ulm university universität
uulm

Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
Institut für Mess-, Regel- und Mikrotechnik

Erkennung von dynamischen Objekten in Stereobilddaten

Masterarbeit

von

Daniel Jung

31.10.2018

Betreuer: M. Sc. Martin Herrmann

1. Prüfer: Prof. Dr.-Ing. Klaus Dietmayer
2. Prüfer: Prof. Dr. Frank Steiper

Hiermit versichere ich, dass ich die vorliegende Arbeit mit dem Titel

Erkennung von dynamischen Objekten in Stereobilddaten

bis auf die offizielle Betreuung selbstständig und ohne fremde Hilfe angefertigt habe und die benutzten Quellen und Hilfsmittel vollständig angegeben sind. Aus fremden Quellen direkt oder indirekt übernommene Gedanken sind jeweils unter Angabe der Quelle als solche kenntlich gemacht.
Ich erkläre außerdem, dass die vorliegende Arbeit entsprechend den Grundsätzen guten wissenschaftlichen Arbeitens gemäß der „Satzung der Universität Ulm zur Sicherung guter wissenschaftlicher Praxis“ erstellt wurde.

Ulm, den 31.10.2018

Daniel Jung

Inhaltsverzeichnis

1 Bildverarbeitung

1.1 Boxfilter

Boxfilter Faltung

$$\widetilde{I}(i,j) = I * f = \sum_{\alpha=-m}^{m} \sum_{\beta=-n}^{n} I(i-\alpha, j-\beta) \cdot f(\alpha,\beta)$$

Relevant:

$$\sum_{\alpha=-m}^{m} \sum_{\beta=-n}^{n} f(\alpha,\beta) \stackrel{!}{=} 1$$

Boxfilter - Mittelwertfilter

$$f = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\widetilde{I}(i,j) = \frac{1}{9} \sum_{\alpha=-1}^{1} \sum_{\beta=-1}^{1} I(i-\alpha, j-\beta)$$

Veranschaulicht wird dieses Vorgehen in 1.2. Im linken Abschnitt ist das Originalbild mit einem größeren Objekt und einigen Störpixeln zu sehen. Im rechten Abschnitt ist die gefilterte Version zu sehen: Die einzelnen Störpixel im oberen Abschnitt des Bildes sind deutlich abgeschwächt. Die Grenzen des Objektes in der rechten, unteren Ecke sind jedoch ebenso verschwommen. Bei mehrfacher Anwendung eines Mittelwertfilters konvergieren alle Pixelwerte gegen den Mittelwert der Pixel des gesamten Bildes. Der Mittelwertfilter ist somit ein Tiefpassfilter und sorgt anschaulich dafür, dass Bilder verschwimmen.

Gaußfilter:

$$N(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1.1}$$

Die Variable σ ergibt sich aus der Standardnormalverteilung mit $\mu = 0$. Damit ergeben sich Filterkerne wie

256	256	256	256	256	256	256	256	256	256
256	256	60	256	256	256	256	256	120	256
256	120	256	256	256	60	256	256	256	256
256	256	256	256	256	256	256	256	256	256
256	256	256	256	256	256	60	256	60	256
256	256	256	256	256	120	120	120	120	256
256	120	256	256	256	120	0	0	120	60
256	256	256	256	60	120	0	0	120	256
256	256	256	256	256	120	120	120	120	256
256	256	256	256	256	256	60	256	256	256

256	234	234	234	256	256	256	240	240	240
240	219	219	234	234	234	234	240	240	240
240	219	219	234	234	234	234	240	240	240
240	240	240	256	234	212	212	212	234	234
256	256	256	256	240	204	189	167	204	204
240	240	240	256	226	160	117	95	139	182
240	240	240	234	189	117	67	67	117	189
240	240	240	234	189	117	67	67	117	189
256	256	256	234	204	139	117	117	182	232
256	256	256	256	240	204	189	189	232	240

Abbildung 1.1: Beispiel für 3x3 Boxfilter

$$G_{3x3} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad G_{5x5} = \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

1.2 Kantendetektion

Sobelfilter zur Kantendetektion: Analogie zentrale Differenz:

$$f'(x) = \frac{f(x+\delta x) - f(x-\delta x)}{2\delta x}$$

$$S_x = \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad S_y = \frac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$S_u = \frac{1}{8}\begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix} \qquad S_v = \frac{1}{8}\begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

Canny Algorithmus: Berechne Kantenbild $D(x, y)$ und Gradientenrichtung $\phi(x, y)$

$$D(x,y) = \sqrt{g_x(x,y)^2 + g_y(x,y)^2}$$
$$\phi(x,y) = \tan^{-1}\left(\frac{g_y(x,y)}{g_x(x,y)}\right)$$

Wegen 8er Nachbarschaft: Runden auf 45° [Wag06]

1.3 Interpolation

Die Interpolation beschreibt die Suche nach einer Funktion, welche vorgebene Funktionswerte f_i an Stützstellen (x_i) am Besten approximiert. Im Falle der Bildbearbeitung lässt sich die Funktion beschreiben als $f : \mathbb{R}^2 \mapsto \mathbb{R}, f(x_n) = f_n, n = 1...4$. Für die Bildverarbeitung bilden die Stützstellen x_i im Allgemeinen ein Rechteck in dessen Inneren interpoliert wird, was auch bei der Vorstellung der nachfolgenden Verfahren der Fall ist.

Nearest-Neighbor-Interpolation

Wie der Name suggeriert wird bei diesem Verfahren der Wert als Ergebnis der Interpolation zurückgegeben, dessen Stützstelle den kleinsten Abstand zum angefragten Punkt hat. Als Norm für den Abstand wird hierbei häufig die Euklidische Norm herangezogen, generell sind jedoch auch andere Arten nutzbar.

$$f : \mathbb{R}^2 \mapsto \mathbb{R}, f(x) = x_k, k = \underset{l \in 1,\ldots,4}{arg\,min}(||x - x_l||)$$

Bilineare Interpolation

Die bilineare Interpolation nutzt eine lineare Ansatzfunktion um Werte zwischen den Stützstellen zu berechnen. Im Gegensatz zur Nearest-Neighbor-Interpolation ist es hier vor der Auswertung der Funktion notwendig einige Parameter zu berechnen.

$$\begin{aligned} f : \mathbb{R}^2 \mapsto \mathbb{R}, f(x,y) &= \sum_{i=0}^{1}\sum_{j=0}^{1} a_{ij}x^i y^j \\ &= a_{00} + a_{10}x + a_{01}y + a_{11}xy \end{aligned}$$

Durch Lösen eines linearen Gleichungssystems können die Parameter a_{ij} für die bilineare Interpolation berechnet werden. Das Gleichungssystem ergibt sich aus $f(x_i, y_i) = f_i, i = 1...4$, welches die Funktionsparameter und Funktionswerte an den Stützstellen sind.

$$\begin{pmatrix} 1 & x_0 & y_0 & x_0y_0 \\ 1 & x_1 & y_1 & x_1y_1 \\ 1 & x_2 & y_2 & x_2y_2 \\ 1 & x_3 & y_3 & x_3y_3 \end{pmatrix} \begin{pmatrix} a_{00} \\ a_{10} \\ a_{01} \\ a_{11} \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{pmatrix}$$

Bikubische Interpolation

Die bikubische Interpolation nutzt statt einer linearen eine kubische Ansatzfunktion. Die Funktion hat dann die folgende Form:

$$f : \mathbb{R}^2 \mapsto \mathbb{R}, f(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$$

Die Parameter a_{ij} werden wie bei der linearen Interpolation durch ein lineares Gleichungssystem berechnet. Zusätzlich zu den Funktionswerten f_i an den Stützstellen (x_i, y_i) müssen hier auch die Werte der Ableitungen f_x, f_y, f_{xy} bekannt sein um das Gleichungssystem lösen zu können. Dieses hat resultierend aus der kubischen Ansatzfunktion eine Größe von 16×16.

In beiden Fällen werden die Parameter a_{ij} analytisch berechnet um nicht zur Laufzeit eine Reihe von Gleichunssystemen Lösen zu müssen. Eine sinnvolle Vorverarbeitung projeziert zudem die Stützstellen $(x_i, y_i), i = 1..4$ auf die Eckpunkte des Einheitsquadrats. Dies hat zur Folge, dass sich die Funktion $f : \mathbb{R}^2 \mapsto \mathbb{R}$ auf $f : [0, 1]^2 \mapsto \mathbb{R}$ reduziert und die Rechenzeit deutlich reduziert werden kann, da sich das Gleichungssystem zur Berechnung vereinfachen lässt.

1.4 Morphologische Operationen

Morphologische Operationen werden auf Binärbildern angewendet um bestimmte Strukturen zu konkretisieren. Binärbilder zeichnen sich dadurch aus, dass die einzelnen Pixel nur die Werte 0 oder 1 annehmen können. Für die morphologischen Operationen wird, ähnlich wie beim Boxfilter, eine Maske über das Bild bewegt, anhand derer ein Ergebnis für ein bestimmtes Pixel bestimmt wird. Diese Maske ist ebenfalls binär und wird als Strukturelement bezeichnet. Die übereinanderliegenden Pixel des Bildes und des Strukturelements werden logisch miteinander verknüpft. Je nach angewendeter Operationen wird so ein Rückschluss auf das zu bearbeitende Pixel gezogen.

Zu den Standardoperationen gehören die Erosion $A \ominus B = \{p \in \mathbb{Z}^2 | B_p \subseteq A\}$ sowie die Dilation $A \oplus B = \{p \in \mathbb{Z}^2 | B_p \cap A \neq \emptyset\}$ des Bildes A mit dem Strukturelement B. Allgemein wird die Fläche von Objekten durch die Erosion verkleinert und durch die Dilation vergrößert.

Aufbauend auf diesen beiden Operationen sind die Methoden des Openings und Closings. Das Opening eines Bildes $A \circ B$ beschreibt die Anwendung einer Erosion auf das Bild A mit anschließender Dilation des Ergebnisses: $A \circ B = (A \ominus B) \oplus B$. Das Closing $A \bullet B$ wendet zuerst eine Dilation gefolgt von einer Erosion auf das Bild an: $A \bullet B = (A \oplus B) \ominus B$. Das Opening wird unter anderem eingesetzt um Rauschen im Bild zu entfernen oder die Ränder von Objekten hervorzuheben und Verbindungen zu kappen. Das Closing hat den Zweck kleine Objekte hervorzuheben und Lücken in größeren Flächen zu schließen.

In Abbildung 1.2 sind die vier Operationen anhand einer Binärstruktur veranschaulicht. Das hierbei verwendete Strukturelemet beschreibt die sogenannte Achter-Nachbarschaft.

Vierer-Nachbarschaft

$$B = \begin{pmatrix} 0 & 1 & 0 \\ 1 & \otimes & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Achter-Nachbarschaft

$$B = \begin{pmatrix} 1 & 1 & 1 \\ 1 & \otimes & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Die Dilation füllt die Ränder des Objekts auf und schließt die Lücken im Inneren. Das Closing entsteht nun durch Erosion des Ergebnisses. Das Objekt im Bild hat seine Kontur nahezu vollständig erhalten wobei die Lücken im Inneren geschlossen sind. Die Erosion trägt Pixel am Rand des Objektes ab. So werden die Arme an der rechten Seite entfernt und auch die Lücken im Inneren werden größer. Durch die anschließende Dilation werden die Löcher teilweise wieder geschlossen, die Arme an der Seite bleiben jedoch verschwunden und das Objekt wird so in Richtung seines Flächenzentrums komprimiert.

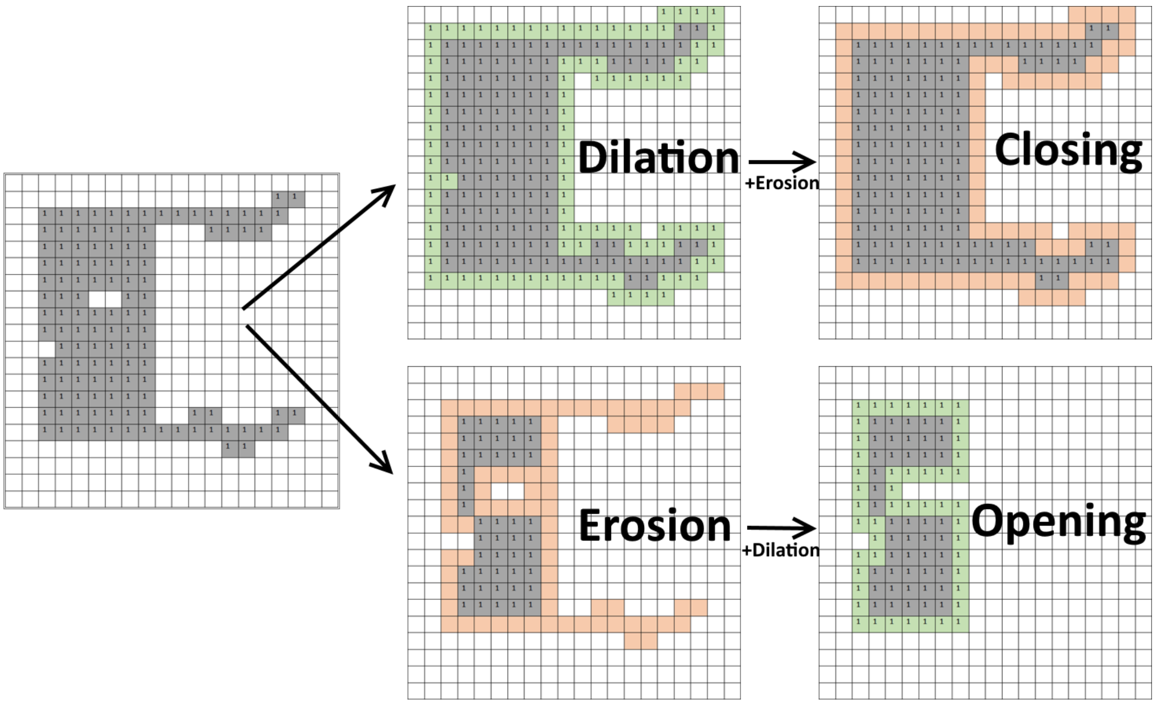

Abbildung 1.2: Morphologische Operationen - Dilation, Erosion, Opening und Closing

2 Kalibrierung

2.1 Single-Kamera-Kalibrierung

2.2 Stereokalibrierung

2.3 Rektifizierung

3 Background Subtraction

Background Subtraction bzw. Foreground Detection beschäftigen sich mit der Detektion von sich bewegenden Objekten in der Bildverarbeitung. Um Objekte durch ihre Bewegung erkennen zu können ist es notwendig, eine Folge von Bildern in ihrem zeitlichen Verlauf zu analysieren und so Änderungen feststellen zu können. Das Ziel der Background Subtraction ist es, die Dimension des Bildes zu reduzieren. In vielen Anwendungen, insbesondere im Bereich Straßenverkehr und Fahrerassistenz, sind nur dynamische Objekte relevant. Um nicht die knappe Rechenzeit auf statischen Hintergrund zu verschwenden wird so versucht die relevanten Bildausschnitte zu extrahieren und nur diese zu analysieren. Ergebnis der Background Subtraction ist im Allgemeinen eine binäre Maske $B \in \{0,1\}^{m \times n}$. Pixel welche dynamische Objekte enthalten haben dabei den Wert 1, alle anderen Pixel sind Hintergrund und erhalten den Wert 0. Auf diese Weise kann die Maske logisch auf das Originalbild $I \in \mathbb{R}^{m \times n}$ angewendet werden um den Vordergrund F zu erhalten:

$$F_{ij} = \begin{cases} I_{ij} & B_{ij} = 1 \\ 0 & B_{ij} = 0 \end{cases}$$

3.1 Frame Difference

Die Methoden die mit Mitteln der Frame Difference arbeiten bilden die Differenz aus dem zu analysierenden Bild $I^t \in \mathbb{R}^{m \times n}$ mit einem Referenzbild $R^t \in \mathbb{R}^{m \times n}$. Die so entstandene Differenz D^t wird anschließend mittels Threshold auf Änderungen überprüft:

$$\begin{aligned} D_{ij}^t &= |R_{ij}^t - I_{ij}^t| \\ B_{ij}^t &= D_{ij}^t > Threshold \end{aligned}$$

Das Referenzbild kann hierbei ein statisches, zuvor festgelegtes Bild sein oder es wird das zeitlich vorangegangene Bild als Referenz verwendet. Die Wahl des Schwellwerts bestimmt wie stark die Abweichung sein muss, damit Pixel als Vordergrund klassifiziert

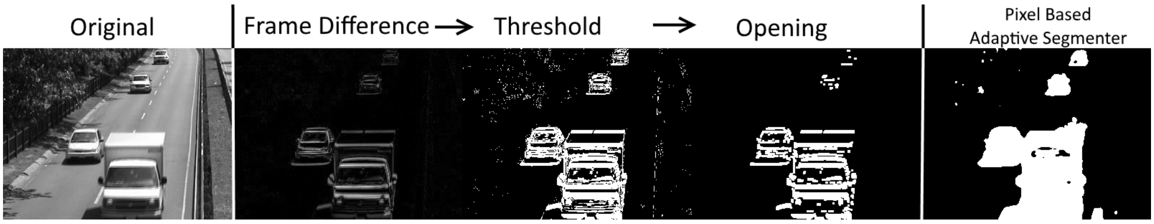

Abbildung 3.1: Background Subtraction - Original, Frame Difference, PBAS

werden. Bei zu hohem Schwellwert werden eventuell dynamische Pixel nicht als solche erkannt, bei zu geringem Schwellwert ist das Gegenteil der Fall und es kommt häufig zu Rauschen im Bild. Allgemein stellt das Rauschen beim Bilden der Frame Difference ein großes Problem dar. Um hierbei stabilere Ergebnisse zu erhalten wird in [XZD17] vorgeschlagen, statt nur zwei Bilder die Differenz aus je zwei aufeinander folgenden Bildern zu verwenden und die Ergebnisse logisch zu verknüpfen:

$$B_{ij}^t = \begin{cases} 1 & (D_{ij}^{t-1} > Threshold) \cap (D_{ij}^t > Threshold) = 1 \\ 0 & sonst \end{cases}$$

3.2 Weitere Methoden

Es existieren eine Reihe weiterer Verfahren, welche sich wesentlich an der Bildung der Frame Difference orientieren. Dazu zählt beispielsweise der Mittelwertfilter, welcher als Referenzbild den Mittelwert der letzten l Bilder berechnet:

$$R_{ij}^t = \frac{1}{l} \sum_{k=1}^{l} I_{ij}^{t-k}$$

Einen ähnlichen Ansatz verwendet das Gaußverfahren. Hierbei werden die letzten l Pixelwerte durch eine Gaußverteilung angenähert:

$$R_{ij} \tag{3.1}$$

4 Stereoberechnung

4.1 Lokale Verfahren

Minimierung der Energiefunktion

$$E(d) = \sum_{(x,y)} M(x, y, d(x, y)) \tag{4.1}$$

4.2 Globale Verfahren

Graph Cuts, Belief Propagation, Dynamic Programming Minimierung einer globalen Energiefunktion:

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d) \tag{4.2}$$

$$E_{data}(d) = \sum_{(x,y)} M(x, y, d(x, y)) \tag{4.3}$$

$$E_{smooth} = \sum_{(x,y)} p(d(x, y) - d(x + 1, y)) + p(d(x, y) - d(x, y + 1)) \tag{4.4}$$

[SSZ18]

Sum of Absolute Differences	$\sum_{u,v} \lvert I_1(u,v) - I_2(u+d,v) \rvert$
Sum of Squared Differences	$\sum_{u,v}(I_1(u,v) - I_2(u+d,v))^2$
Normalized Cross-Correlation	$\frac{\sum_{u,v}(I_1(u,v)-\bar{I}_1)\cdot(I_2(u+d,v)-\bar{I}_2)}{\sqrt{\sum_{u,v}(I_1(u,v)-\bar{I}_1)^2\cdot(I_2(u+d,v)-\bar{I}_2)^2}}$
Normalized Sum of Squared Differences	$\sum_{u,v}\left(\frac{(I_1(u,v)-\bar{I}_1)}{\sqrt{\sum_{u,v}(I_1(u,v)-\bar{I}_1)^2}} - \frac{(I_2(u,v)-\bar{I}_2)}{\sqrt{\sum_{u,v}(I_2(u,v)-\bar{I}_2)^2}}\right)^2$
Rank	$\sum_{u,v}(\hat{I}_1(u,v) - \hat{I}_2(u+d,v)$ $\hat{I}_k(u,v) \coloneqq \sum_{m,n} I_k(m,n) < I_k(u,v)$
Census	$\sum_{u,v} HAMMING(\hat{I}_1(u,v), \hat{I}_2(u+d,v))$ $\hat{I}_k(u,v) \coloneqq BITSTRING_{m,n}(I_k(m,n) < I_k(u,v))$

[Fus18]

Boxmatching

4.3 Kombinierte Verfahren

4.3.1 Semi-Global-Matching

4.3.2 Pyramidenverfahren

5 Segmentierung

5.1 Region Growing Algorithmus

5.2 DBScan Algorithmus

5.3 Kombinierte Segmentierung

6 Klassifikation

Die Klassifikation beschäftigt sich mit der Einordnung von Objekten zu bestimmten Klassen. Die Objekte werden dabei durch einen sogenannten Merkmalsvektor $x \in \mathbb{R}^n$ beschrieben, welcher definierte Eigenschaften des Objekts enthält. Die benutzten Merkmale können je nach Anwendungsfall unterschiedlich ausfallen. Im Bereich der Klassifikation von Verkehrsteilnehmern bieten sich geometrische Eigenschaften wie Ausmaße des Objekts, Umfang, Volumen oder Dichte an. Die zuzuordnende Klasse oder auch Label genannt ist im Allgemeinen eine ganze Zahl $y \in \mathbb{Z}$. Der Klassifikator f lässt sich somit beschreiben als:

$$f : x \in \mathbb{R}^n \mapsto y \in \mathbb{Z}$$

Er ordnet den Merkmalen eines Objekts eine Klasse zu. Die Form des Klassifikators kann dabei variieren und hängt vom Anwendungsfall ab. Für gewöhnlich muss ein Klassifikator trainiert werden. Unter dem Begriff Training versteht man in diesem Zusammenhang den Vorgang, die wählbaren Parameter der Klassifikationsfunktion so einzustellen, dass er auf einen bestimmten Trainingsdatensatz $(x_i|y_i)_{i=1..n}$ möglichst gute Daten liefert. Das Training kann als Optimierungsproblem dargestellt werden:

$$\min_{\alpha} \sum_{i=1}^{n} \omega(f_\alpha(x_i), y_i)$$

Die Funktion $\omega : \mathbb{Z}^2 \mapsto \mathbb{R}$ ist eine Fehlerfunktion welche testet, ob die zu testende Klasse mit der vom Klassifikator ausgewerteten Klasse übereinstimmt:

$$\omega(y_1, y_2) = \begin{cases} 0 & y_1 = y_2 \\ d(y_1, y_2) & y_1 \neq y_2, d > 0 \end{cases}$$

Durch Minimierung der Summe der Fehler kann so eine bestmögliche Approximation an die Trainingsdaten gefunden werden. Im Folgenden werden einige Klassifikatoins-verfahren vorgestellt.

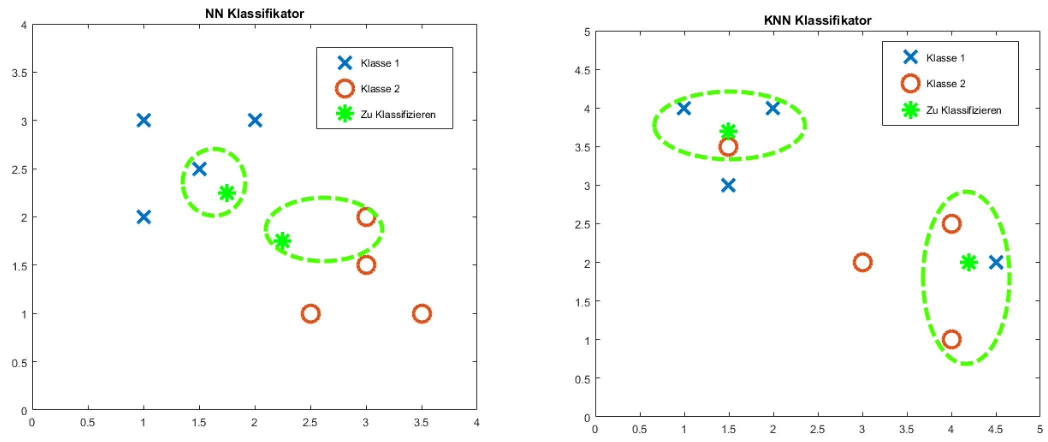

Abbildung 6.1: Klassifikation - Beispiel für NN- und KNN-Klassifikator

6.1 K-Nearest-Neighbor Klassifikation

Der Nearest-Neighbor-Klassifikator arbeitet ohne einen Optimiegunssprozess direkt auf den Trainingsdaten $(x_i|y_i)_{i=1..m} \in \mathbb{R}^n \times \mathbb{Z}$. Ein Objekt mit Merkmalen $b \in \mathbb{R}^n$ wird derjenigen Klasse zugeordnet, zu der der Abstand $d = \|b - x_i\|$ am geringsten ist:

$$f(b) = y_k, k = \underset{k \in \{1..m\}}{arg\,min}(||b - x_k||)$$

Die Norm $\| \cdot \|$ ist hierbei beliebig, häufig wird jedoch die euklidische Norm $\| \cdot \|_2$ verwendet.

In Abbildung 6.1 ist ein Beispiel für eine solche Klassifikation zu sehen. Auf der linken Seite wird ein einfacher Nearest-Neighbor-Klassifikator angewendet. Auf der rechten Seite ist jedoch sichtbar, dass das Rauschen in den Daten zu einer Fehlklassifikation führen würde. Aus diesem Grund wurde der Klassifikator erweitert zum K-Nearest-Neighbor-Klassifikator. Dieser bezieht nicht nur das Trainingsdatum mit dem geringsten Abstand ein, sondern sucht die k nächsten Nachbarn. Das Ergebnis der Klassifikation ist dann diejenige Klasse, welche am häufigsten in dieser Ergebnismenge vorkommt. In der Grafik ist der KNN-Klassifikator für den fall $k = 3$ dargestellt.

Der KNN-Klassifikator finden häufig zu Testzwecken Anwendung, da er einfach zu implementieren ist und stabile Resultate liefert. Da er kein Training benötigt ist jedoch der Aufwand zur Evaluation hoch und der Algorithmus ist damit nicht sehr effizient. Zu produktiven Zwecken in Echtzeitsystemen wird er daher nur selten verwendet.

6.2 Logistische Regression

Binäre Klassifikation

$$h_\beta(x) = \frac{1}{1 + e^{-(\beta_0 + \sum_{k=1}^{m} \beta_k x_k)}}$$
$$f_\beta(x) = \begin{cases} 1 & h_\beta(x) \geq 0.5 \\ 0 & h_\beta(x) < 0.5 \end{cases}$$

6.3 Support Vector Machine

Die Klassifikation durch die sogenannte Support Vector Machine wird vorallem für binäre Klassifikationsprobleme verwendet. Ziel der Methode ist es, eine Hyperebene zu finden, welche zwei linear trennbare Mengen voneinander trennt. Der Klassifikator hat die Form

$$f_{w,b}(x) = \begin{cases} 1 & w^\intercal x + b > 0 \\ 0 & sonst \end{cases}$$

Der Vektor $w \in \mathbb{R}^n$ ist die Normale der Ebene, das Skalar $b \in \mathbb{R}$ ist der sogenannte Bias, also die Verschiebung. Diese beiden Parameter müssen durch das Training des Klassifikators berechnet werden. Hierzu wird das Optimierungsproblem anhand der Trainingsdaten $(x_i|y_i)_{i=1..m} \in \mathbb{R}^n \times \mathbb{Z}$ formuliert:

$$\min \tfrac{1}{2}||w||_2^2$$
$$y_i(w^\intercal x_i + b) \geq 1, \forall (x_i|y_i)$$

Die Nebenbedingung fordert hierbei, dass beide Klassen tatsächlich linear trennbar sind. Da dies im Allgemeinen nicht der Fall ist wird eine sogenannte Schlupfvariable eingefügt, welche die Verletzung der Nebenbedingung erlaubt:

$$\min \tfrac{1}{2}||w||_2^2 + C \textstyle\sum_{i=1}^{m} \xi_i$$
$$y_i(w^\intercal x_i + b) \geq 1 - \xi_i, \forall (x_i|y_i)$$

In Abbildung 6.2 ist eine lineare Trennebene zu sehen. Die Methode der Support Vector Machine ermöglicht neben einer linearen auch eine nichtlineare Trennebene, was in Abbildung 6.3 zu sehen ist. Das Optimierungsproblem gestaltet sich dabei deutlich komplexer, weshalb in der Arbeit auf die lineare Variante zurückgegriffen wird.

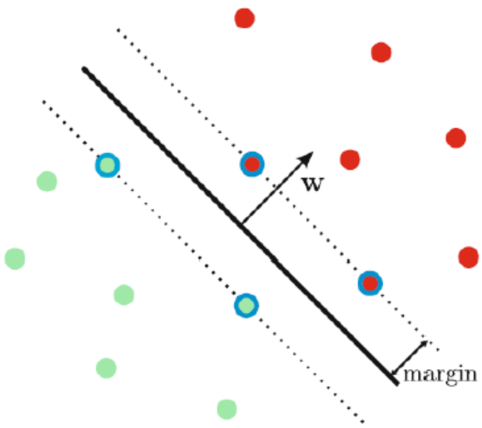

Abbildung 6.2: SVM linear

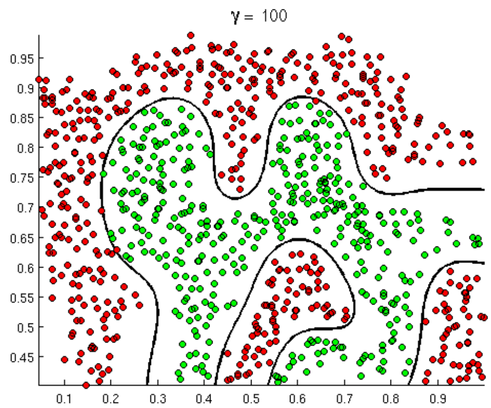

Abbildung 6.3: SVM nichtlinear

7 Tools und Datensatze

7.1 Background Subtraction Datasets

ChangeDetection.NET

Der changedetection Datensatz ist eine im Jahr 2012 Ö entstandene Sammlung von Videos, welche unterschiedliche Szenarien und Kameraeigenschaften darstellen. Die Motivation der Urheber ist, dass kein universeller Algorithmus existiert, welcher alle Szenarien angemessen gut auswerten kann. Im Datensatz vorhandene Szenarien sind:

Datensatz 2012:

- Dynamischer Hintergrund
- Wackelnde Kamera
- Unregelmäßge Objektbewegungen
- Schatten
- Wärmebildkameras

Datensatz 2014:

- Wettereinfluss
- Niedrige Bildrate
- Nachtaufnahmen
- Überwachungskameras
- Luftturbulenzen durch thermische Störungen

Ebenso mitgeliefert werden zu den Videos gehörige Ground-Truth-Daten, welche die tatsächliche Bewegung in den einzelnen Bildern beschreiben. Diese Daten wurden mitunter manuell erstellt. Als Vordergrund gelten unter anderem Regionen, welche nicht Teil des statischen Hintergrunds sind und bei denen es sich insbesondere um Objekte wie Personen oder Fahrzeuge handelt. Auch Objekte, die eine kurze Standzeit im Bild haben, sollen nicht als Hintergrund klassifiziert werden. Dagegen werden Störungen wie Lichtreflexion, Schatten oder Turbulenzen in der Luft nicht als Vordergrnud detektiert.

Background Model Challenge

Der Datensatz der Background Model Challenge (BMC) enthält eine Sammlung an Videos, welche ebenfalls zur Evaluation von Background-Detection-Algorithmen benutzt werden können. Hierzu ist eine Reihe von unterschiedlichen Videosqeuenzen vorhanden, welche verschiedene Szenen abbilden. Der Unterschied zum oben beschriebenen changedetection-Datensatz ist, dass auch synthetische Videos zur Verfügung stehen. Dies sind Videos, welche durch das Rendern einer aufgebauten 3D-Szene am Computer erstellt wurden. Der Vorteil dieses Vorgehens ist, dass die zugehörigen Ground-Truth-Daten direkt mit erzeugt werden können und keine aufwändige Nachbearbeitung nötig ist. Dafür werden jedoch in der Praxis auftretende Effekte vernachlässigt, wie beispielsweise eine sich bewegende Kamera oder Turbulenzen. Die synthetischen Videos sind daher nur bedingt nutzbar.

7.2 Stereo Datensätze

Kitti Datensatz

Der Kitti Datensatz ist eine mfangreiche Sammlung von Datesätzen, welche im Zusammenhang mit autonomem Fahren verwendet werden. So sind unter anderem Datensätze vorhanden für Stereovision, Szenenfluss oder Objekt-Tracking vorhanden. Die Videodaten sowie zugehörige Ground-Truth-Daten wurden mit Hilfe eines Sensorfahrzeugs erzeugt, welches mit Lidar und GPS ausgerüstet ist. Die Sequenzen wurden sowohl innerhalb wie außerhalb von Städten aufgenommen und enthalten eine ausreichende Anzahl an Fahrzeugen und Fußgängern. Zusätzlich zu den Ground-Truth-Daten sind auch Benchmarks und Evaluationsalgorithmen vorhanden um angewendete Algorithmen vergleichbar zu machen. Die Seite bietet auch die Möglichkeit der Einreichung von Algorithmen und listet diese geordnet nach ihrer Effektivität auf.

Der Kitti Datensatz hat besondere Relevanz für diese Arbeit, da sich die Szenerie ausschließlich auf den Straßenverkehr konzentriert. Nachteilig ist, dass pro Szene nur zwei Bilder vorhanden sind und somit keine kontinuierliche Verkehrsszene abgebildet wird.

Stereo Ego-Motion Dataset

Der Stereo Ego-Motion Datensatz enthält 494 hochauflösende Videos von vier Objekten: Eine Auto, eine Katze, ein Stuhl und ein Hund. Im Video werden diese Objekte mit der Kamera umkreist um ein Rundumbild zu erzeugen. Dazu sind ebenfalls Ground-Truth-Daten vorhanden. Mit diesem Vorgehen können komplette 3D-Meshes anhand der Stereodaten erzeugt werden. Für das einfache Benchmarking von Stereodaten ist der Datensatz auch zu verwenden, wobei die Eigenkamerabewegung nur bedingt zum Thema der Arbeit passt.

Middlebury Stereo Dataset

Der Middlebury Datensatz enthält fünd verschiedene Datensätze mit insgesamt 71 unterschiedlichem Motiven. Enthalten sind Ground-Truth-Daten sowie ein Softwaretool zur Evaluation. Auch das Einreichen erzielter Ergebnisse ist möglich. Besonders interessant an diesem Datensatz ist, dass die Objekte unter variablen Lichtverhältnissen aufgenommen wurden. So stehen zu jedem Objekt Bilder mit vier unterschiedlichen Beleuchtungen. Dies stellt den Stereoalgorithmus zusätzlich vor Herausforderungen, da auch in der Realität und vorallem im Straßenverkehr Licht eine bedeutende Rolle spielt.

7.3 Tools

OpenCV

OpenCV ist eine offene Softwarebibliothek für die Sprache C++, die auf den Bereich der maschinellen Bildverarbeitung spezialisiert ist. In OpenCV sind mehr als 2500 Algorithmen vorhanden. Am Projekt beteiligen sich laut eigenen Angaben mehr als 47000 Entwickler und die geschätze Downloadzahl liegt bei ca. 14 Millionen [tea18]. Einen Überblick über die Kernfunktionalität der Bibliothek gibt folgende Liste [con18; Cor18]:

- Bild- und Videoverarbeitung und Darstellung
- Objekterkennung und Extraktion von Objektmerkmalen
- Kamerakalibrierung und Stereovision
- Maschinelles Lernen und Clusteringverfahren
- Eigenbewegungsschätzung
- Gesichts- und Gestenerkennung

OpenCV ist sehr effizient und wird produktiv in der Industrie eingesetzt. So setzen sowohl offene Softwareprojekte oder Startups auf OpenCV, aber auch größere Firmen wie Google, Microsoft oder Yahoo sind vertreten. Die Einsatzgebiete reichen dabei von der Verkehrsüberwachung über Roboternavigation bis hin zur Gesichtserkennung in Japan.

OpenCV kann dabei nicht nur in C++ zum Einsatz kommen: Es sind auch Schnittstellen zu Python, Java und Matlab verfügbar, wobei alle gängigen Betriebssysteme unterstützt werden.

CUDA

Point Cloud Library

Javascript Object Notation

8 Evaluation Background Subtraction

9 Evaluation Stereoberechnung

10 Fahrzeugdetektion

10.1 Programmablauf

10.2 3D-Grafik

10.3 Evaluation

10.3.1 Fehlernormen

$d_C(x, y)$ berechnete Disparity Map $d_T(x, y)$ Ground Truth Map

Bad Pixel Percentage

$$P = \frac{1}{N} \sum_{(x,y)} (|d_C(x, y) - d_T(x, y)| > \delta_d) \tag{10.1}$$

δ_d Fehlertoleranz

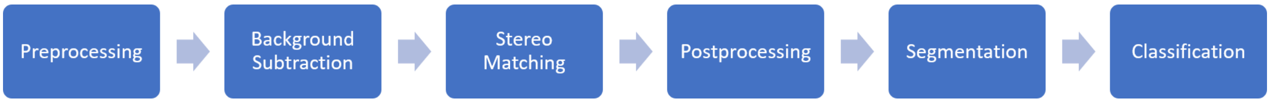

Abbildung 10.1: Pipeline zur Detektion

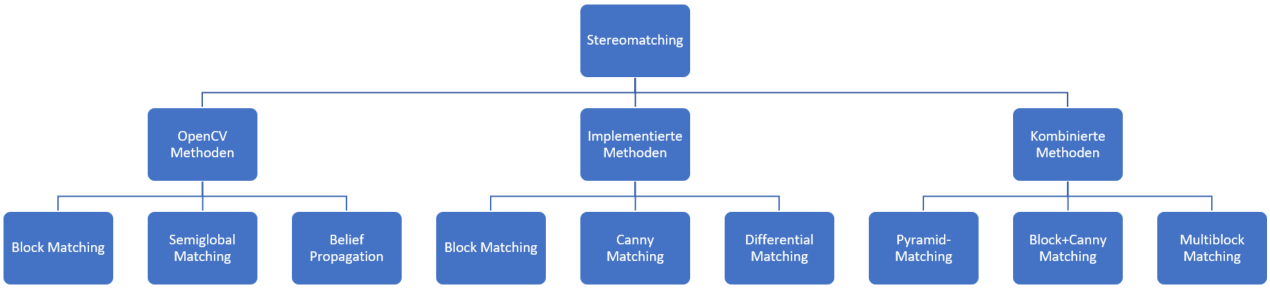

Abbildung 10.2: Pipeline zur Detektion

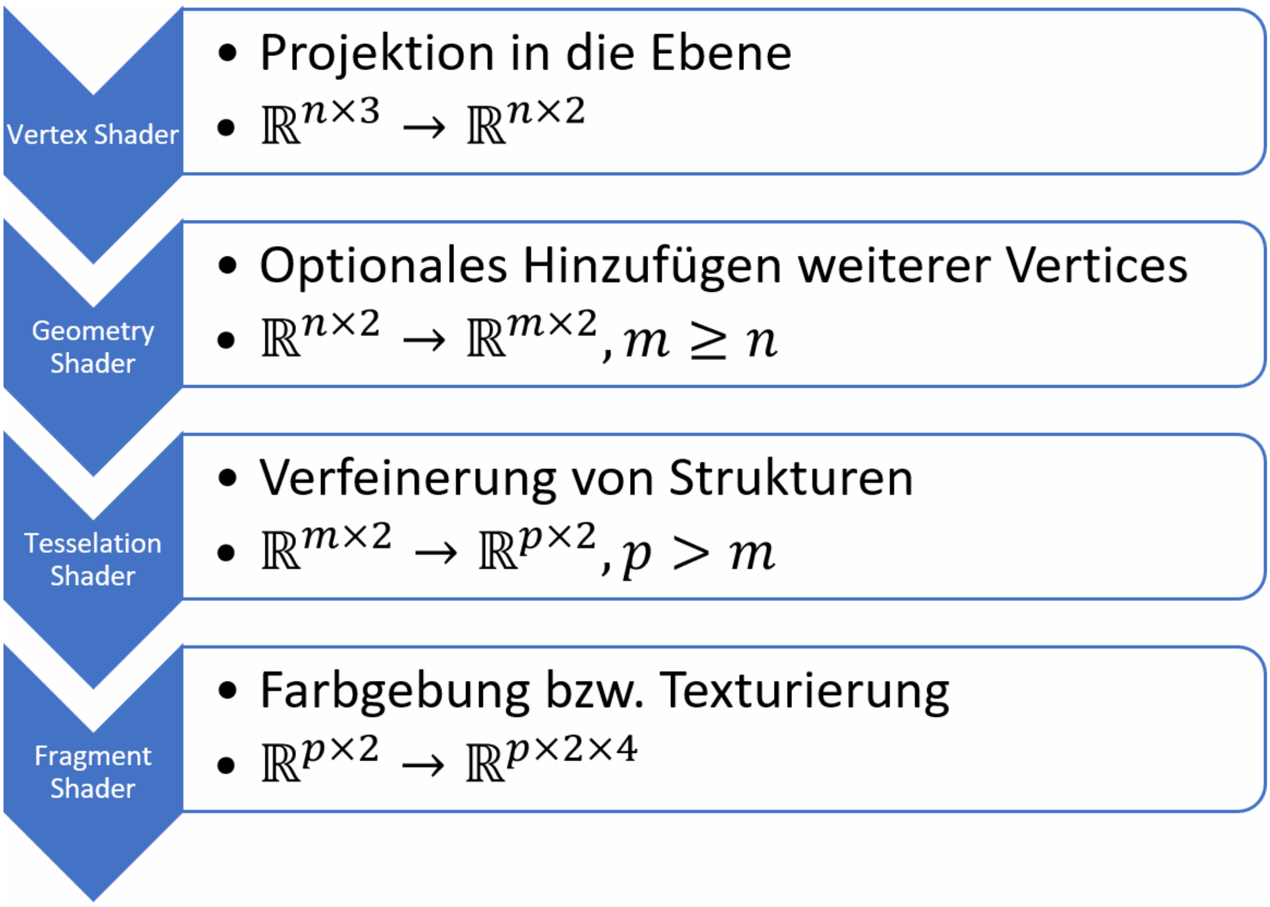

Abbildung 10.3: Pipeline zur Detektion

Root Mean Squared Error

$$E = \left(\frac{1}{N} \sum_{(x,y)} |d_C(x,y) - d_T(x,y)|^2 \right)^{\frac{1}{2}} \tag{10.2}$$

Literaturverzeichnis

[con18] contributors, Wikipedia: *Wikipedia - OpenCV*. 2018. URL: https://de.wikipedia.org/wiki/OpenCV.

[Cor18] Corporation, NVIDIA: *NVidia - What is OpenCV?* 2018. URL: https://developer.nvidia.com/opencv.

[Fus18] Fusiello, Andrea: *Stereo Matching: an Overview*. 2018. URL: http://www.diegm.uniud.it/fusiello/teaching/mvg/stereo.pdf.

[SSZ18] Scharstein, Daniel; Szeliski, Richard und Zabih, Ramin: *A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algrithms*. In: 2018.

[tea18] team, OpenCV: *OpenCV - About*. 2018. URL: https://opencv.org/about.html.

[Wag06] Wagner, Christoph: *Kantenextraktion - Klassische Verfahren*. 2006.

[XZD17] Xu, Z.; Zhang, D. und Du, L.: *Moving Object Detection Based on Improved Three Frame Difference and Background Subtraction*. In: *2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII)*, Seiten 79–82, 2017.