RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

LEHRSTUHL FÜR INFORMATIK VI

Prof. Dr.-Ing. H. Ney

Studienarbeit

Automatische Iris-Detektion und Merkmalsextraktion in digitalen Farbbildern des menschlichen Auges

11. Dezember 2002

Stephan Mayer

Matrikelnummer 206005

Betreuer:
Dipl.-Inform. Daniel Keysers

In Zusammenarbeit mit

Prof. Dr. med. Michael Diestelhorst

Universitäts-Augenklinik zu Köln

Inhaltsverzeichnis

1 Einführung

Die Farbe der menschlichen Iris hängt im wesentlichen von dem Pigment Melanin ab. Bei einem hohen Anteil dieses Farbstoffes in der vorderen Irisschicht, dem mesodermalen Stroma, erscheint das Auge braun, bei niedrigerem Anteil grün bis blau. [5]

Möglicherweise ist die Einwirkung von UV-Licht ein wichtiger Faktor bei der Entstehung von Farbstörungen und Tumoren in der Iris. Um dies quantitativ zu untersuchen, soll ein System entwickelt werden, das die gewünschten Bereiche aus einer digitalen Aufnahme des Auges extrahiert und die darin enthaltenen Farbverteilungen reproduzierbar und quantitativ beschreibt.

Um ausschließlich die Farben der Iris und nicht die umgebene Haut oder die restlichen Bestandteile des Auge zu analysieren, soll aus den Augenaufnahmen die Iris automatisch ausgeschnitten und in vier gleich große Teile aufgeteilt werden. Dies ist nötig um unterschiedliche Verteilungen in diesen Bereichen beurteilen zu können. Dazu werden die Farben des Ausgangsbildes in Graustufen umgerechnet und mit Hilfe des Sobel-Operators die Kanten ermittelt.

Auf dieses Sobel-Bild wird die Hough-Transformation für Kreise angewandt um den Außenkreis der Iris zu finden. Für das Auffinden des bei den Aufnahmen verwendeten ringförmigen Blitzes (s. Abbildung 1) wird die Hough-Transformation ein zweites Mal verwendet. Nach der erfolgreichen Suche wird der Mittelpunkt des Iris-Innenkreises gleich dem des Außenkreises gesetzt und der Radius so gewählt, dass er minimal bleibt und der Blitz im Innenkreis enthalten ist.

Der durch Innen- und Außenkreis begrenzte Ring wird in die Teile „top“, „bottom“, „left“ und „right“ zur Farbanalyse aufgeteilt. Auf diesen vier Teilen wird die Farbanalyse durchgeführt, wobei in einer Textdatei für jeden Bildpunkt die RGB- sowie die HSV-Farbwerte ausgegeben werden.

2 Grundlagen

Die Detektion der Iris und des ringförmigen Blitzes wird mit Hilfe der Sobel-Operatoren und der Hough-Transformation in dem verkleinerten und in Graustufen umgewandelten Bild durchgeführt.

2.1 Konvertierung und Skalierung des Bildes

Zur Ermittlung der Grauwerte wird von den drei Farbwerten (rot, grün, blau) eines jeden Bildpunktes der Durchschnittswert berechnet und auf einen ganzzahligen Wert gerundet.

Zur Skalierung eines $M \times N$-großen Bildes um den Skalierungsfaktor $\frac{1}{n}$ wird das Bild in $(\lfloor \frac{M}{n} \rfloor \times \lfloor \frac{N}{n} \rfloor)$-große Bereiche aufgeteilt und das Mittel der jeweils enthaltenen Graustufen berechnet. In der Praxis wurde $n = 10$ gewählt. Abbildung 1 zeigt eine in Graustufen konvertierte und skalierte Aufnahme eines menschlichen Auges.

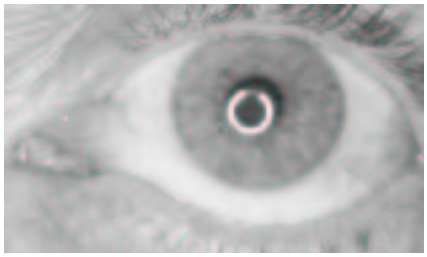

Abbildung 1: skalierte Augenaufnahme

2.2 Der Sobel-Operator

Zum Hervorheben der Kanten im Bild kommt ein Differenzfilter zur Anwendung. Der sogenannte Sobel-Operator, eine Mischung aus Differentiation und Gauß-Filter, unterdrückt während der Kantendetektion ein mögliches Bildrauschen. Er berechnet den Gradienten in der zur Randkurve orthogonalen Richtung und liefert somit geglättete Kanten. [3]

Dazu setzt man, bildlich gesprochen, nacheinander die Masken $S_x, S_y, S_/$ und $S_\backslash$ (siehe (1) - (4)) auf jeden Bildpunkt, so dass das mittlere Maskenelement ($h(0,0)$, siehe Tabelle 1) genau darauf liegt, multipliziert die Einträge in der Maske mit den jeweils darunter liegenden Farbwert und addiert die Ergebnisse.

$$S_x = \frac{1}{4}\begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} = \begin{pmatrix} \frac{1}{4} & 0 & -\frac{1}{4} \\ \frac{1}{2} & 0 & -\frac{1}{2} \\ \frac{1}{4} & 0 & -\frac{1}{4} \end{pmatrix} \tag{1}$$

$$S_y = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} = \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 0 & 0 \\ -\frac{1}{4} & -\frac{1}{2} & -\frac{1}{4} \end{pmatrix} \tag{2}$$

$$S_/ = \frac{1}{4}\begin{pmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & -\frac{1}{4} & -\frac{1}{2} \\ \frac{1}{4} & 0 & -\frac{1}{4} \\ \frac{1}{2} & \frac{1}{4} & 0 \end{pmatrix} \tag{3}$$

$$S_\backslash = \frac{1}{4}\begin{pmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} & -\frac{1}{4} & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{4} & \frac{1}{2} \end{pmatrix} \tag{4}$$

In Formel (5) ist dieser Vorgang mathematisch dargestellt. Die dabei verwendete Nummerierung der Einträge in den vier Masken ist in Tabelle 1 aufgelistet.

$$Z_m(k,l) = \sum_{i,j\in\{-1,0,1\}} f(k+i, l+j) \cdot h_m(i,j) \qquad m \in \{x, y, /, \backslash\} \tag{5}$$

Diesen Vorgang nennt man auch Faltung. [3] So erhält man durch die Anwendung aller Sobel-Masken die vier Zwischenwerte $Z_x(k,l), Z_y(k,l), Z_/(k,l)$ und $Z_\backslash(k,l)$ von denen der Maximalwert der Beträge (siehe Formel (6)) in das Lösungsbild übertragen wird.

$$S(k,l) = \max\{|Z_x(k,l)|, |Z_y(k,l)|, |Z_/(k,l)|, |Z_\backslash(k,l)|\} \tag{6}$$

h(-1,-1)	h(-1,0)	h(-1,1)
h(0,-1)	h(0,0)	h(0,1)
h(1,-1)	h(1,0)	h(1,1)

Tabelle 1: Indizierung der Einträge in den Sobel-Masken

Da die Pixel am Rand des Ausgangsbildes nicht alle für die Anwendung der Sobel-Masken benötigten Nachbarpixel besitzen, werden sie ignoriert, was zur Folge hat, dass die Lösungsmatrix jeweils in Höhe und Breite um zwei Pixel kleiner als das Ausgangsbild ist. Alternativ wäre es auch möglich, die Pixel am Rand des Bildes auf den Wert 0 zu setzen oder die Bildpunkte der gegenüberliegenden Kante des Bildes in die Rechnung einzubeziehen.

Der Sobel-Operator erkennt damit vier verschiedene Richtungen der Kanten. Berücksichtigt man noch zusätzlich das Vorzeichen, so kann man unter Verwendung der vier Sobel-Masken acht verschiedene Richtungen unterscheiden.

Abbildung 2 zeigt 3 Beispiele für die Anwendung des Sobel-Operators. In Abbildung 3 ist eine Augenaufnahme vor und nach der Verwendung der Sobel-Masken zu sehen.

Abbildung 2: Beispiele für die Anwendung der Sobel-Filterung

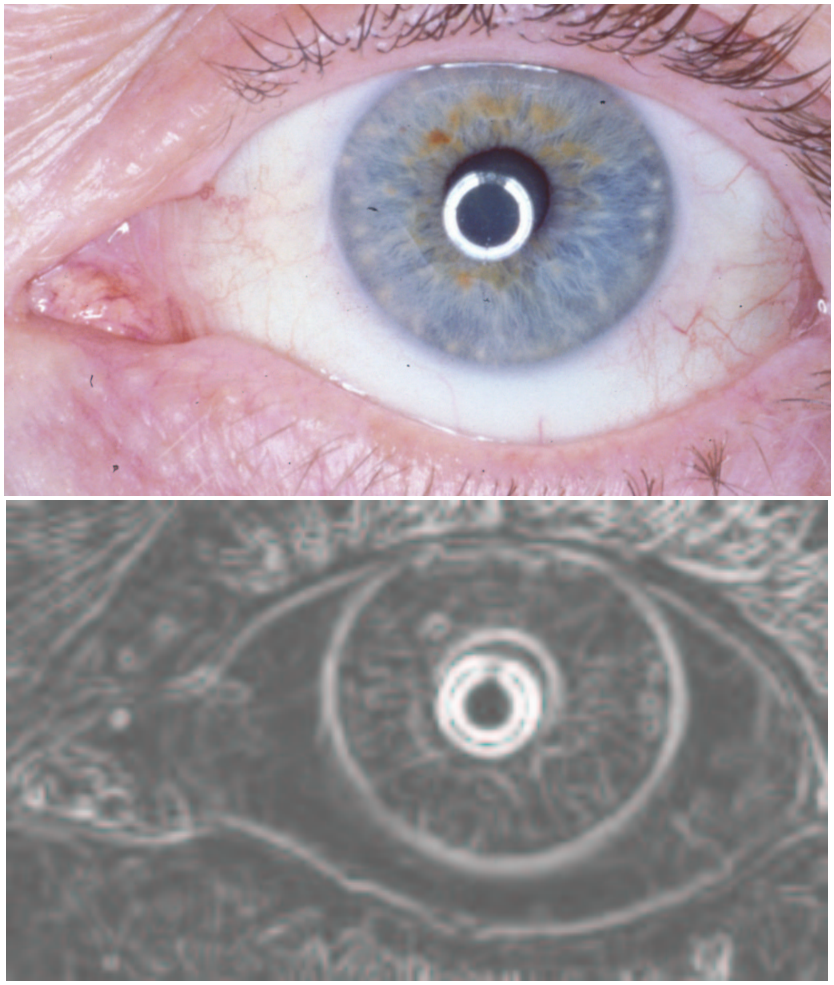

Abbildung 3: oben: Ausgangsbild; unten: graustufenkonvertiertes und um den Faktor 10 verkleinertes Ausgangsbild nach Anwendung des Sobel-Operators

2.3 Die Hough-Transformation für Geraden

Die Hough-Transformation ist eine Methode zum Auffinden von parametrisierbaren Kurven in einem Bild. Sie bildet alle Punkte von solchen Randkurven in einen Punkt des Abbildungsraumes ab [2]. Eine solche Randkurve kann beispielsweise eine Gerade sein, die durch die Gleichung

$$y = m \cdot x + b \tag{7}$$

beschrieben wird.

Um eine Aussage über das Vorkommen von Geraden in einem Bild erstellen zu können, werden die Farbwerte aller Pixel, die auf einer Geraden liegen, addiert und in einer zweidimensionalen Matrix, dem sogenannten *accumulator array*, eingetragen. Die Indizes dieses Eintrages sind dabei gleich den Parametern der betrachteten Geraden. Jeder Wert an der Stelle (m_0, b_0) in diesem Abbildungsraum beinhaltet also die Summe der Grauwerte aller Bildpunkte, die im Ausgangsbild auf der Geraden $y = m_0 \cdot x + b_0$ liegen.

Der Punkt $(2, 3)$ stellt beispielsweise die aufsummierten Bildpunkte, die auf der Geraden $y = 2x + 3$ liegen, dar.

Um ein möglichst gutes Ergebnis zu erhalten, benutzt man für die Verwendung der Hough-Transformation Bilder, die keine größeren einfarbigen Flächen enthalten. Das erreicht man beispielsweise mit der Anwendung des in Abschnitt 2.2 beschriebenen Sobel-Operators.

2.4 Die Hough-Transformation für Kreise

Wie in Abschnitt 2.3 erwähnt, lässt sich die Hough-Transformation für beliebige parametrisierbare Randkurven anwenden. So ist auch das Auffinden von Kreisen, die durch die Formel

$$(x - x_0)^2 + (y - y_0)^2 = r_0^2 \qquad x_0, y_0, r_0 = \text{konst.} \tag{8}$$

beschrieben werden, möglich.

Dazu wird für jeden Punkt (x_0, y_0) des Bildes die Summe aller Grauwerte der Punkte (x, y), die um den Radius r_0 von (x_0, y_0) entfernt sind und die Kreisgleichung (8) erfüllen, in ein dreidimensionales accumulator array an der Stelle (x_0, y_0, r_0) eingetragen.

Dabei muss man für jede mögliche Kombination von Mittelpunkt (x_0, y_0) und Radius (r_0) alle Pixel durchlaufen und die Erfüllung der Kreisgleichung überprüfen. Der Eintrag im Ergebnisraum mit dem maximalen Wert kennzeichnet die Parameter des im Bild gesuchten Kreises.

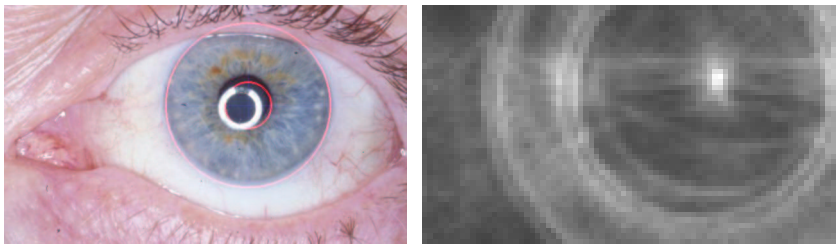

Abbildung 4: Augenaufnahme und Schnittebene (vergrößerter Ausschnitt) im accumulator array mit dem maximalen Eintrag (heller Punkt)

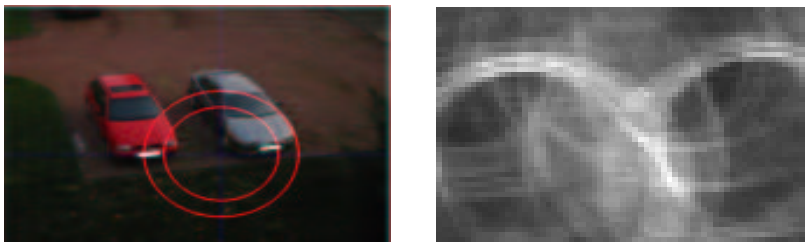

Abbildung 5: Aufnahme ohne auffindbares Kreisobjekt und Schnittebene (vergrößerter Ausschnitt) im accumulator array

Abbildung 4 zeigt eine Augenaufnahme und einen Ausschnitt eines zweidimensionalen Schnittes des Ergebnisraums mit dem maximalen Wert (hellster Punkt im Zentrum der Kreise). In Abbildung 5 ist eine Aufnahme ohne offensichtliche Kreisgeometrie zu sehen. Entsprechend sehen die Schnitte im accumulator array wie im rechten Bild aus. Die vermeintlichen Mittelpunkte liegen relativ verstreut im Array und der Betrag des maximalen Wertes liegt nur unwesentlich über dem Durchschnittswert der Beträge im Array.

Je nach Größe des gesuchten Radius werden im accumulator array unterschiedlich viele Farbwerte summiert. Darum kann man anhand des absoluten Maximalwertes nicht genau bestimmen, ob ein Kreis gefunden wurde. Aus diesem Grund definiert man ein Gütekriterium (siehe [2], Seite 19), mit dem man die Maxima der einzelnen Schnitte im Array besser vergleichen und eine Aussage über das Vorhandensein eines Kreises im Bild machen kann.

$$G_{\text{HOUGH}} = \frac{\text{Maximum im Schnittbild}}{\text{Mittelwert des Schnittbildes}} \tag{9}$$

Gleichung (9) gibt die Güte des gefundenen Kreises in einem einzelnen Schnitt des accumulator arrays mit einem bestimmten Radius an.

2.5 Optimierung der Hough-Transformation

Bei der einfachen Anwendung der Hough-Transformation für Kreise wird für jeden Bildpunkt (x_0, y_0) des Ausgangsbildes überprüft, ob er der Mittelpunkt eines Kreises mit dem Radius r_0 ist. Dabei wird für jedes andere Pixel mittels der Kreisgleichung (8) berechnet, ob dieses auf dem Kreis um (x_0, y_0) liegt.

Da im diskreten Bild nur sehr wenige Punkte, die dem Kreis zugeordnet werden können, die Kreisgleichung erfüllen, muss diese entsprechend angepasst werden.

$$(x - x_0)^2 + (y - y_0)^2 - {r_0}^2 \leq \varepsilon \quad \varepsilon \in \mathbb{R} \tag{10}$$

Der Wert für ε muss so gewählt werden, dass genügend Punkte des Kreises die Gleichung erfüllen und somit in das Accumulator Array aufgenommen werden. Ein zu großer Wert würde zu viele Pixel dem Kreis zuordnen, ein zu kleiner Wert zu wenige.

Ein weiteres Problem ist das wiederholte Überprüfen *aller* Punkte des Bildes. Kennt man die „region-of-interest“, so kann man zwar die untersuchten Punkte auf einen Bereich des Bildes begrenzen, aber es muss trotzdem eine komplette Fläche überprüft werden. Es liegt also nahe, einen Algorithmus zu finden, der entlang der Kreislinie läuft, dabei nur die zum Kreis gehörenden

Punkte findet und zugleich das Problem des unbekannten ε aus Gleichung (10) löst.

Aus diesen Gründen kommt der von Bresenham [1] entwickelte Midpoint-Algorithmus (Anhang B, [4]) zum Zeichnen von Kreisen auf pixelbasierten Medien wie zum Beispiel Computerbildschirmen oder Druckern zur Anwendung. Allerdings werden im Zusammenhang mit der Hough-Transformation die Bildpunkte ausgelesen statt gezeichnet. Wegen der Kreissymmetrie und der besseren Performance wird nur ein Kreisausschnitt von 45° betrachtet. Die restlichen Segmente lassen sich durch Vertauschen der Indizes und durch Vorzeichenänderung herleiten, wie in Abbildung 6 zu sehen ist.

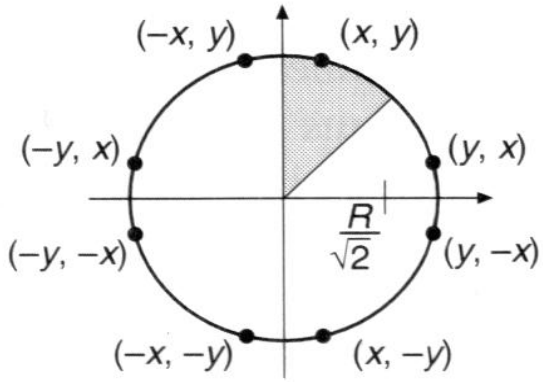

Abbildung 6: betrachtetes Kreissegment beim Midpoint-Algorithmus [1]

Betrachtet wird das Kreissegment zwischen $(x = 0, y = r)$ und $(x = y = \lceil \frac{R}{\sqrt{2}} \rceil)$ mit einem Kreismittelpunkt bei (0,0). Schreibt man die Kreisgleichung als Funktion, so erhält man

$$F(x,y) = x^2 + y^2 - R^2 \qquad x \in \{0, 1 \ldots \lceil \frac{R}{\sqrt{2}} \rceil\} \quad , y \in \{\lceil \frac{R}{\sqrt{2}} \rceil \ldots R\} \quad (11)$$

Für Punkte, die auf der Kreislinie liegen, hat die Funktion den Wert 0, für Punkte außerhalb des Kreises ist sie positiv und für Punkte innerhalb des Kreises negativ.

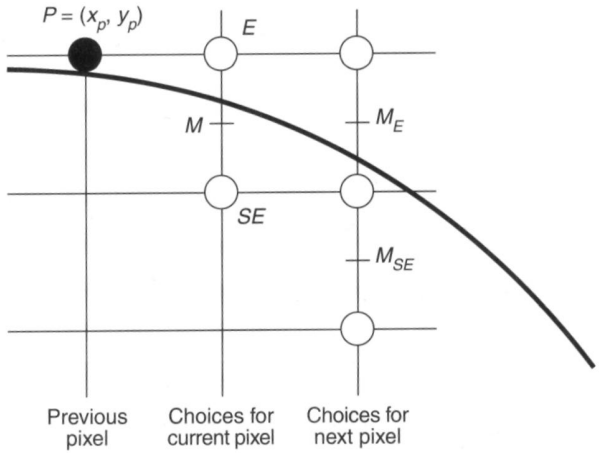

Abbildung 7: Pixelgitter mit Kreiskurve [1]

Wurde der Punkt (x_p, y_p) ermittelt, wird überprüft ob $(x_p + 1, y_p - \frac{1}{2})$ innerhalb des Kreises liegt.

$$d_{old} = F(x_p + 1, y_p - \frac{1}{2}) = (x_p + 1)^2 + (y_p - \frac{1}{2})^2 - R^2 \tag{12}$$

Ist $d_{old} < 0$, liegt der Punkt also innerhalb des Kreises und es gilt:

$$\begin{aligned} d_{new} &= F(x_p + 2, y_p - \frac{1}{2}) = (x_p + 2)^2 + (y_p - \frac{1}{2})^2 - R^2 \\ &= d_{old} + 2 \cdot x_p + 3 \end{aligned} \tag{13}$$

und $(x_p + 1, y_p)$ ist der nächste Punkt.

Ansonsten:

$$\begin{aligned} d_{new} &= F(x_p + 2, y_p - \frac{3}{2}) = (x_p + 2)^2 + (y_p - \frac{3}{2})^2 - R^2 \\ &= d_{old} + 2 \cdot x_p + 2 \cdot y_p + 5 \end{aligned} \tag{14}$$

Bildname	Zeit (s)	Auflösung
40l_001.tif	7:67	1280 x 705
40r_001.tif	7:67	1280 x 705
41l_001.tif	7:56	1280 x 699
41r_001.tif	7:56	1280 x 699
55l_001.tif	5:65	582 x 636
55r_001.tif	6:82	1101 x 690
91l_001.tif	7:73	1188 x 722
91r_001.tif	8:06	1188 x 722

Tabelle 2: Laufzeiten der Iriserkennung an 8 Testbildern

und $(x_p + 1, y_p - 1)$ ist der nächste Punkt.

Abbildung 7 verdeutlicht diesen Vorgang. Ist der Punkt $P = (x_p, y_p)$ gefunden, wird mit der Funktion (11) überprüft, ob der imaginäre Punkt M, der zwischen den Punkten E und SE liegt, inner- oder außerhalb des Kreises liegt. Liegt er innerhalb des Kreises wird SE als nächsten Punkt gewählt, ansonsten E.

In der Praxis erbrachte der Midpoint-Algorithmus auf einem System mit 256 MB Arbeitsspeicher und einem 700 MHz Duron Prozessor von AMD eine Laufzeitverringerung von ca. 4 Minuten im Durchschnitt auf etwa 5-8 Sekunden, je nach Bildgröße. Tabelle 2 zeigt exemplarisch die Laufzeiten der Iriserkennung an 8 Testbildern. Die korrekte Erkennung der Iris und des Kreisblitzes stieg ebenfalls von 81 auf 98 der benutzten 100 Testbilder.

2.6 Das RGB- und HSV-Farbmodell

Das RGB-Modell entspricht der Funktionsweise von Computer-Monitoren. Durch eine additive Mischung wird die gewünschte Farbe durch die drei Grundfarben Rot, Grün und Blau dargestellt. Die Anteile der Primärfarben liegen im Bereich $[0, 1]$. Sind die Werte der Anteile alle gleich, so beschreiben sie einen Grauton, von Schwarz ($R = G = B = 0$) bis Weiß ($R = G = B = 1$) und liegen auf der Hauptdiagonalen des in Abbildung 9 dargestellten RGB-Würfels.

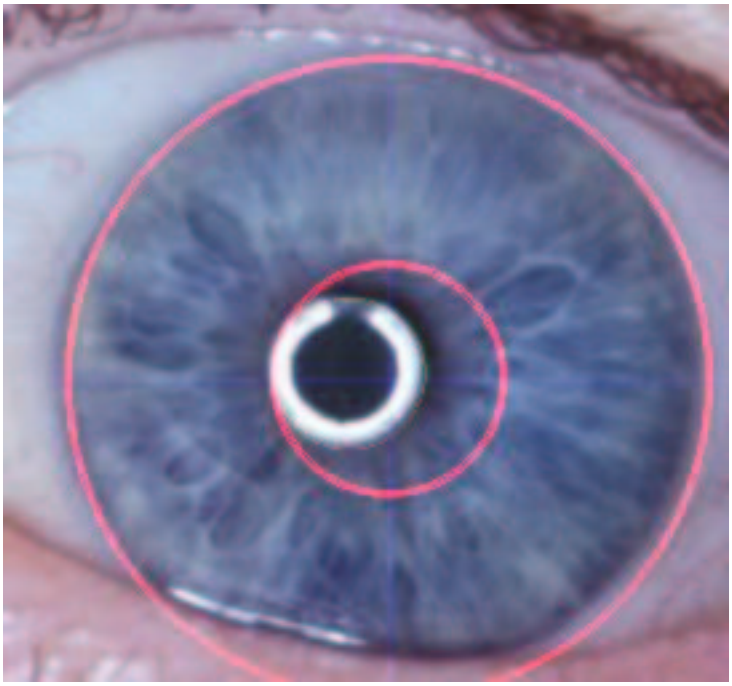

Abbildung 8: erkanntes Kreissegment nach Anwendung der Hough-Transformation

Um intuitive Farbvorstellungen zu verwirklichen, wurde von Alvy Ray Smith 1978 das HSV-Modell [7] entwickelt. Es beschreibt Farben mit dem Farbton (**H**ue), der Sättigung (**S**aturation) und der Helligkeit (**V**alue). Es läßt sich aus dem RGB-Modell durch eine nichtlineare Transformation herleiten.

$$
\begin{aligned}
V = &\quad \max(R,G,B) \qquad 0 \leq R,G,B \leq 1 \\
S = &\quad \begin{cases} \frac{V-\min(R,G,B)}{V} & \text{falls} \quad V > 0 \\ 0 & \text{falls} \quad V = 0 \end{cases}
\end{aligned}
\tag{15}
$$

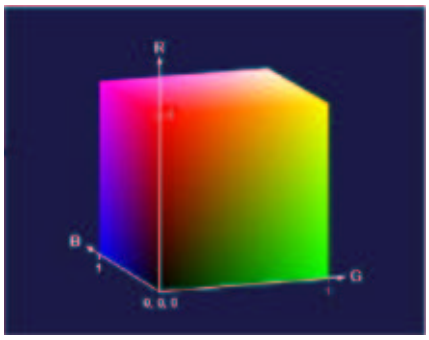

Abbildung 9: Der RGB-Würfel [7]

$$H = \begin{cases} \left\{ \begin{array}{l} (1 - \frac{R-G}{R-B}) \cdot 60° \text{ falls } \max(R,G,B) = R,\ \min(R,G,B) = B \\ (1 - \frac{G-R}{G-B}) \cdot 60° \text{ falls } \max(R,G,B) = G,\ \min(R,G,B) = B \\ (3 - \frac{G-B}{G-R}) \cdot 60° \text{ falls } \max(R,G,B) = G,\ \min(R,G,B) = R \\ (3 + \frac{B-G}{B-R}) \cdot 60° \text{ falls } \max(R,G,B) = B,\ \min(R,G,B) = R \\ (5 - \frac{B-R}{B-G}) \cdot 60° \text{ falls } \max(R,G,B) = B,\ \min(R,G,B) = G \\ (5 + \frac{R-B}{R-B}) \cdot 60° \text{ falls } \max(R,G,B) = R,\ \min(R,G,B) = G \end{array} \right\} \\ \text{falls } S > 0 \\ \\ \text{undefiniert falls } S = 0 \end{cases} \tag{16}$$

Die Helligkeit V stellt die vertikale Achse dar. Der Farbton H wird als Winkel um diese Achse angegeben. Mit S als Abstand von der vertikalen Achse ergibt sich als Farbraum ein Zylinder, dessen Grundfläche schwarz ist und auf dessen Deckfläche sich alle Farben mit $\max(R,G,B) = 1$ befinden.

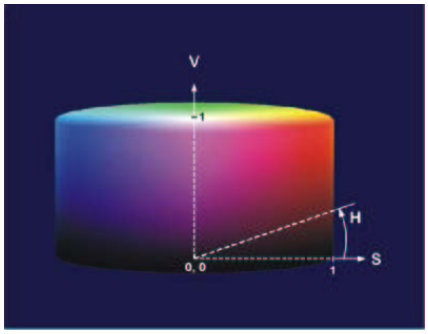

Abbildung 10: grafische Darstellung des HSV-Raumes [7]

2.7 Farbanalyse

Die Abbildungen 12 und 13 zeigen die aufsummierten Bestandteile der RGB- bzw. HSV-Kanäle. Mit Hilfe dieser Verteilungen und der in Abbildungen 14 dargestellten zwei- bzw. dreidimensionalen Darstellung der HSV-Werte soll an der Universitäts–Augenklinik zu Köln unter der Leitung von Prof. Dr. med. Michael Diestelhorst die in der Einführung beschriebenen Vermutung über die Einwirkung von UV-Licht im Auge untersucht werden.

Im Diagramm der H-Werte in Abbildung 13 ist zu erkennen, dass fast alle Werte im Bereich von 240-25° liegen. Das zeugt von einer „relativen Einfarbigkeit“ des Iris-Ausschnitts und zeigt gleichzeitig, dass keine Bestandteile des Bildes, die nicht zur Iris gehören, im Ausschnitt enthalten sind.

Letzteres kann man auch im S-Diagramm in Abbildung 13 sehen. Das Diagramm enthält keine „unbunten“ Stellen. Schwarze und weiße Bildpunkte sind also nicht im Bild enthalten.

Wie man in den Scatterplots von Abbildung 14 sehen kann, orientieren sich die Werte auf der H-S-Ebene annähernd an einer imaginären Linie. Diese eindimensionale Anordnung illustriert die in der Einführung erwähnte Abhängigkeit der Irisfarbe von dem Pigment Melanin. Die „Lücken“ in der Wolke sind auf numerische Ungenauigkeiten zurückzuführen.

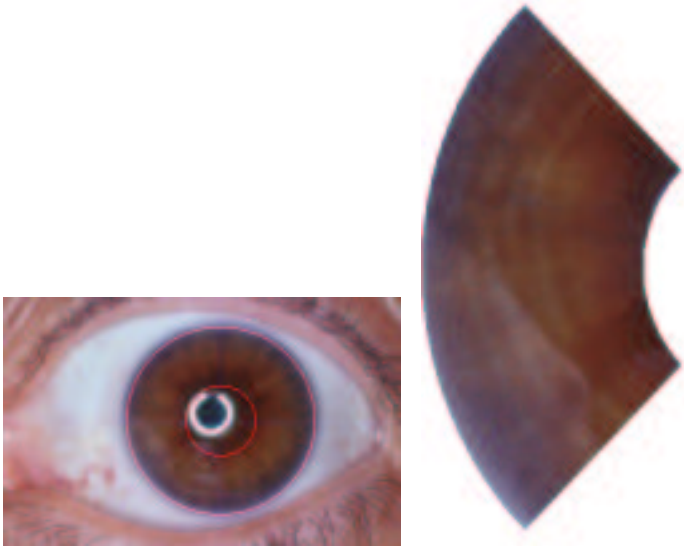

Abbildung 11: Augenaufnahme mit extrahierten Iris-Ausschnitt „left“

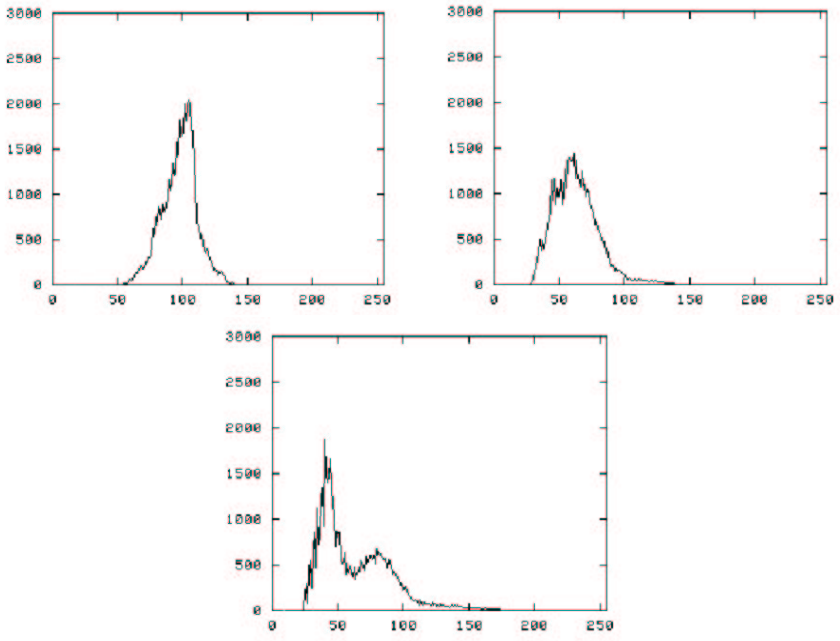

Abbildung 12: grafische Darstellung der Farbverteilungen im R-, G- und B-Farbkanal

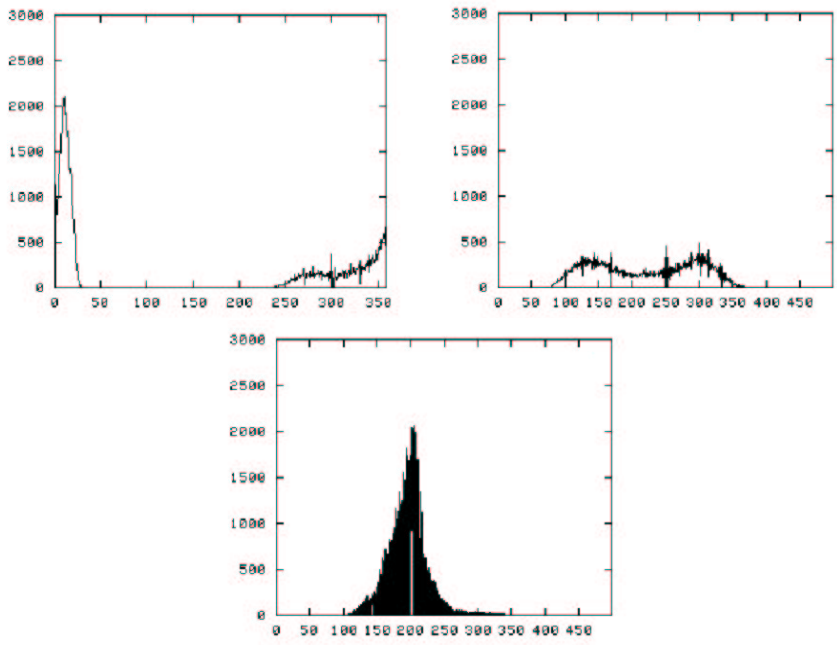

Abbildung 13: grafische Darstellung der Verteilungen im H-, S- und V-Kanal

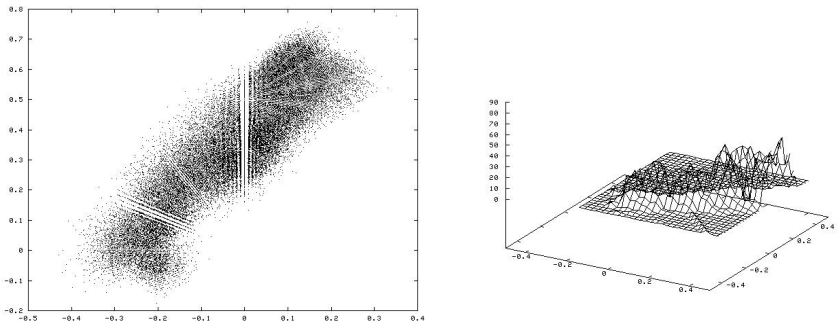

Abbildung 14: Scatter-Plot-Darstellung der HSV-Verteilungen

3 Programmablauf

Die farbigen Ausgangsbilder mit einer Größe von ungefähr 1300 × 700 Pixeln werden zur Kantendetektion in Graustufen konvertiert und auf ein Zehntel der Seitenlängen heruntergerechnet. Die Erkennung würde auch auf Bildern mit der Originalgröße funktionieren, doch wegen der wesentlich geringeren Laufzeit und der ausreichenden Erkennungsrate werden die verkleinerten Bilder zur Kantendetektion verwendet.

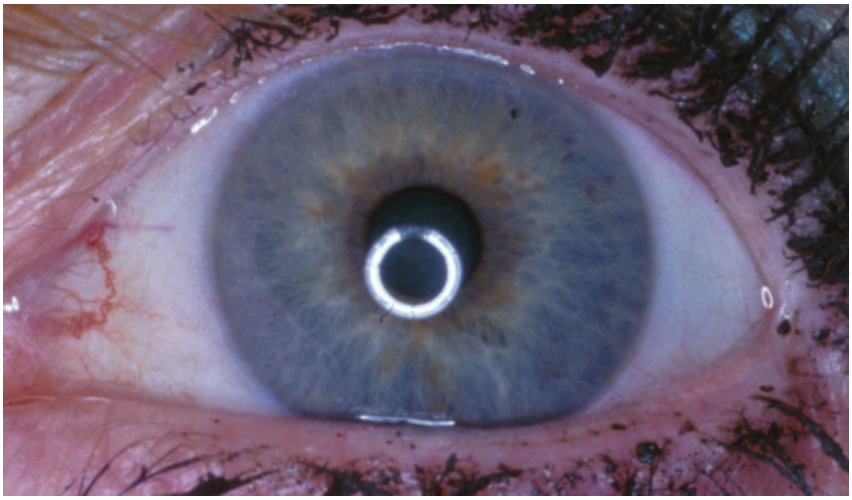

Abbildung 15: Ausgangsbild für Iris-Detektion

Auf die reduzierte Bildgröße wird, wie in Abschnitt 2.2 beschrieben, der Sobel-Operator angewendet. Abbildung 3 zeigt das Ergebnis dieser Funktion anhand einer anderen Augenaufnahme. Deutlich ist der Umfang der Iris und des bei der Aufnahme entstandenen Kreisblitzes zu sehen.

Mit diesem Sobel-Bild wird die Hough-Transformation für Kreise (siehe Kapitel 2.4) durchgeführt. Sie erkennt im ersten Schritt den Außenkreis der Iris und im zweiten Schritt den Kreisblitz. Da der Blitz in den meisten Fällen nicht genau innerhalb der Pupille liegt wird zur weiteren Betrachtung ein Kreis gewählt, der den gleichen Mittelpunkt wie der Außenkreis der Iris hat und bei kleinstmöglichstem Radius den Kreisblitz beinhaltet. Die ermittelten Kreise werden in die Originalgröße zurückgerechnet und umschließen, wie in Abbildung 16 zu sehen, einen Ring, der im Idealfall die gesamte Iris umfasst

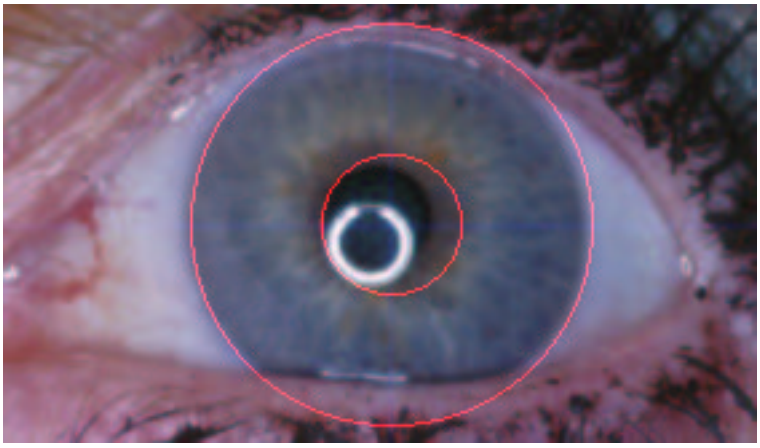

Abbildung 16: erkannte Iris (Außenkreis) und Pupille und Kreisblitz umfassender Innenkreis

und zur Farbanalyse verwendet werden kann.

Abbildung 17: Iris-Ausschnitte „top“ und „bottom“ mit Lid-überdeckung

Dazu wird dieser Ring in die vier Teile „top“, „bottom“, „left“und „right“aufgeteilt und in entsprechenden Bilddateien gespeichert. Diese Bilder haben eine Hintergrundfarbe, die nicht in den Bildausschnitten enthalten ist und die auch nicht in der Analyse berücksichtigt wird. So ist es möglich, Bildpunkte, die nicht zur Iris gehören, zu löschen indem man sie mit der Hintergrundfarbe übermalt. Das kann beispielsweise nötig werden wenn ein Lid die Iris teilweise überdeckt (siehe Abbildung 17).

Für jedes dieser Teilbilder wird eine Textdatei erstellt in der für jeden Bildpunkt die Rot-, Grün- und Blau-Werte und die dazugehörigen HSV-Werte aufgelistet werden.

Die Analyse der aufgelisteten Werte bleibt der Universitäts-Augenklinik Köln

Abbildung 18: Iris-Ausschnitte „left“ und „right“

vorbehalten. Dort sollen die in dieser Arbeit entstandenen Programme an Aufnahmen von ungefähr 4000 Patienten angewendet werden, um so die eingangs geschilderten Zusammenhänge zu untersuchen.

4 Fazit

Die hohe Erkennungsrate und die Geschwindigkeit machen die Kombination von Hough-Transformation für Kreise und Midpoint-Algorithmus zur optimalen Verbindung. So ist auch eine Anwendung auf wesentlich größere Bilder problemlos möglich. Allerdings erbrachte in der Praxis die Verwendung der Bilder in Originalgröße keine wesentlich bessere Erkennungsrate.

An der Uniklinik Köln sollen ca. 4000 Aufnahmen von menschlichen Augen fotografiert und einer Farbanalyse unterzogen werden. Auf diesem Wege soll die These von der Einwirkung des UV-Lichts bei Farbstörungen und Tumoren in der menschlichen Iris untersucht werden. Dazu wird es nötig sein, bestimmte Charakteristiken in den Diagrammen für gesunde Augen festzulegen damit man bei entsprechenden Abweichungen die Farbstörungen und Krankheiten in der Iris automatisch erkennen kann.

Bei einem positiven Ergebnis könnte es möglich werden, dass die hier gezeigten Verfahren zur Unterstützung bei Diagnosen und Screenings eingesetzt werden können.

Literatur

[1] J. E. Bresenham,
A Linear Algorithm for Incremental Digital Display of Circular Arcs,
Communications of the ACM, 20(2), 100–106, 1977.

[2] T. Lehmann,
Automatisierung der Schielwinkelmessung durch digitale Bildverarbeitung,
Diplomarbeit, Lehrstuhl für Meßtechnik der Rheinisch-Westfälischen Technischen Hochschule Aachen, Prof. Dr.-Ing. D. Meyer-Ebrecht, 1991.

[3] T. Lehmann, W. Oberschelp, E. Pelikan, R. Repges,
Bildverarbeitung für die Medizin,
Springer, 1997.

[4] J. Foley, A. van Dam, S. K. Feiner, J. F. Hughes, R. L. Phillips,
Introduction to Computer Graphics,
Addison Wesley, 1997.

[5] W. Kahle,
Taschenatlas der Anatomie,
Band 3: Nervensystem und Sinnesorgane,
Georg Thieme Verlag Stuttgart, 1991.

[6] LIBJPEG Documentation,
Using the IJG JPEG library
http://www.the-labs.com/JPEG/libjpeg.html.

[7] Gisela Schnabel,
Die Farbmodelle HSV und HLS - Widersprüche in Theorie und Praxis,
http://www.hu-berlin.de/rz/rzmit/rzm17/8.pdf
RZ-Mitteilungen Nr. 17, Februar 1999.

A Bedienungsanleitung der Programme

A.1 Voraussetzung

Die Ein- und Ausgaben der Programme sind rein textuell, so dass zur Benutzung der Programme die Eingabeaufforderung verwendet werden muss.

A.2 iris

Aufruf: `iris <Bildname> [<Ausgabeordner>]`

Beispiel: `iris 40l_001.tif output`

Das Programm `iris` erkennt automatisch im übergebenen Bild (TIFF-Format) den Außenradius der Iris und den durch die Aufnahme bedingten Kreisblitz. Aus der Position und der Größe des Blitzes und aus dem Zentrum der Iris berechnet sich ein minimaler Kreis, der im Idealfall die komplette Pupille enthält.

Der aus den beiden Kreisen resultierende Ring wird zur weiteren Berechnung in 4 Teilbereiche (top,left,right,bottom) unterteilt. Diese 4 Bilder werden im TIFF-Format im angegebenen Ausgabeordner gespeichert. Wurde kein Ordner angegeben, so werden die Bilder im aktuellen Arbeitsverzeichnis abgelegt.

Alle TIFF-Bilder werden zusätzlich im JPEG-Format für die Darstellung in einem Web-Browser ausgegeben.

A.3 iris_dir

Aufruf: `iris_dir <Bilderordner> [<Ausgabeordner>]`

Beispiel: `iris_dir bilder output`

`iris_dir` liest alle TIFF-Bilder aus dem Bilderordner aus und ruft `iris` mit dem Bildnamen und dem Ausgabeverzeichnis auf. Zusätzlich wird eine

HTML-Datei in dem Ausgabeordner erstellt. Sie zeigt eine Übersicht über die bearbeiteten Bilder und die vier jeweils dazugehörigen Teilbereichsbilder.

In dieser HTML-Datei werden die Bilder nur im JPEG-Format dargestellt. Da aber die Diagramme und Analysen ausschließlich anhand der TIFF-Bilder durchgeführt werden, müssen eventuelle manuelle Veränderungen an den entsprechenden TIFF-Dateien vorgenommen werden.

A.4 analyse

Aufruf: `analyse <Iris-Teilbild> [<Ausgabedatei>]`

Beispiel: `analyse 40l_001_bottom.tif ergebnis.dat`

`analyse` liest ein Bild im TIFF-Format ein und erstellt eine Ausgabedatei mit den RGB- und HSV-Werten der Bildpunkte im übergebenen Bild. Für jeden Pixel wird eine Zeile in der Form R;G;B;H;S;V geschrieben.

Bildpunkte, die gleich der Hintergrundfarbe sind, werden dabei ignoriert. Die Hintergrundfarbe ist standardmäßig auf Weiß (RGB: 255,255,255) eingestellt.

A.5 analyse_dir

Aufruf: `analyse_dir <Ordner mit Iris-Teilbildern> <Ausgabeordner>`

Beispiel: `analyse_dir output ergebnisse`

Mit `analyse_dir` wird `analyse` auf alle TIFF-Bilder im übergebenen Ordner angewendet. Die Ausgabe von `analyse` wird in einer Datei mit einem Namen bestehend aus Bildname und ".dat"-Endung im Ausgabeordner gespeichert.

B Midpoint-Algorithmus

```
void MidpointCircle(int radius, int value)
{
  int x,y;
  float d;
  x = 0;
  y = radius;

  d = 5.0/4 - radius;
  CirclePoints(x,y,value);
  while (y>x) {
    if (d<0) {
      d += x * 2.0 + 3;
      x++;
    } else {
      d += (x - y) * 2.0 + 5;
      x++;
      y--;
    }
    CirclePoints(x,y,value);
  }
}


void CirclePoints(float x, float y, int value)
{
  WritePixel( x, y, value);
  WritePixel( y, x, value);
  WritePixel( y,-x, value);
  WritePixel( x,-y, value);
  WritePixel(-x,-y, value);
  WritePixel(-y,-x, value);
  WritePixel(-y, x, value);
  WritePixel(-x, y, value);
}
```