. Bei neuer Belegung der Werte durch den Programmaufruf wird überprüft, ob die Patchgröße sinnvoll gewählt ist. So muss ein Patch mindestens 80000 Pixel enthalten. Weniger Pixel pro Patch sind nicht sinnvoll, weil sich dann die Berechnungen der Histogrammverebnungen verschlechtern und die Grauwerttransformationen auf Histogrammen beruhen, die 65536 Grauwerte enthalten. Ferner sollte das Bild in mindestens zwölf Patches eingeteilt werden. Dabei müssen sich mindestens zwei Patches nebeneinander sowie übereinander befinden.

Der Parameter `-i` Mit dem Parameter `-i` kann die Interpolationsmethode vom Benutzer ausgewählt werden. Hinter dem Bezeichner `-i` muss eine ganze Zahl angegeben werden. Bei 0 wird die Interpolation im Programmfluss übersprungen, d.h. es werden die neuen Werte aus dem Histogramm des nächst benachbarten Patch genommen und die Patchgrenzen können im Ausgabebild sichtbar werden. Bei 1 wird im Anschluss an die lokalen Histogrammverebnungen eine bilineare Interpolation durchgeführt, bei 2 in dieser Version neu eine bikubische. Als `default`-Wert ist die bikubische Interpolation eingestellt.

Der Parameter `-s` Dieser Parameter bewirkt eine zusätzliche Stauchung der verebneten Histogramme, um die teilweise sehr starken Kontraste im bearbeiteten Bild abzumildern. Das Stauchen des Histogramms bewirkt, dass das Bild nicht mehr Grauwerte aus dem Intervall [0..65535] enthält, sondern aus einem kleineren. Die dabei verwendete Histogrammtransformation ist linear und wird aus den maximalen Gradienten jedes Bildpatches bzw. den Median der Gradienten abgeleitet. Damit der Stauchungsfaktor nicht zu klein wird, wird eine untere Schwelle überprüft. Dieser Wert wird zusammen mit dem Parameterkennzeichen `-s` als Dezimalzahl übergeben. Negative Werte und Werte größer als 1 werden abgelehnt. In diesen Fällen wird der Betrag des Wertes verwendet bzw. der Parameter wird automatisch auf 1.0 gesetzt. Dieser Wert bewirkt keine Veränderung des Histogramms. Bei Angabe einer 0, was im Prinzip einer Stauchung des gesamten Histogramms auf einen einzigen Grauwert gleichkäme, wird die Histogrammstauchung übersprungen. Der `default`-Wert des Parameters ist 0.4.

Der Parameter -b Der Parameter `-b` kann benutzt werden, wenn die radiometrische Auflösung des Ausgabebilds verändert werden soll. Hinter dem Bezeichner `-b` muss eine positive, ganze Zahl angegeben werden. Sie gibt an, mit wievielen Bits pro Pixel das Ausgabebild gespeichert werden soll. Die `default`-Belegung der Variable ist 16, d. h. das Ausgabebild wird mit derselben Grawerttiefe abgespeichert wie das Eingabebild aufweist. Alternativ kann derzeit die Ausgabedatei auch mit 8 Bits pro Pixel gespeichert werden.

Der Parameter -o Das Programm schreibt die Ausgabedatei `default`-mäßig in dasselbe Verzeichnis, in dem auch die Eingabedatei liegt. Wird der Name des Eingabebilds mit

`C:\verzeichnis\dateiname.tif`
angegeben, dann trennt das Programm die Pfadangaben vom Dateinamen und benennt das Ausgabebild

`transf_dateiname.tif`
und schreibt dieses in das Verzeichnis

`C:\verzeichnis\`.
Die Angabe des Parameters `-o` bewirkt, dass die Ausgabedatei in ein anderes Verzeichnis und unter einem anderen Namen gespeichert wird. Dem Bezeichner muss eine Zeichenkette folgen, die aus dem neuen Verzeichnis und dem neuen Dateinamen besteht.

Der Parameter -v Einzelne Zwischenergebnisse und Statistiken können über die Konsole ausgegeben werden, wenn der Parameter `-v` aufgeführt wird, der ohne zusätzliche Argumente beim Programmaufruf angegeben wird.

Hilfestellung mittel Parameter -h Bei Programmaufruf

`multidodging.exe -h`
wird eine kleine Hilfe im Eingabefenster angezeigt. Hier werden die optionalen Parameter kurz beschrieben.

3 Programmablauf

In diesem Abschnitt werden wir das Verfahren des multidodgings vorstellen. Zunächst werden wir noch einmal den bisherigen Ansatz zusammenfassen. Dann gehen wir auf die Verbesserung des Verfahrens ein, die in der Integration der bikubischen Interpolation liegt.

3.1 Bisheriger Ansatz des dodging-Verfahrens

Seien B das eingelesene und B' das bearbeitete Bild. Nach dem Einlesen von B wird das Bild in kleinere, sich nicht überlappende Bildausschnitte (Patches) P_i zerlegt. In jedem Patch wird dann eine Histogrammverebnung berechnet. Dazu werden zunächst das Histogramm H_i der Grauwerte des Eingabepatches und das kumulatives Histogramm K_i der Grauwerte des Eingabepatches sowie die Look-Up-Tabelle für die Abbildung A_i der transformierten Grauwerte bestimmt. Dieses Vorgehen ist in (1) detailiert beschrieben.

Die Transformation der Grauwerte ist bzgl. folgendes Wertebereichs definiert:

$$A_i : [0, 65535] \rightarrow [0, 65535]\,, \tag{1}$$

d. h. dass alle Grauwerte nach der Histogrammverebnung durch 16 Bit-Zahlen repräsentiert werden können. Wird das Programm `multidodging` ohne anschließende Interpolation ausgeführt, dann liefert Gleichung 1 bereits die Grauwerte für das Ausgabebild:

$$B'(x, y) = A_i\left(B(x, y)\right). \tag{2}$$

Bei einer bilinearen Interpolation werden zunächst zunächst die vier nächstgelegenen Patches des Bildes bestimmt. Das ist erstens derjenige, in dem der Punkt (x, y) liegt, und zweitens sind das diejenigen Nachbarpatches, die an den Quadranten des Patches angrenzen, indem der Punkt (x, y) liegt. Die Historgammverebnung führt oft zu sehr kontraststarken Bildern, so dass eine weitere Stauchung der Grauwerte auf ein kleineres weniger als 65536 Grauwerte enthaltendes Intervall sinnvoll sein kann.

3.2 Lineare und kubische Interpolation

Die Berechnung der bilinearen Interpolation wurde bereits in (1) ausführlich erklärt. Um den Unterschied zur bikubischen Interpolation zu verdeutlichen, beschränken wir uns hier nur auf den eindimensionalen Fall.

Gegeben seien vier Punkte (x_1, y_1), (x_2, y_2), (x_3, y_3) und (x_4, y_4). I. A. müssen diese Punkte nur eine Bedingung erfüllen, nämlich $x_i \neq x_j\ \forall i \neq j$. Da unsere Patches alle dieselbe Größe haben, können wir die Berechnungsformeln für die Geradensegmente wegen $\Delta x = x_{i+1} - x_i = 1\ \forall i$ vereinfachen und erhalten

$$y^* = y_i + (x^* - x_i)(y_{i+1} - y_i) \qquad \text{mit } x^* \in [x_i, x_{i+1}] \tag{3}$$

für beliebige Punkte (x^*, y^*) auf dem Geradensegment zwischen den Punkten (x_i, y_i) und (x_{i+1}, y_{i+1}). Bei dieser Darstellung wird bereits deutlich, dass die lineare Interpolation nur die beiden nächsten Nachbarn eines Punktes verwendet.

Bei der kubischen Interpolation wird ein Polynom dritten Grades durch alle vier Punkte gelegt, d. h. alle vier Punkte haben Einfluss auf das Ergebnis bei der Bestimmung der interpolierten Werte. Algebraisch ist bei der kubischen

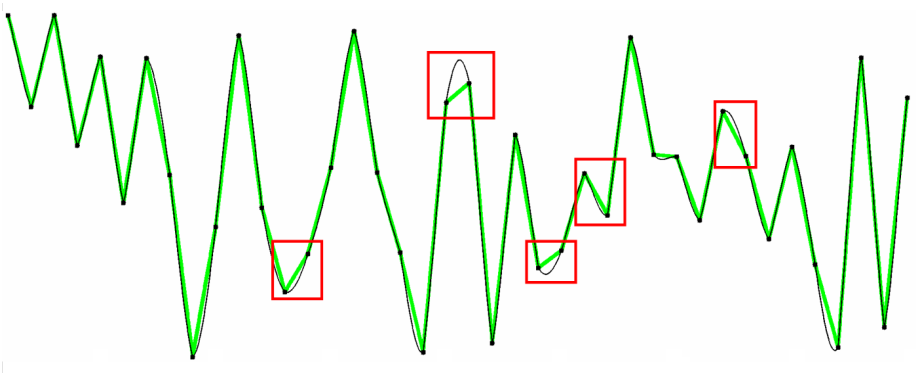


Abbildung 1: Vergleich zwischen linearer (dicke grüne Linie) und kubischer (dünne schwarze Linie) Interpolation mit 40 Stützpunkten. Geschmeidigere Übergänge gibt es vor allem bei benachbarten Stützstellen mit ähnlichen Funktionswerten.

Interpolation ein Gleichungssystem zu lösen. Mit $x_1 = -1$, $x_2 = 0$, $x_3 = +1$ und $x_4 = +2$ ergeben sich für das Polynom

$$p(x) = a + bx + cx^2 + dx^3 \tag{4}$$

die folgenden vier Bedingungen

$$\begin{array}{rcrcrcrcr} y_1 & = & a & - & b & + & c & - & d \\ y_2 & = & a & & & & & & \\ y_3 & = & a & + & b & + & c & + & d \\ y_4 & = & a & + & 2b & + & 4c & + & 8d \end{array} \tag{5}$$

und die Lösung dieses Gleichungssystems ergibt die Koeffizienten

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{1}{3} & -\frac{1}{2} & 1 & -\frac{1}{6} \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 \\ -\frac{1}{6} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}. \tag{6}$$

Um den Unterschied zwischen der linearen und der kubischen Interpolation besser zu veranschaulichen, haben wir 40 Stützpunkte mit zufällig bestimmten y_i-Werten aus dem Intervall $[0, 1]$ für beide Interpolationsmethoden verwendet. Abbildung 1 zeigt diese beiden Funktionen. Die lineare Interpolationsmethode (rot) ergibt eine Zick-zack-Kurve, die blaue Kurve der kubischen Interpolation hat an zahlreichen Stellen glattere Übergänge. Wenn man die Differenzen zwischen beiden Funktionen betrachtet, fällt auf, dass diese besonders groß sind, wenn die benachbarten Stützstellen etwa gleichgroße y-Werte haben.

Am folgenden kleinen Beispiel, siehe Abbildung 2, werden die Differenzen zwischen linearer und kubischer Interpolation besonders deutlich. Gegeben sind

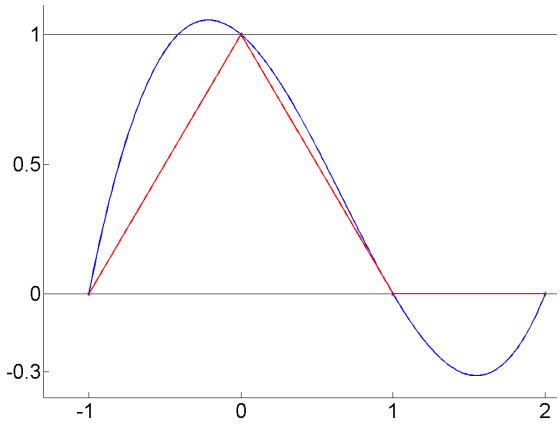


Abbildung 2: Vergleich zwischen linearer (rot) und kubischer (blau) Interpolation mit 4 Stützpunkten. Der zulässige Wertebereich für die y-Werte wird durch die beiden horizontalen Linien gekennzeichnet.

die vier Stützpunkten $(-1, 0)$, $(0, 1)$, $(1, 0)$ und $(2, 0)$, wobei die y-Werte den Wertebereich $[0, 1]$ haben. In Abbildung 2 sieht man deutlich, dass alle linear interpolierten Werte im zulässigen Intervall liegen, dagegen die kubisch interpolierten Werte auch außerhalb liegen können. Infolgedessen muss ein kubisch interpoliertes Signal nachträglich durch eine lineare Transformation auf den zulässigen Wertebereich skaliert werden.

3.3 Bilineare und bikubische Interpolation

Gegeben sei nun ein Punkt (x^*, y^*) des Bildes B, deren vier nächstgelegenen Patches P_1, P_2, P_3 und P_4 sind, siehe Abbildung 3. Die Mittelpunkte dieser Patches spannen das räumliche Gitter auf, dass einer bilinearen Interpolation zu Grunde liegt. Sie bilden unsere benötigten Stützstellen, die wir mit (x_1, y_1), (x_2, y_2), (x_3, y_3) und (x_4, y_4) bezeichnen. Die zugehörenden Funktionswerte f_i stammen aus den Histogrammverebnungen und sind

$$f_i = A_i\left(B(x^*, y^*)\right) \qquad i = 1 \ldots 4. \tag{7}$$

Die bilineare Interpolation besteht aus dem Hintereinanderausführen von drei linearen Interpolationen. Entsprechend der Darstellung in Abbildung 3 werden zunächst zwei horizontale lineare Interpolationen durchgeführt, um die Funktionswerte f_5 und f_6 der beiden zusätzlichen Stützstellen (x_5, y_5) und (x_6, y_6) zu bestimmen. Anschließend werden diese beiden neuen Stützpunkte für eine vertikale lineare Interpolation verwendet, um den neuen Grauwert f^* für die Bildposition (x^*, y^*) zu bestimmen.

Die bikubische Interpolation verläuft analog: Anstelle von vier Stützpunkten werden nun 16 benötigt, die in einem 4×4-Gitter angeordnet sind. Mittels

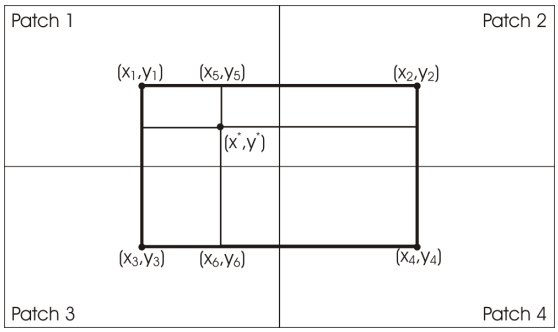


Abbildung 3: Bilineare Interpolation über vier Patches.

vier horizontaler kubischer Interpolation und einer anschließenden vertikalen kubischen Interpolation wird dann der neue Grauwert für ein Pixel bestimmt.

Wir erwarten eine ungefähr vier- bis fünffache Laufzeit für die bikubische Interpolation gegenüber der bilinearen Interpolation, weil die Gleichungen 4 und 6 gegenüber der Gleichung 3 deutlich komplexer sind. Wir haben auf eine genaue Analyse verzichtet und diese Frage ausschließlich empirisch untersucht.

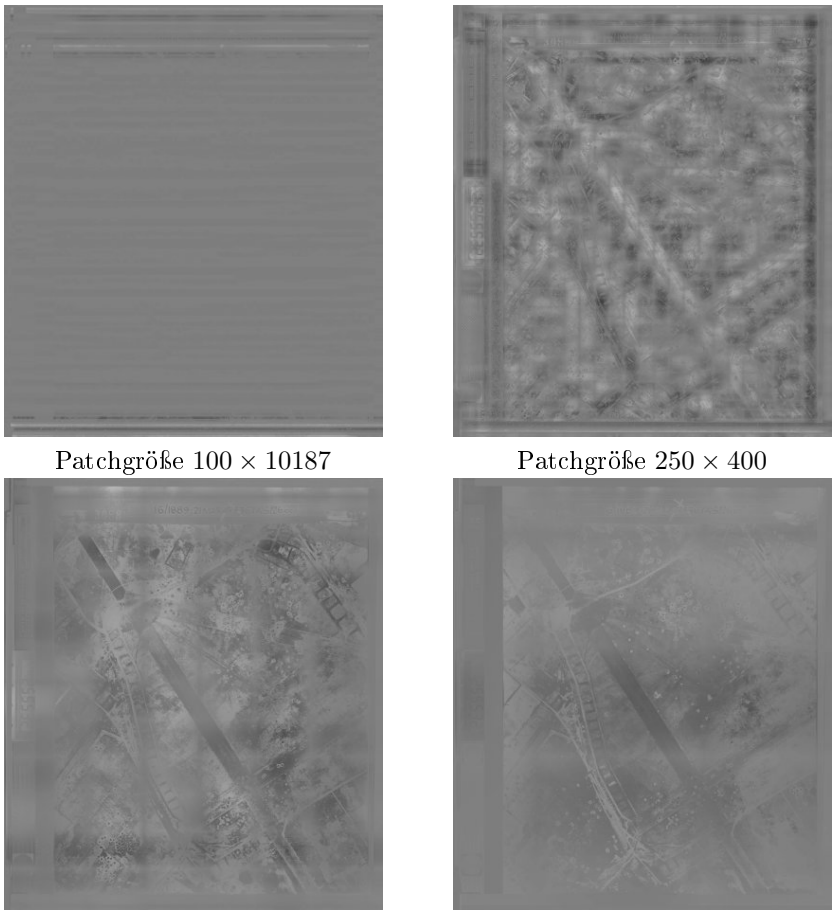
4 Vergleich zwischen bilinearer und bikubischer Interpolation

Um den Unterschied zwischen einem bearbeiteten Bild B'_l mit linearer Interpolation der Patches und einem bearbeiteten Bild B'_k mit bikubischer Interpolation feststellen zu können, berechnen wir 8-Bit-Differenzbilder D mit

$$D = 128 + 3\,(B'_k - B'_l)\,. \tag{8}$$

Bei Gleichheit der Grauwerte in beiden bearbeiteten Bildern zeigt das Differenzbild den mittleren Grauwert 128 an. An den Stellen, wo B'_k heller ist als B'_l, zeigt auch das Differenzbild helle Grauwerte (analog dunkle Grauwerte, wenn B'_k dunkler als B'_l ist). Der Faktor 3 erhöht dabei den Kontrast im Bild und wurde empirisch so festgelegt, dass die großen Differenzbeträge nicht außerhalb des gültigen Wertebereichs $[0, 255]$ liegen.

Die Differenzbilder verdeutlichen, wie stark sich die beiden Interpolationsmethoden auf die bearbeiteten Bilder B'_l und B'_k auswirken. In Abbildung4 zeigen wir die kompletten Differenzbilder, die aus der Bearbeitung eines 11619×10187-Pixel großen Bildes mit verschiedenen Patchgrößen entstanden. In den Differenzbildern werden wieder die Ränder der Patches sichtbar, weil sich entlang der Patchgrenzen die Interpolationsmethoden nicht stark unterscheiden. Es ist zudem deutlich zu erkennen, dass die Unterschiede beim zeilenweisen Multidodging kaum wahrnehmbar ist und am deutlichsten bei mittelgroßen Patches hervortreten. Dies wird auch bei der Präsentation eines Ausschnitts aus den Differenzbildern deutlich, siehe Abbildung 5.



Patchgröße 100×10187 | Patchgröße 250×400

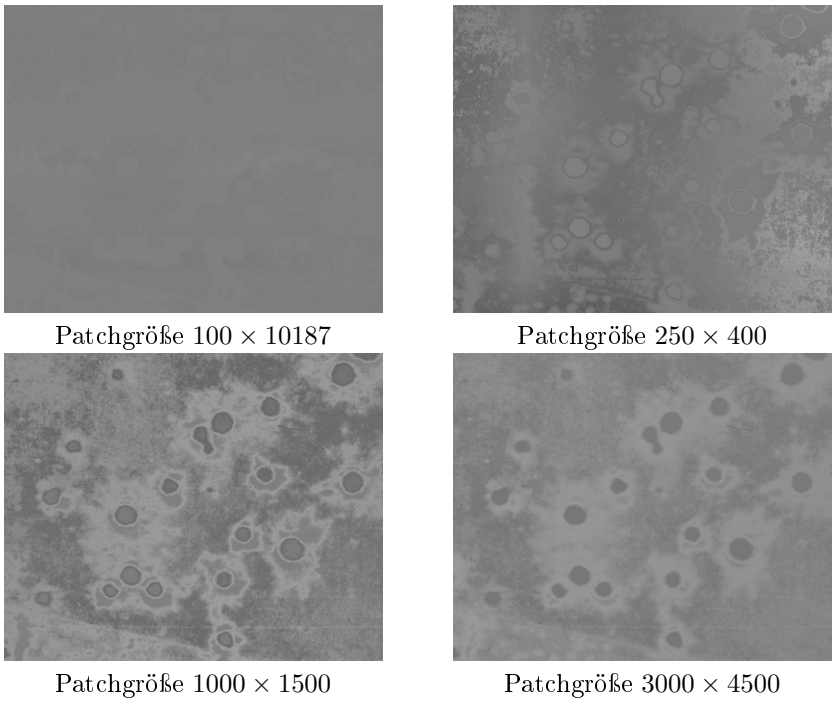
Patchgröße 1000×1500 | Patchgröße 3000×4500

Abbildung 4: Differenzbild bei unterschiedlicher Patchgröße

Unsere Versuchsbilder hatten eine Größe von ungefähr 12000×10000 Pixeln. Für das Einlesen des Bildes, das Anlegen des Speicherplatzes für die Histogramme je Patch und das Berechnen der patchweisen Histogrammverebnungen benötigt das Programm `multidodging.exe` ca. eine Minute. Die bilineare Interpolation wird in ca. drei Minuten durchgeführt, aber im Vergleich dazu braucht die bikubische Interpolation mit ca. 16 Minuten erheblich länger. Die Anzahl der Patches spielt dabei keine Rolle.

5 Schlussfolgerung

Bei rein optischer Inspektion der bearbeiteten Bilder B'_l und $B'k$ durch das menschliche Auge lassen sich bei den meisten Bildern keine Unterschiede fest-



Patchgröße 100×10187 | Patchgröße 250×400

Patchgröße 1000×1500 | Patchgröße 3000×4500

Abbildung 5: Ausschnitt aus Differenzbild bei unterschiedlicher Patchgröße

stellen. Lediglich den Differenzbildern ist zu entnehmen, dass sich beide Ausgabebilder unterscheiden. Bei globalem oder streifenweisem Dodging sind diese Unterschiede kaum erwähnenswert, bei mittlerer Patchgröße sind die Unterschiede insbesondere in den Bereichen von Bombentrichtern deutlich erkennbar.

Die Bildverbesserung mit bikubischer Interpolation benötigt ca. 17 Minuten, dasselbe Verfahren mit bilinearer Interpolation dagegen lediglich ca. vier Minuten. Der erwartete Rechenzeitgewinn durch die Verwendung größerer Patches bei der bikubische Interpolation konnte nicht erzielt werden. Da sich die Ausgabeergebnisse nicht besonders stark unterscheiden, wenn man die Bilder visuell betrachtet, bleibt die bilineare Interpolation deutlich im Vorteil.

Allerdings ist nicht auszuschließen, dass weitere automatisierten Schritte zur Bildinterpretation verbessert werden können, wenn sie die Ausgaben mit bikubischer Interpolation verwenden. Wir ziehen da vor allem die automatische Detektion von Bombentrichtern in Betracht, siehe (3).

Eine andere - von der Bildstruktur abhängige - Segmentierung des Bildes wird auf der Grundlage der vorgestellten Ergebnisse nicht zu einer wesentlichen Verbesserung des Verfahrens führen, weshalb wir dies nicht weiter untersucht haben.

Literatur

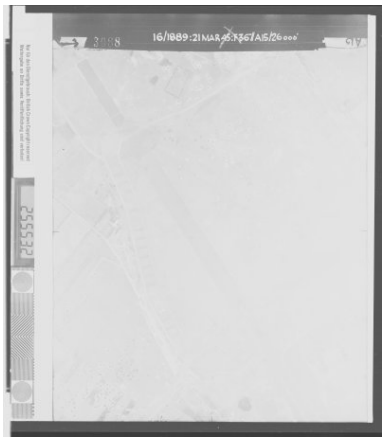
[1] Martin Drauschke: Automatisches Dodging von Luftbildern. Technischer Bericht Nr. 1, 2006 (TR-IGG-P-2006-01), Photogrammetrie, Universität Bonn, 2006.

[2] Martin Drauschke, Ansgar Brunn, Kai Kulschewski und Wolfgang Förstner: Automatic Dodging of Aerial Images. In: Eckhardt Seyfert (Hrsg.), Publikationen der DGPF: *Von der Medizintechnik bis zur Planetenforschung - Photogrammetrie und Fernerkundung für das 21. Jahrhundert*, Band 16, S. 173-180, 2007.

[3] Laura Jensen: Automatische Detektion von Bombentrichtern. Bachelorarbeit. Universität Bonn, 2008.

Anhang

In diesem Report zeigen wir unsere Ergebnisse an zwei Beispielbildern. Die Abbildungen 6 und 7 zeigen die Originalbilder von zwei Aufnahmen sowie einen Ausschnitt des rechten.



255515.tif

255532.tif

Abbildung 6: Eingescannte Luftbildaufnahmen als Eingabebilder für Multidodging.

Wir haben insgesamt 10 verschiedene Patchgrößen ausprobiert. Neben einer globalen Version (1 Patch) und zwei Versionen mit streifenförmigen Patches (in Zeilen- und Spaltenrichtung) haben wir die sieben Patchgrößen 250×400, 500×800, 800×1200, 1000×1500, 1500×2000, 2000×3000 und 3000×4500 verwendet. Beim ersten Beispiel, `255532.tif` und seinem Ausschnitt, zeigen wir alle Ergebnisse im Vergleich, beim zweiten,`255515.tif`, nur noch die Ergebnisse von vier ausgewählten Patchgrößen.

Abbildung 7: Ausschnitt mit Bombentrichtern aus Luftbild `255532.tif`.

Beispiel 1

Abbildung 8: Globales Multidodging, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 9: Zeilenweises Multidodging mit Patchgröße 100×10187, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 10: Spaltenweises Multidodging mit Patchgröße 8000×100, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 11: Multidodging mit Patchgröße 250×400, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 12: Multidodging mit Patchgröße 500×800, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 13: Multidodging mit Patchgröße 800 × 1200, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 14: Multidodging mit Patchgröße 1000 × 1500, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 15: Multidodging mit Patchgröße 1500 × 2000, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 16: Multidodging mit Patchgröße 2000×3000, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 17: Multidodging mit Patchgröße 3000×4500, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Ausschnitt aus Beispiel 1

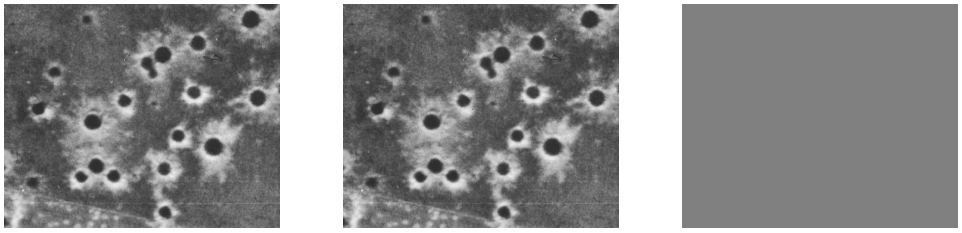


Abbildung 18: Globales Multidodging, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

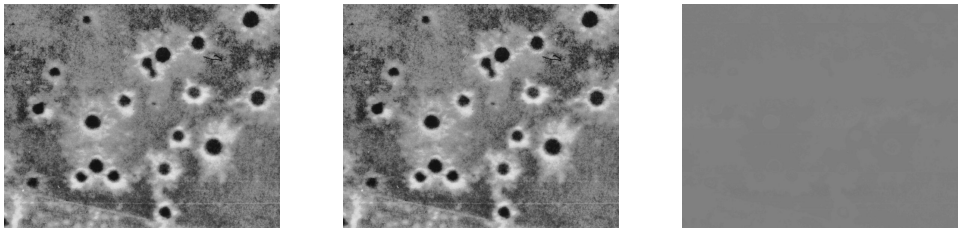


Abbildung 19: Zeilenweises Multidodging mit Patchgröße 100×10187, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

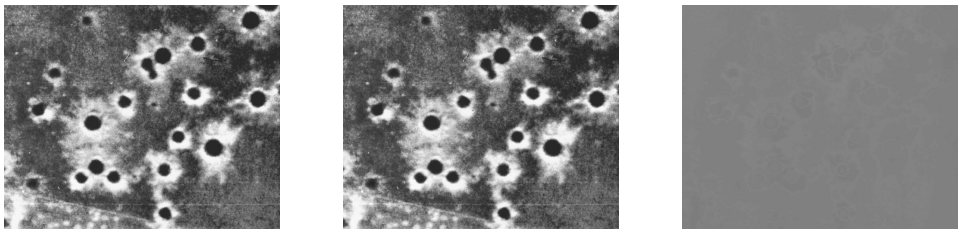


Abbildung 20: Spaltenweises Multidodging mit Patchgröße 8000×100, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

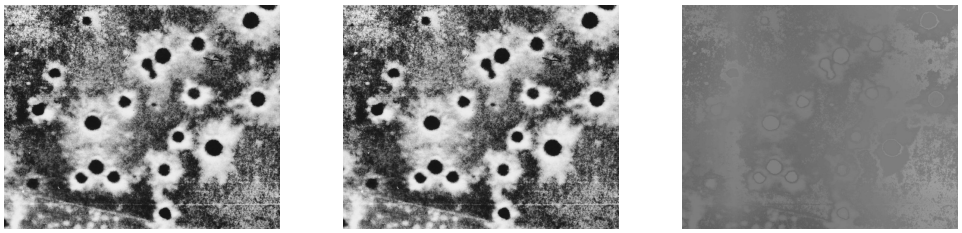


Abbildung 21: Multidodging mit Patchgröße 250×400, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

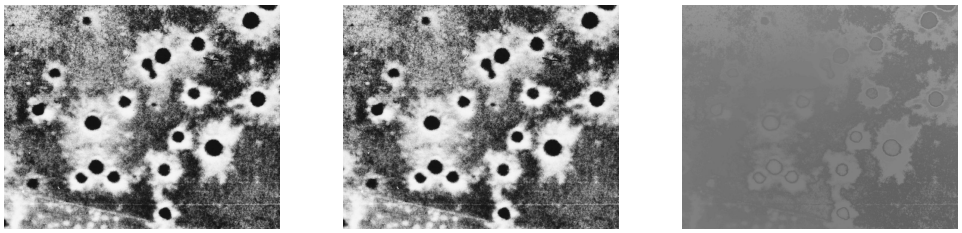


Abbildung 22: Multidodging mit Patchgröße 500×800, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

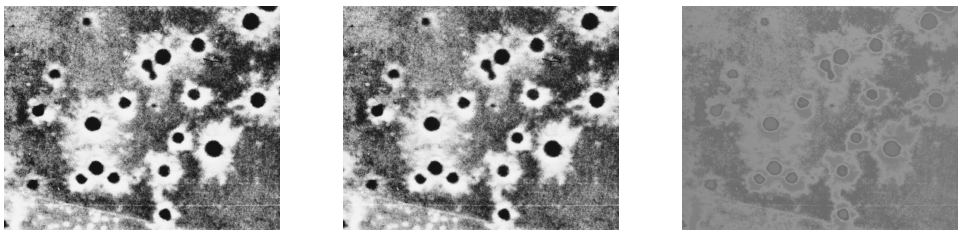


Abbildung 23: Multidodging mit Patchgröße 800×1200, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

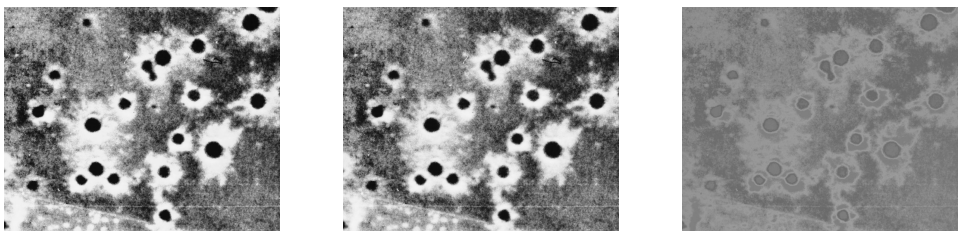


Abbildung 24: Multidodging mit Patchgröße 1000×1500, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

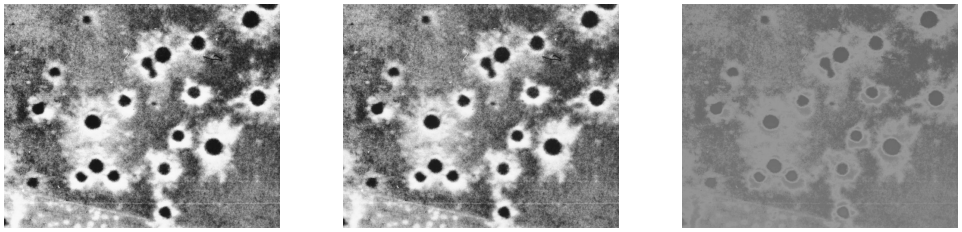


Abbildung 25: Multidodging mit Patchgröße 1500×2000, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

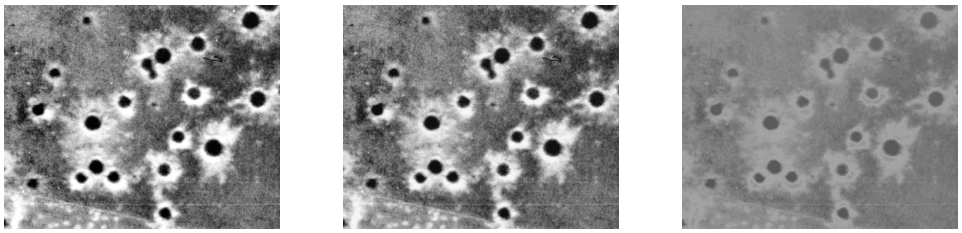


Abbildung 26: Multidodging mit Patchgröße 2000×3000, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

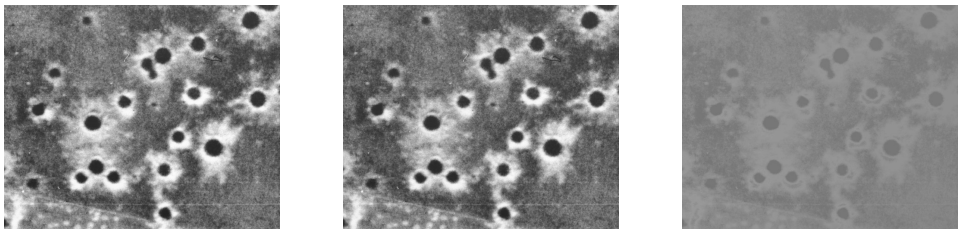


Abbildung 27: Multidodging mit Patchgröße 3000×4500, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Beispiel 2

Abbildung 28: Zeilenweises Multidodging mit Patchgröße 100×10187, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 29: Multidodging mit Patchgröße 500×800, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 30: Multidodging mit Patchgröße 1000×1500, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild

Abbildung 31: Multidodging mit Patchgröße 3000×4500, v.l.n.r.: bikubische Interpolation, bilineare Interpolation, Differenzbild