

Aufgabe 2

Gruppe 4

Jonas Eckhoff

Anton Jabs

Florian Brach

Felix Kieckhäfer

10. Dezember 2018

Hier kann noch eine neue Zusammenfassung rein.

1 Die Bilderkennung

Das Ziel der Bilderkennung ist es, zwei Punkte zu identifizieren, von denen man den Abstand in den Weltkoordinaten kennt. Anschließend kann man dann durch eine Koordinatentransformation von den Bildkoordinaten zu den Weltkoordinaten die Position der Kamera in der Welt in Relation zum bekannten Objekt bestimmen. Für dieses Objekt haben wir ein rotes Rechteck gewählt.

1.1 Bildvorverarbeitung

Zuerst soll das Bild so angepasst werden, dass das rote Rechteck einfacher identifizierbar wird. Normalerweise würde man damit beginnen das Bild erstmal vorzuglätten (zum Beispiel mit einem Gauss-Filter) um Rauschen zu minimieren. Dies hat sich aber aus Performancegründen als nicht so hilfreich herausgestellt. Stattdessen kann hier gut die Eigenschaft, dass das Rechteck rot ist, verwendet werden. Wenn man sich die RGB-Kanäle (a, b und c) anschaut, fällt auf, dass das rote Rechteck sehr hell im rot Kanal ist, allerdings fast gar nicht in dem blauen und grünen Kanal sichtbar ist. Alle anderen Farben haben auch einen Blau- oder Grünanteil. Aus diesem Grund wird im ersten Schritt ein Graustufenbild erzeugt, indem der blaue und grüne Kanal vom roten Kanal abgezogen werden.

(a) Rotkanal.png (b) Grünkanal.png (c) Blaukanal.png
(d) Rechteckbild.png (e) Rot-0.5(Grün+Blau) (f) Binärbild.png

In diesem Graustufenbild (e) sind nun hauptsächlich helle Rottöne sichtbar. Nun kann ein Binärbild (f) erstellt werden, indem alle Grautöne über einem Schwellenwert weiß dargestellt werden und alle anderen schwarz.

1.2 Rechteckserkennung


Nun sollen die Eckpunkte des Rechteckes gefunden werden. Im ersten Schritt wird ein Gitter über das Bild gelegt, das so fein gewählt wird, dass ein Gitterpunkt auf jeden Fall im Rechteck liegen muss. Dafür wird angenommen, dass das Rechteck eine Mindestgröße im Bild hat und nicht zu weit weg ist. Ob ein Gitterpunkt (x_1, y_1) nun wahrscheinlich in einem Rechteck liegt, wird mithilfe des Algorithmus 1 festgestellt.

Dies bestimmt den Punkt (x_1, y_2) und (x_1, y_3) . Nun lässt sich der Mittelpunkt $(x_1, y_m) = (x_1, \frac{y_2+y_3}{2})$ bestimmen. Nun wird der gleiche Algorithmus nochmal in x-Richtung durchgeführt mit Startwert (x_1, y_m) um x_l und x_r zu finden. Um leicht schräge Seiten auszugleichen wird nun der Algorithmus auf $(x_l + a, y_m)$ und $(x_r - a, y_m)$ angewandt, um die vier Rechtecksecken (x_{ol}, y_{ol}) , (x_{or}, y_{or}) , (x_{ul}, y_{ul}) und (x_{ur}, y_{ur}) zu finden.

Nun wird noch überprüft, ob die gefundenen Ecken ein sinnvolles Rechteck bilden. Dafür darf $y_{ol} - y_{ul}$ nicht mehr als 30% größer oder kleiner als $y_{or} - y_{ur}$ oder $x_r - x_l$ sein, da das Rechteck leicht nach hinten gekippt sein kann (vorausfahrendes Auto lenkt). Außerdem soll

Algorithmus 1 Pseudo-Code

```
1:  $(x, y) = (x_1, y_1)$ 
2: while  $I(x, y) = \text{white}$  do
3:    $y = y - 1$ 
4: end while
5:  $y_2 = y$ 
6:  $(x, y) = (x_1, y_1)$ 
7: while  $I(x, y) = \text{white}$  do
8:    $y = y + 1$ 
9: end while
10:  $y_3 = y$ 
```



rechteckgrafik.pdf

Abbildung 1: Beispiel-Rechteck

noch $\frac{y_m - y_{ol}}{y_m - y_{or}}$ ungefähr $\frac{y_{ul} - y_m}{y_{ur} - y_m}$ sein. Wenn diese Bedingungen eingehalten werden, handelt es sich wahrscheinlich um ein Rechteck.

Um möglichst schnell den richtigen Startpunkt (x_1, y_1) zu finden, wird spiralförmig um den Mittelpunkt des im letzten Bild gefundenen Rechtecks gesucht.

2 Umrechnung der Pixelkoordinaten in Weltkoordinaten

3 Abstandsregler

3.1 Simulink-Modell

Für den Abstandsregelung werden die zuvor beschriebenen Objekterkennung und die Matrixtransformation genutzt. Aus dem RGB-Eingangsvideosignal wird der Rotkanal durch Subtraktion der beiden anderen Kanäle extrahiert und anschließend in ein Schwarzweißbild konvertiert.

In der *finderechteck*-Funktion werden die vier Eckpunkte des Quadrats ermittelt. Dabei wird nur ein Signal weiterleitet, wenn ein Quadrat gefunden wurde.

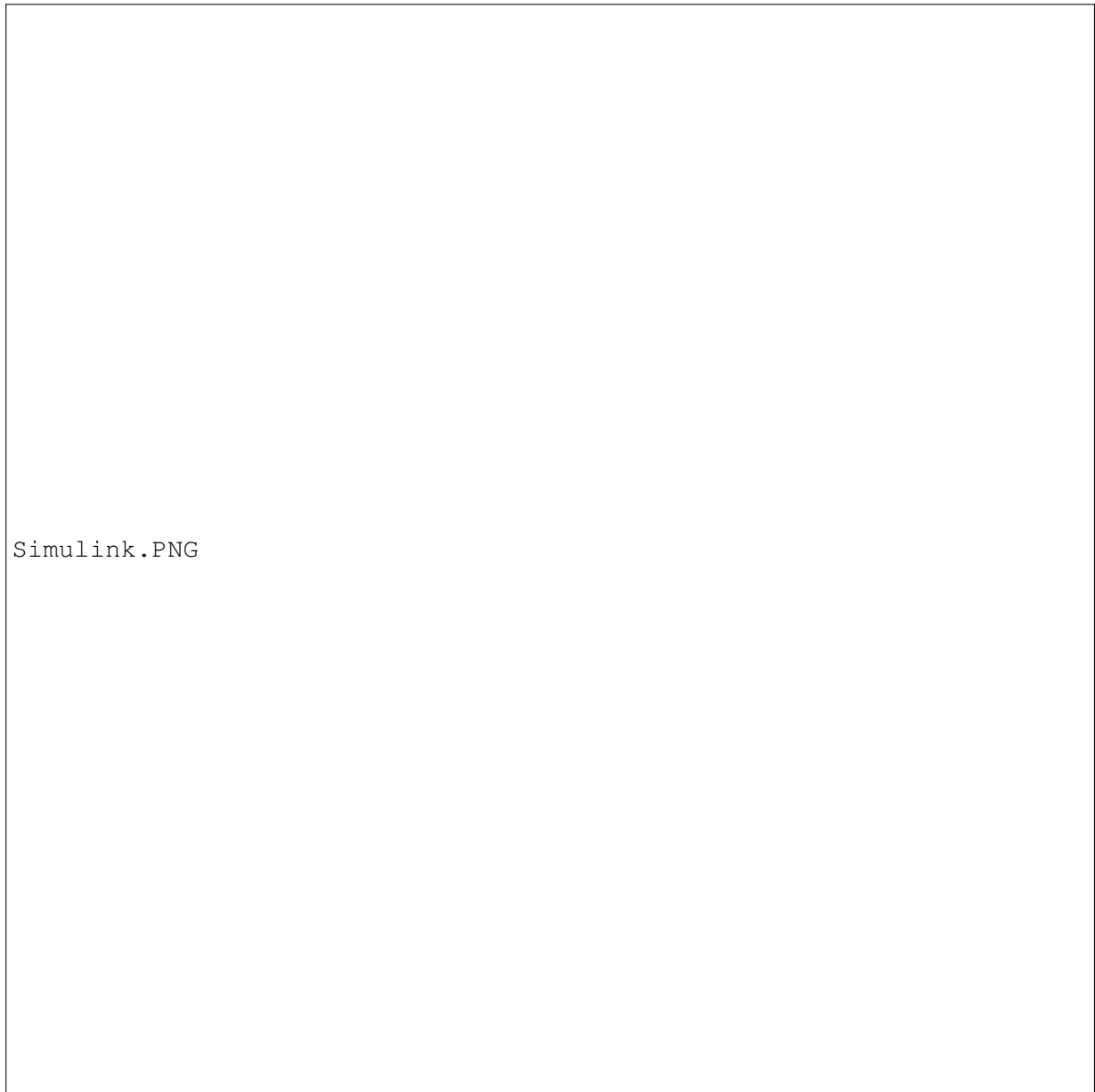
Mit der vorher definierten Höhe des Quadrats über dem Boden, den Kameraparametern und der Transformationsmatrix können die Weltkoordinaten der beiden unteren getrackten Punkte bestimmt werden. Nach der Mittlung und Trennung der Signale wird die x -, und y -Position ausgegeben.

3.2 Abstandsregler

Der implementierte Abstandsregler nutzt die zuvor berechneten Abstände zum getrackten Objekt. Der Abstand in Fahrtrichtung wird durch x_{pos} beschrieben, der Abstand quer zur Fahrtrichtung durch y_{pos} . Für die Abstandsregelung wird der aktuelle Abstand mit einer Zieldistanz verglichen, mittels eines P-Regler geregelt und auf Werte zwischen -0.5 und 0.5 begrenzt. Um ein Rückwärtsfahren bei Unterschreitung des geforderten Abstands zu gewährleisten, wird bei

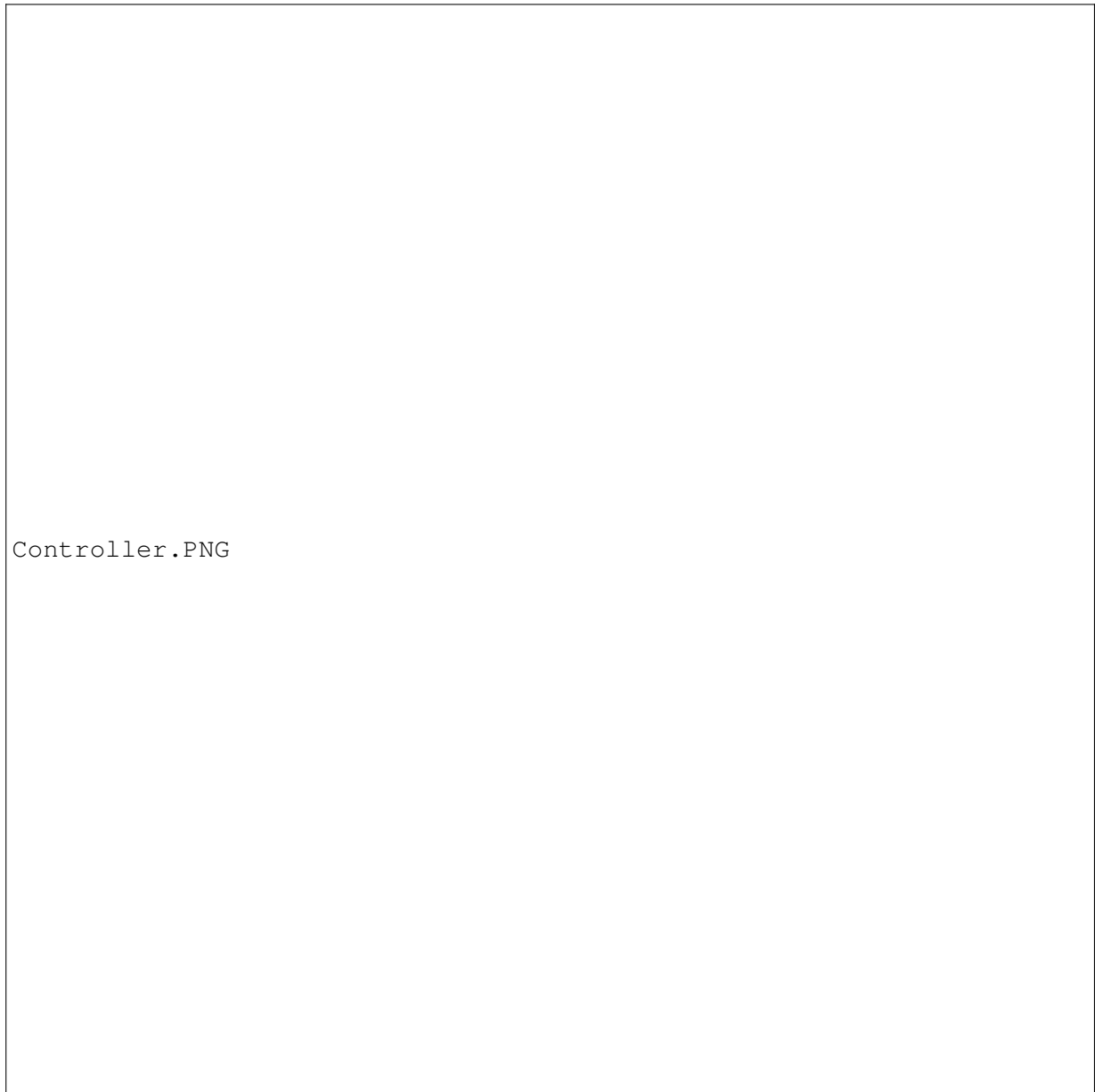
einem negativen Wert die Drehrichtung des Motors umgedreht.

Für die Regelung des Lenkungswinkels wird der gemessene Querabstand durch eine Verstärkung auf einen Wert zwischen -1 und 1 gebracht.



Simulink.PNG

Abbildung 2: Abstandsbestimmung Kamera-Objekt in Simulink



Controller.PNG

Abbildung 3: Abstandsregler in Simulink