2 Theoretische Grundlagen

Dieses Kapitel beschäftigt sich mit den wissenschaftlichen Grundlagen, die zum Verständnis dieser Arbeit beitragen. Es werden zunächst in Abschnitt 2.1 die wichtigsten Farbmodelle RGB und HSV sowie deren Umwandlung ineinander beschrieben. Ebenso wie die Konvertierung in ein Grauwert- und Binärbild, da diese bei der Fahrspurerkennung oft zum Einsatz kommen. Anschließend wird in Abschnitt 2.2 die Hough-Transformation von Geraden als letzter Arbeitsschritt in der Spurerkennung erklärt und verschiedene Ansätze wie die „Randomized Hough Transform“ erläutert.

2.1 Farbumwandlungen

Es gibt zwei gute Gründe, um Farben in der Bildverarbeitung zu nutzen. Zum einen ist es dem Menschen möglich, tausende verschiedene Farbschattierungen und Intensitäten zu unterscheiden, wohingegen es gerade einmal zwei Dutzend Grauschattierungen sind. Und zum anderen ist die Farbe an sich ein guter Deskriptor, mit dessen Hilfe es ein Leichtes ist, Objekte zu erkennen. [Gonz08]

Um Bilder besser verarbeiten zu können, werden diese oft in ein anderes Farbmodell konvertiert. Ausgangspunkt ist meistens ein RGB-Modell, das man anschließend beispielsweise in ein HSV- oder Graustufenbild konvertieren kann. Nach einer solchen Umwandlung fällt es meist leichter, die gewünschten Formen oder Objekte zu segmentieren. Ein Graustufenbild kann außerdem in einem weiteren Schritt in ein Binärbild konvertiert werden. Dieses Vorgehen wird auch Binarisierung genannt.

2.1.1 RGB-Modell

Das wohl bekannteste technik- oder hardwareorientierte Farbmodell ist das RGB-Modell, das beispielsweise bei der Farbdarstellung auf Fernsehgeräten oder Computern verwendet wird. Die drei Grundfarben dieses Modells sind die Farben Rot, Grün und Blau, die in der zugrunde liegenden Normtafel der „Commission Internationale de l'Eclaire", kurz CIE, als Primärfarben festgelegt wurden. Die CIE-Normfarbtafel in Abbildung 2.1 wurde 1931 als zweidimensionaler Farbraum definiert, um „eine Normierung und Vergleichbarkeit zu erreichen". ([Nisc11], S. 43) Da sich diese Darstellung an dem Helligkeitsempfinden des menschlichen Auges orientiert, ergibt sich dadurch die unverwechselbare Hufeisenform.

Wählt man drei Eckpunkte und spannt ein Dreieck zwischen ihnen auf, so erhält man ein CIE-Farbdreieck, dass je nach den gewählten Punkten eine andere Farbskala (engl. Color Gamut) definiert. Innerhalb dieser Farbskala können alle Farben durch additives[1] Farbmischen der Eckfarben erzeugt werden. Ein mögliches RGB-Dreieck moderner Farbmonitore ist ebenfalls in Abbildung 2.1 zu sehen. Daran ist zu erkennen, dass nicht alle sichtbaren Farben im RGB-Modell darstellbar sind.

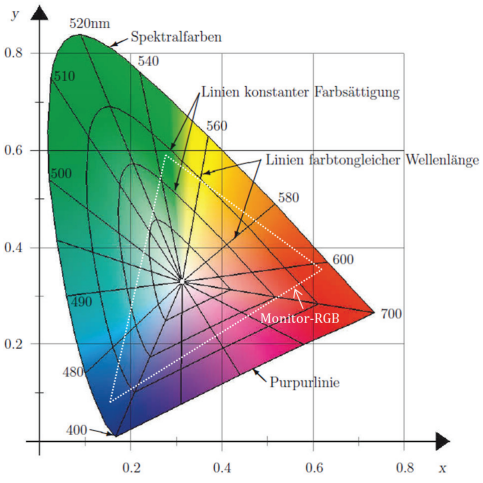

Abb. 2.1: CIE-Normfarbtafel [1]

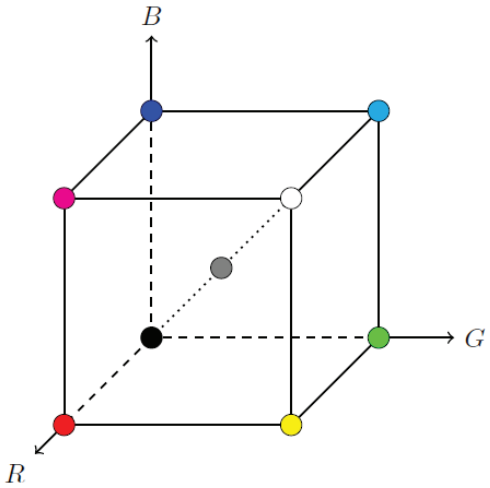

Abb. 2.2: RGB-Darstellung als Würfel [2]

Eine weitere Möglichkeit, sich den Farbraum vorzustellen, bildet die Darstellung des RGB-Modells als normierter dreidimensionaler Vektorraum mit den drei Grundfarben Rot, Grün und Blau als Basisvektoren und den Grauwerten auf der Hauptdiagonalen von Schwarz

1 Additive Farbmischung bezeichnet die Überlagerung von ausgestrahltem Licht.
2 In Anlehnung an: [Nisc11], S. 44, Abb. 3.13
3 In Anlehnung an: [Burg15], S. 310, Abb. 12.1, [Gonz08], S. 402, Abb. 6.7

zu Weiß. Die Werte liegen nun zwischen 0 und 1 anstatt, wie bei Digitalbildern üblich, zwischen 0 und 255. Dieser sogenannte Einheitswürfel ist in Abbildung 2.2 zu sehen. Jeder Punkt innerhalb dieses Würfels kann als Linearkombination der drei Basiskomponenten Rot, Grün und Blau dargestellt werden. [Süße14] [Burg15] [Henn07]

Ein RGB-Bild besteht ebenfalls aus drei Komponentenbildern, die sich überlagern und somit das fertige Farbbild bilden. Bestehen die Komponentenbilder jeweils aus 8 Bits, so besteht das RGB-Bild dementsprechend aus 24 Bits und kann somit $(2^8)^3 = 16.777.216$ Farben darstellen. Die meisten Monitore können jedoch nur deutlich weniger Farben abbilden. Aus diesem Grund wurde ein Standard für sogenannte „safe colors“ festgelegt, um die Farbdarstellung auf unterschiedlichen Wiedergabegeräten möglichst einheitlich zu halten. In den Abbildungen 2.3 und 2.4 ist der Unterschied zwischen einem vollständigen RGB-Würfel und einem RGB-Würfel mit „sicheren Farben“ zu erkennen.

Es wird bei den „safe colors“ davon ausgegangen, dass 256 die minimale Anzahl an Farbkombinationen der RGB-Werte ist, die von den meisten Bildschirmen dargestellt werden können. Davon fallen 40 weg, von denen bereits bekannt ist, dass sie von den verschiedenen Betriebssystemen unterschiedlich verarbeitet werden. Die verbleibenden 216 Farben bilden schließlich die „sicheren Farben“. Hierbei können die RGB-Werte jeweils nur einen von sechs Werten annehmen, also gilt: $6^3 = 216$. Die sechs möglichen Werte sind 0, 51, 102, 153, 204 und 255. Diese werden gewöhnlich in Hexadezimalwerten angegeben und so steht beispielsweise FFFFFF für die Farbe Weiß. [Gonz08]

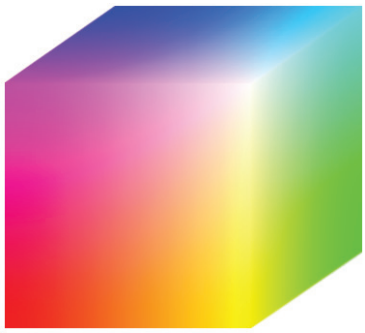

Abb. 2.3: RGB-Würfel bunt [1]

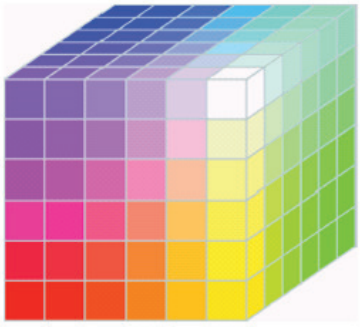

Abb. 2.4: RGB-Würfel der „safe colors“ [2]

Bei der Arbeit mit OpenCV ist darauf zu achten, dass die Reihenfolge in einem Farb-Array BGR anstatt RGB ist.

1 [Gonz08], S. 403, Abb. 6.8
2 [Gonz08], S. 406, Abb. 6.11

2.1.2 HSV-Modell

Das HSV-Modell wurde in den 1970er Jahren entwickelt, um eine einfache numerische Spezifikation für die Bildverarbeitung zur Hand zu haben. [Hanb03]

Im Gegensatz zu dem RGB-Modell gehört das HSV-Modell zu den wahrnehmungs- oder benutzerorientierten Farbmodellen und richtet sich nach der menschlichen Farbwahrnehmung. Dies hat den Vorteil, dass sich die Farben nun gut aus menschlicher Sicht beschreiben lassen - im Gegensatz zur Beschreibung der RGB-Farben. [Gonz08]

Die drei charakteristischen Größen sind in diesem Fall der Farbton (engl. Hue), die Farbsättigung (engl. Saturation) und die Farbhelligkeit (engl. Value). Dabei steht die Farbsättigung für die Menge an weißem Licht, die in der Farbe enthalten ist. Dementsprechend sieht man bei Sättigung 0 keine Farbe sondern nur Schwarz, Weiß oder Grautöne. Diese wiederum unterscheiden sich durch ihre Lichtintensität, also der Gesamtmenge an Licht. [Süße14]

Dadurch, dass das HSV-Modell den Farbanteil (Hue, Saturation) von dem Grauanteil (Value) klar trennt, ist es ideal für Bildverarbeitungsalgorithmen, die auf Farb-Deskriptoren basieren und somit auch für die Menschen, die diese entwickeln oder mit ihnen arbeiten. [Gonz08]

Eine weitere Bezeichnung für dieses Farbmodell ist das HSB-Modell. (engl. Brightness) Das HSI-Modell (engl. Intensity) hat zwar die gleiche Form, seine Werte werden jedoch unterschiedlich berechnet. Das HSL- (engl. Luminance) oder auch HLS-Modell hingegen bezeichnet eine etwas abgewandelte Version des HSV-Modells, bei dem der Weißpunkt nicht auf der gleichen Ebene mit den reinen Farben liegt, sondern wie der Schwarzpunkt davon getrennt ist. Es entsteht dadurch eine „Doppelpyramide“ und die Berechnung der einzelnen Werte unterscheidet sich ebenfalls. [Burg15]

Ein weiterer Unterschied zu dem RGB-Modell besteht in dessen Visualisierung. Anstelle eines Würfels wird das HSV-Modell „traditionell in Form einer umgekehrten, sechseckigen Pyramide“ ([Burg15], S. 326), einem Hexcone, anschaulich dargestellt (Abb. 2.5), oder mathematisch korrekt in der Form eines Zylinders (Abb. 2.6). Eine weitere Möglichkeit ist die Darstellung als umgekehrter Kegel (Abb. 2.7). Bei jeder Form steht die Längsachse für die Farbhelligkeit, der Radius des Zylinders für die Farbsättigung und der Winkel um die Längsachse herum für den Farbton. [Süße14] [Henn07]

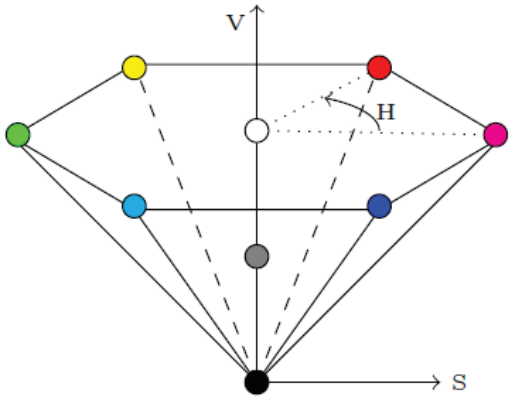

Abb. 2.5: HSV-Darstellung als Pyramide[1]

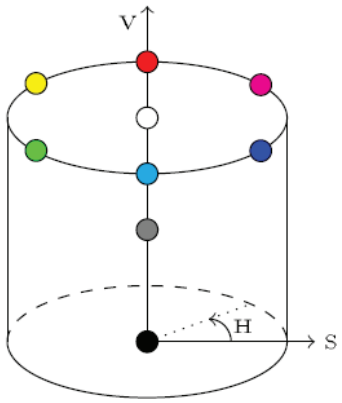

Abb. 2.6: HSV-Darstellung als Zylinder[2]

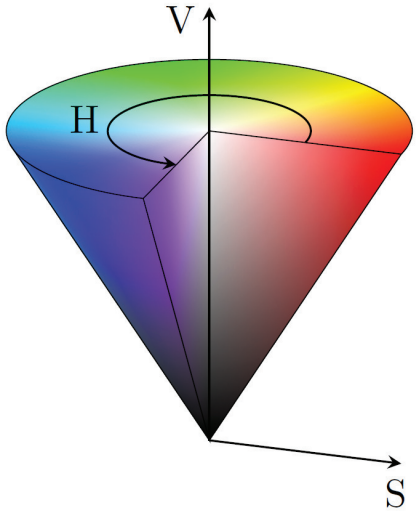

Abb. 2.7: HSV-Darstellung als Kegel

2.1.3 Konvertierung RGB → HSV

Möchte man nun ein Bild aus dem RGB-Modell in das HSV-Modell konvertieren, so muss die Diagonale des RGB-Würfels, die von Schwarz zu Weiß führt, die sogenannte „Unbuntgerade“ mit R=G=B, auf der alle Grauwerte liegen, in die Längsachse des HSV-Zylinders transformiert werden. Dafür wird das Koordinatensystem des RGB-Modells so rotiert, dass die Unbuntgerade senkrecht nach oben zeigt, wie in Abbildung 2.8 zu sehen. [Süße14]

1 In Anlehnung an: [Burg15], S. 326, Abb. 12.11a
2 In Anlehnung an: [Burg15], S. 329, Abb. 12.12

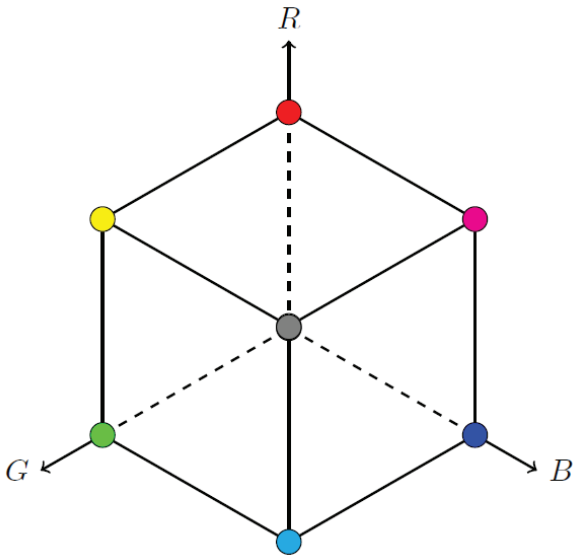

Abb. 2.8: RGB-Würfel rotiert

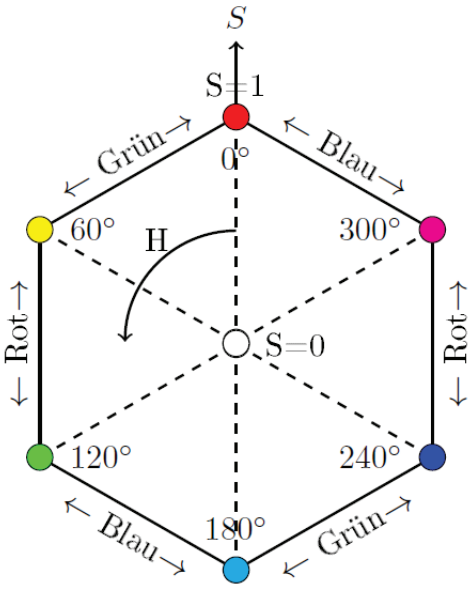

Abb. 2.9: HSV-Hexagon von oben

Projiziert man diesen rotierten Würfel nun auf eine Farbebene (engl. chromatic plane), so erhält man das HSV-Sechseck oder auch Hexagon in Abbildung 2.9, das alle reinen Farben und den Weißpunkt enthält. In dieser Ebene liegen alle Farbinformationen, bestehend aus Farbton (H), als Winkel um die Unbuntgerade, und Farbsättigung (S), als Abstand von der Unbuntgerade. Die Farbhelligkeit (V) hingegen verläuft entlang der Unbuntgerade von dem Schwarzpunkt außerhalb der Farbebene zum Weißpunkt innerhalb der Ebene. [Hanb03]

Durch die Verbindung aller Punkte mit dem Schwarzpunkt entsteht eine auf dem Kopf stehende sechseckige Pyramide (Hexcone) wie in Abbildung 2.5.

Für die Farbhelligkeit V gilt:

$$V = max(R, G, B) \qquad \text{für } R,G,B \in [0,1] \tag{2.1}$$

Dies wurde so festgelegt, da auf diese Weise alle reinen Farben, also alle Farben mit mindestens einer Eins als Komponente, wie bereits erklärt auf einer Ebene liegen. Man kann es sich nach Smith auch so vorstellen, dass die drei Komponenten R,G und B als Balken dargestellt werden. V wäre dann einfach die Höhe des längsten Balken. [Smit78]

Wie man an der Gleichung 2.2 sehen kann, ist die Voraussetzung für eine Konvertierung aus dem RGB- in den HSV-Farbraum das Vorliegen der RGB-Werten, entsprechend dem

Einheitswürfel, in einem Wertebereich zwischen 0 und 1. RGB-Werte eines 8-Bit-Bildes liegen jedoch in einem Wertebereich zwischen 0 und 255 und müssen dementsprechend noch normiert werden. Dies kann entweder im Voraus geschehen oder mit in die Berechnung von V eingebunden werden:

$$V = \frac{max(R,G,B)}{255} \qquad \text{für } R,G,B \in [0,255] \tag{2.2}$$

Die Farbhelligkeit V ist somit ebenfalls $\in [0,1]$. [Hanb03]

Die Sättigung S bezeichnet, wie bereits erwähnt, den euklidischen Abstand von der Unbuntachse. Sie kann außerdem als Differenz von max(R,G,B) und min(R,G,B) berechnet werden.

$$S = max(R,G,B) - min(R,G,B) \tag{2.3}$$

Diese Berechnung ergibt sich daraus, dass sich bei den Außenlinien des Sechsecks immer nur eine RGB-Komponente ändert, während die beiden anderen Komponenten als Maximal- und Minimalwert konstant bleiben, weshalb bei dieser Rechnung auch die Sättigung S konstant bleibt (Abb. 2.9). [Hanb07] Außerdem zeigen alle R = G = B auf die Unbuntachse und somit gilt in diesen Fällen immer S = 0, da wir uns nur vertikal entlang der Achse bewegen. Ist eine Komponente gleich null, so ist auch das Minimum aller Komponenten null und somit S das Maximum der restlichen beiden Komponenten. [Chaw15]

Die Hälfte aller möglichen Farben im HSV-Raum ist nicht Teil des RGB-Modells, was an der spitz zulaufenden Form der Pyramide zu erkennen ist. Hat man beispielsweise einen niedrigen V- und einen hohen S-Wert, so würde man außerhalb des Farbraums landen. Aus diesem Grund wird die Sättigung S so angepasst, dass der ganze Bereich von 0 bis 1 ausgefüllt wird - es entsteht ein Zylinder wie in Abbildung 2.6. [Hanb07] [Hanb03]

Für die Sättigung S gilt nun:

$$S = \begin{cases} \frac{max(R,G,B)-min(R,G,B)}{max(R,G,B)} & \text{für } max(R,G,B) \neq 0 \\ 0 & \text{sonst.} \end{cases} \tag{2.4}$$

Diese Darstellung ist wesentlich unkomplizierter für den Anwender, kann jedoch auch zu Fehlern während der Transformation führen, wie von Hanbury und Serra gezeigt. Durch die Zylinderform wird Schwarztönen, die nicht exakt auf dem Schwarzpunkt liegen, eine

Sättigung zugeschrieben, die diese eigentlich nicht haben dürften. Zudem ist in 2.4 zu erkennen, dass keine klare Trennung mehr von Farbe und Helligkeit besteht, da S von V abhängig ist. [Hanb03]

Schlussendlich ist noch der Farbton H der umgewandelten Farbe zu berechnen. Dieser beginnt traditionell mit 0° bei Rot und bewegt sich von da an gegen den Uhrzeigersinn.

H ist für Grautöne, also für S=0, undefiniert. Der Einfachheit halber wird jedoch oft der Wert 0° angegeben. Allgemein gilt für den Farbton H:

$$H = \begin{cases} \text{Undefiniert } (0°) & \text{für } S = 0 \\ 60° \cdot \frac{G-B}{max(R,G,B)-min(R,G,B)} + 0° & \text{für } max(R,G,B) = R \\ 60° \cdot \frac{B-R}{max(R,G,B)-min(R,G,B)} + 120° & \text{für } max(R,G,B) = G \\ 60° \cdot \frac{R-G}{max(R,G,B)-min(R,G,B)} + 240° & \text{für } max(R,G,B) = B \end{cases} \tag{2.5}$$

$$\text{Für H} < 0° \text{ gilt: } H = H + 360° \tag{2.6}$$

Durch die Multiplikation mit 60° werden die Werte von H in Gradzahlen umgerechnet. Dabei steht diese für jeweils einen Abschnitt im Hexagon. Alle sechs Abschnitte zusammen bilden eine volle Umdrehung mit 360°. Durch die Addition des passenden Winkels wird die Startposition auf die jeweilige maximale Komponente angepasst. [Chaw15], [Open4]

In den Abbildungen 2.10 und 2.11 ist das Ergebnis der BGR2HSV-Funktion `cvtColor (image, hsv_image, COLOR_BGR2HSV_FULL)` der OpenCV Bibliothek im Vergleich zum Ursprungsbild zu sehen. Die Funktion kann auf zwei Arten ausgeführt werden. Zum einen mit dem Übergabeparameter `COLOR_BGR2HSV`, bei dem der Wertebereich von H auf die Hälfte von 0-180° beschränkt ist, und zum anderen mit dem Parameter `COLOR_BGR2HSV_FULL`, mit dem vollständigen Wertebereich für H von 0-360°.

Das Bild wird so bunt dargestellt, da OpenCV die Werte als RGB-Werte interpretiert. Möchte man die die wirklichen HSV-Werte darstellen, so muss man die einzelnen Kanäle herausfiltern und diese separat darstellen, wie in den Abbildungen 2.12 bis 2.14 gezeigt. Jedoch muss darauf geachtet werden, dass hierbei die Farbsättigung S sowie die Farbhelligkeit V ebenfalls in einem Wertebereich von 0-255 angezeigt werden, anstatt von 0-1. [Open4]

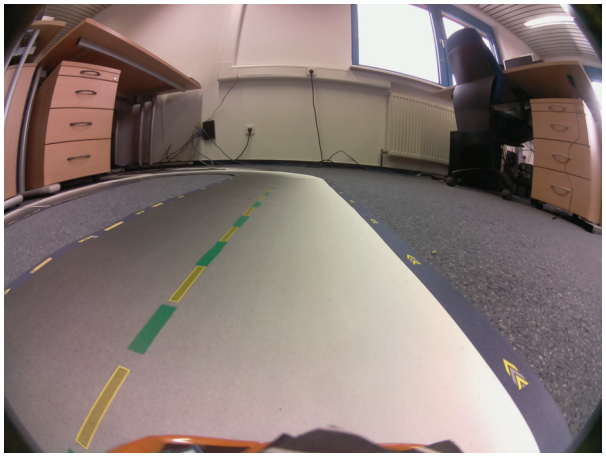

Abb. 2.10: RGB-Ursprungsbild

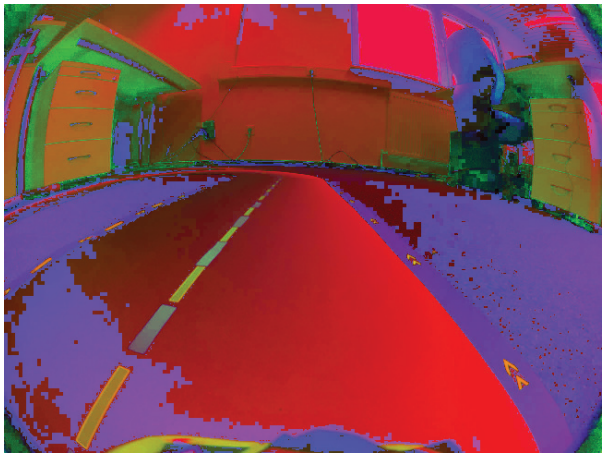

Abb. 2.11: HSV-Bild

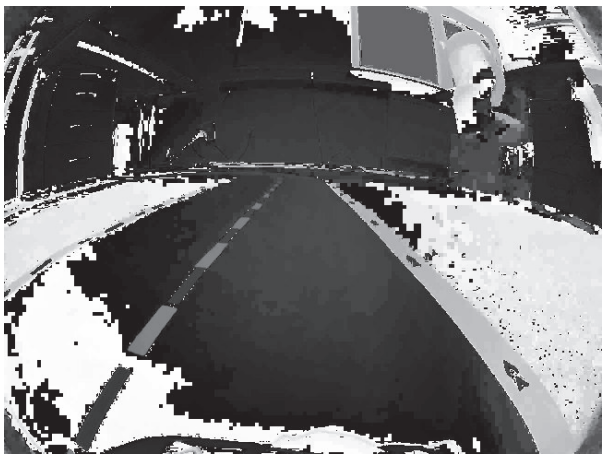

Abb. 2.12: H-Kanal

Abb. 2.13: S-Kanal

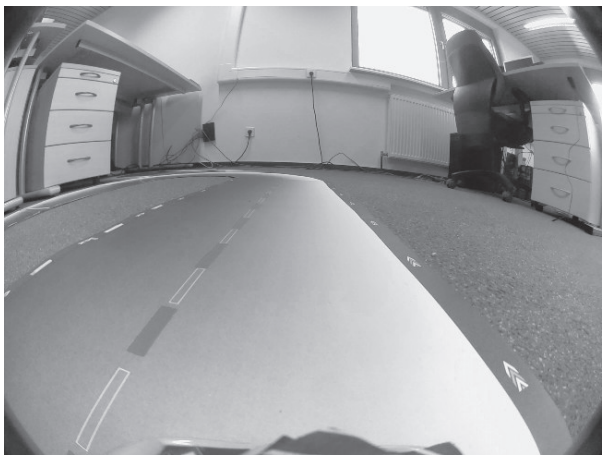

Abb. 2.14: V-Kanal

2.1.4 Konvertierung RGB → Grauwertbild

Um ein RGB-Bild in ein Grauwertbild umzuwandeln, muss zunächst für jedes RGB-Pixel der entsprechende Grauwert berechnet werden. Die einfachste Möglichkeit dafür wäre die Berechnung des arithmetischen Mittelwertes aus den drei Werten R, G und B. Dieser Mittelwert ist jedoch aufgrund des subjektiven Farbempfindens des Menschen nicht optimal. Aus diesem Grund müssen die RGB-Werte entsprechend ihrer Empfindung unterschiedlich gewichtet werden, um eine gleichwertige Intensität der einzelnen Farben zu erreichen.

$$\text{Grauwert} = w_R \cdot R + w_G \cdot G + w_B \cdot B \tag{2.7}$$

Für die Gewichtung existieren verschiedene Werte. Beispielwerte von OpenCV:

$$\text{Grauwert} = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \tag{2.8}$$

Hat ein Pixel zum Beispiel eine Farbe mit den RGB-Werten (100, 200, 150), so gilt für den Grauwert dieses Pixels:

$$\text{Grauwert} = 0.299 \cdot 100 + 0.587 \cdot 200 + 0.114 \cdot 150 = 164{,}4 \tag{2.9}$$

Durch das Einsetzen des äquivalenten Grauwertes für R, G und B gilt also:

$$R = G = B = 164{,}4. \tag{2.10}$$

Da nur ganze Zahlen möglich sind, hat das graue Pixel nun die Werte (164, 164, 164). Angewandt auf alle Pixel in einem Bild folgt daraus, dass das Bild „unbunt“, also ein Grauwertbild, ist.

Eine weitere Möglichkeit bieten Helligkeitsfunktionen anderer Farbsysteme, wie beispielsweise der V-Wert des HSV-Systems, der als Intensitätswert/Grauwert verwendet werden kann. [Burg15]

In den Abbildungen 2.15 und 2.16 ist der Vergleich des Grauwertbildes von OpenCV mit seinem Ursprungsbild zu sehen. Erzeugt wurde dies mit der Funktion `cvtColor(image, gray_image, COLOR_BGR2GRAY)`.

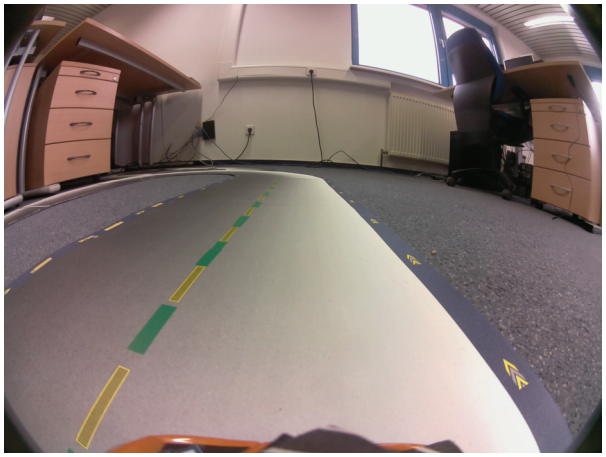

Abb. 2.15: RGB-Ursprungsbild

Abb. 2.16: Grauwertbild

2.1.5 Binarisierung

Unter einer Binarisierung versteht man das Erzeugen eines Binärbildes aus einem vorhandenen Graustufenbild. Es handelt sich dabei um ein Schwellenwertverfahren, bei dem ein bestimmter Schwellenwert vorgegeben wird. Nun wird jedes Pixel mit diesem Wert verglichen. Ist der Grauwert des Pixels kleiner oder gleich dem Schwellenwert, so wird er auf 0 gesetzt. Ist dies nicht der Fall, so wird er stattdessen auf 1 gesetzt. Dadurch entsteht ein Bild, dass lediglich aus zwei unterschiedlich farbigen Pixeln mit den Werten (0, 0, 0) und (1, 1, 1) besteht und aus diesem Grund Binärbild genannt wird. Da die Grauwerte 0 und 1 jedoch für das menschliche Auge nicht zu unterscheiden sind, wird statt 1 oft der Wert 255 genommen, dadurch entsteht ein Schwarzweißbild oder auch ein sogenanntes Zweipegelbild.

$$\text{Neuer Pixelwert} = \begin{cases} 0 & \text{falls Grauwert} \leq \text{Schwellenwert}, \\ 255 & \text{sonst.} \end{cases} \tag{2.11}$$

Bei der Festlegung des Schwellenwertes ist darauf zu achten, dass er so gelegt wird, dass möglichst nur das gewünschte Objekt segmentiert wird und eventuelle Bildstörungen wie beispielsweise Rauschen herausgefiltert werden. Wie sehr die Wahl des Schwellenwertes das Aussehen des Binärbildes beeinflussen kann, ist in den Abbildungen 2.18 bis 2.21 zu sehen. [Nisc11]

Die Abbildungen wurden in OpenCV mit Hilfe der Funktion `threshold(input, output,`

`threshold, binary_max, type)` erstellt. Die Funktion wurde mit verschiedenen Schwellenwerten auf ein Graustufenbild angewendet. Der maximale Binärwert betrug 255 und als Umwandlungstyp wurde `CV_THRESH_BINARY` genutzt, das, wie bereits in Gleichung 2.11 gezeigt, allen Werten über dem Schwellenwert den maximalen Wert zuordnet. Somit erhalten wir ein wirkliches Schwarzweißbild. Der vollständige Aufruf (hier mit Schwellenwert 200) lautet:

```
threshold(gray_image, sw, 200, 255, CV_THRESH_BINARY);
```

Es ist ebenfalls möglich, anstatt eines statischen Schwellenwertes diesen dynamisch an die Bildgegebenheiten anzupassen. Dies verbessert das Ergebnis ungemein, vor allem bei Bildern mit unausgeglichenen Lichtverhältnissen, wie in Abbildung 2.22 zu sehen. Auch hierfür wird von OpenCV eine passende Funktion `adaptiveThreshold(input, output, binary_max, method, type, blocksize, const)` zur Verfügung gestellt.

Im Gegensatz zur Threshold-Funktion übergibt man anstatt eines Schwellenwertes die Methode, mit der der Schwellenwert an die jeweiligen Gegebenheiten angepasst wird, sowie die Blockgröße, also die Anzahl der Pixel, auf die die Methode jeweils angewendet werden soll. Zusätzlich wird eine Konstante übergeben, die vom jeweiligen Mittelwert abgezogen wird, um unerwünschte Pixel, wie beispielsweise Bildrauschen, zu entfernen.

OpenCV bietet zwei Methoden zur Auswahl an: die Mean-Methode `ADAPTIVE_THRESH_MEAN_C`, die den Mittelwert jedes Blockes als Schwellenwert nutzt, und die Gauss-Methode `ADAPTIVE_THRESH_GAUSSIAN_C`, die eine gewichtete Summe als Schwellenwert nimmt. Für die Abbildung 2.22 wurde die Mean-Methode mit Blockgröße Drei und Konstante Sieben genutzt. Diese Werte wurden durch Ausprobieren ermittelt und lieferten in diesem Fall das scheinbar beste Ergebnis. Der vollständige Aufruf lautet:

```
adaptiveThreshold(gray_image, sw, 255, ADAPTIVE_THRESH_MEAN_C, THRESH_
BINARY, 3, 7);
```

Allerdings erhält man diese Verbesserung nur durch eine um einiges höhere Rechenleistung, da im schlechtesten Fall der Schwellenwert jedes einzelnen Pixels angepasst werden muss. [Open4]

Abb. 2.17: Graustufenbild Straße

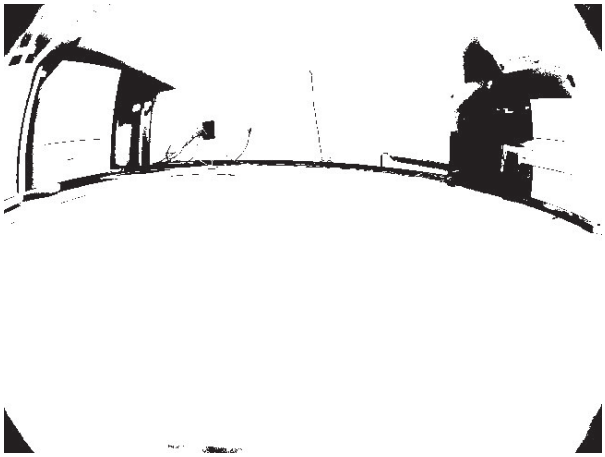

Abb. 2.18: Binärbild mit Threshold 50

Abb. 2.19: Binärbild mit Threshold 100

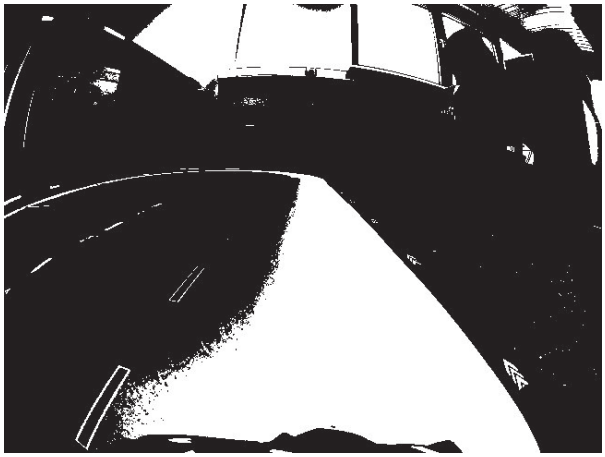

Abb. 2.20: Binärbild mit Threshold 150

Abb. 2.21: Binärbild mit Threshold 200

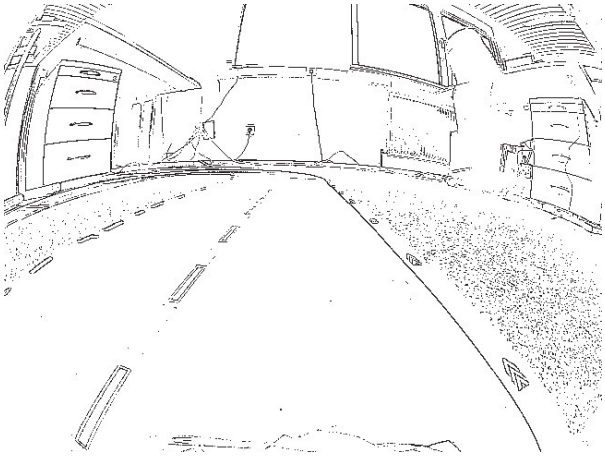

Abb. 2.22: Adaptiver Threshold