Kalibrierungsfreie Bildverarbeitungsalgorithmen zur echtzeitfähigen Objekterkennung im Roboterfußball

Thomas Reinhardt
Nao-Team HTWK

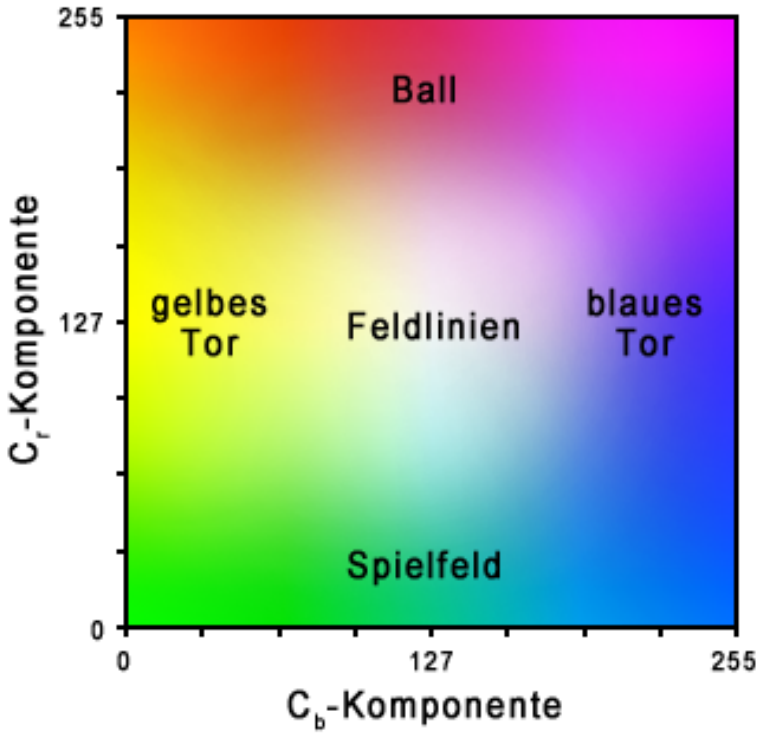
Hochschule für Technik, Wirtschaft und Kultur Leipzig

26. Februar 2011

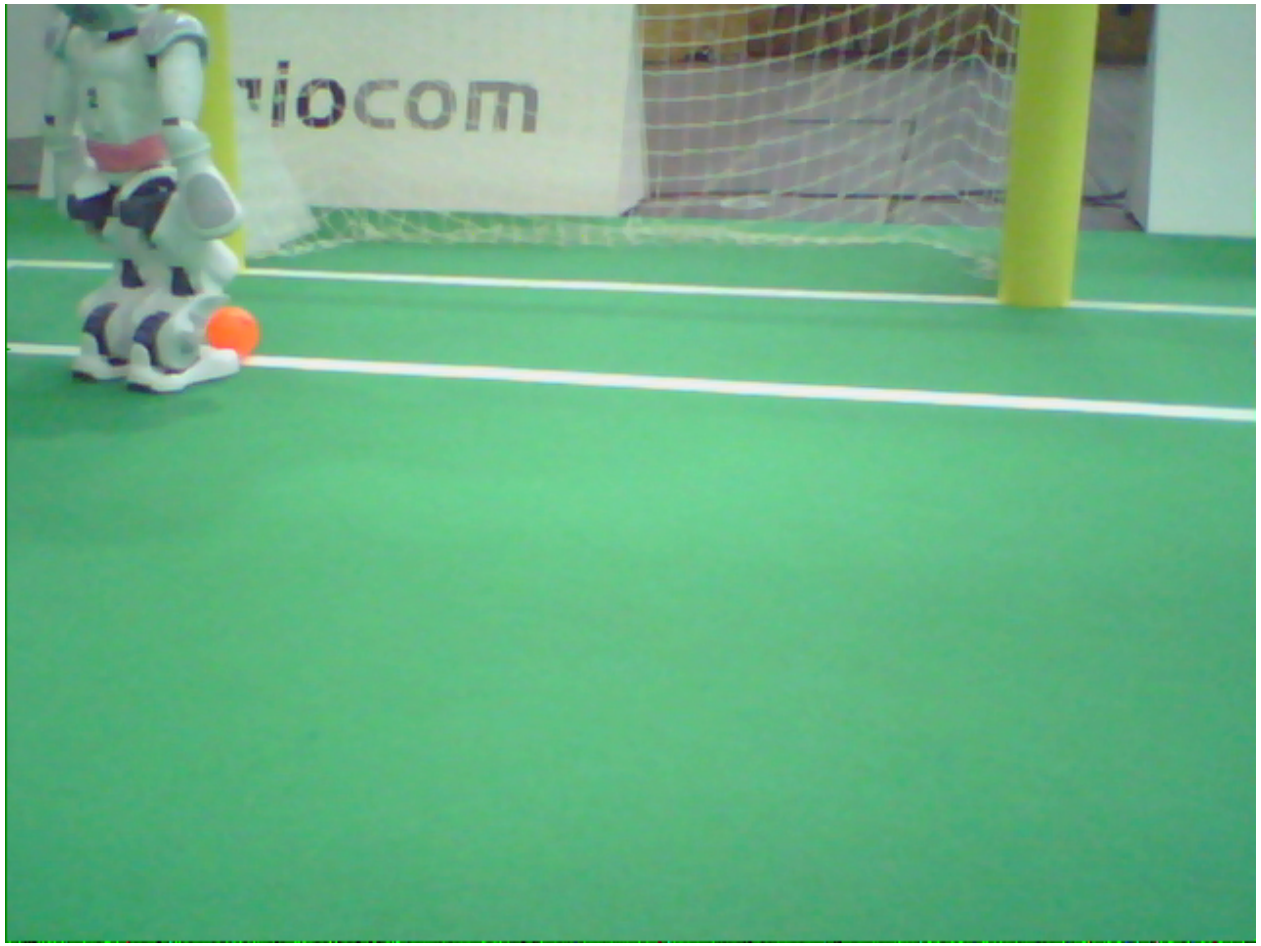
Gliederung

YCbCr-Farbraum

- Helligkeitskomponente (Luminanz, Y)
- zwei Farbkomponenten U und V



Typisches Kamerabild



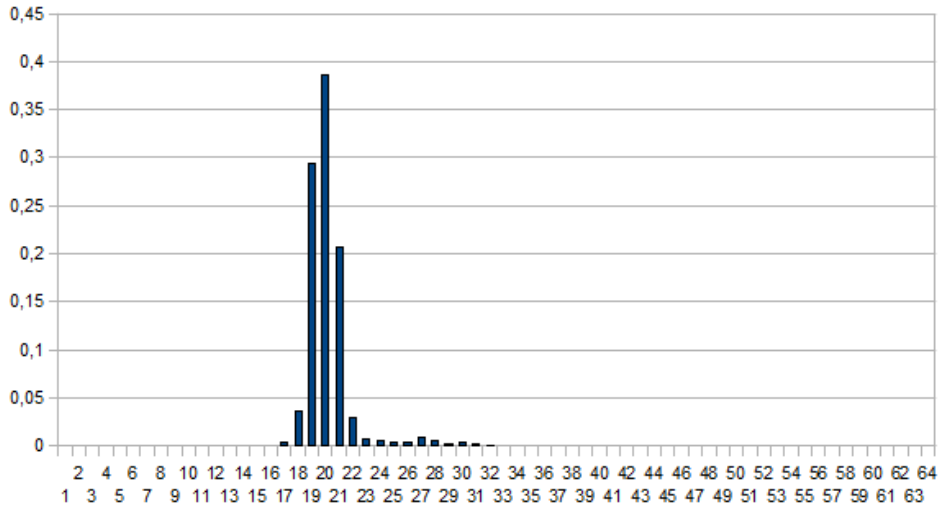
Eigenschaften der Feldfarbe

Eigenschaften des Feldbodens:

- grün gefärbt -> Cr-Komponente,
- statistisch oft die meisten Pixel im Bild
- Zusammenhängend und meist homogen

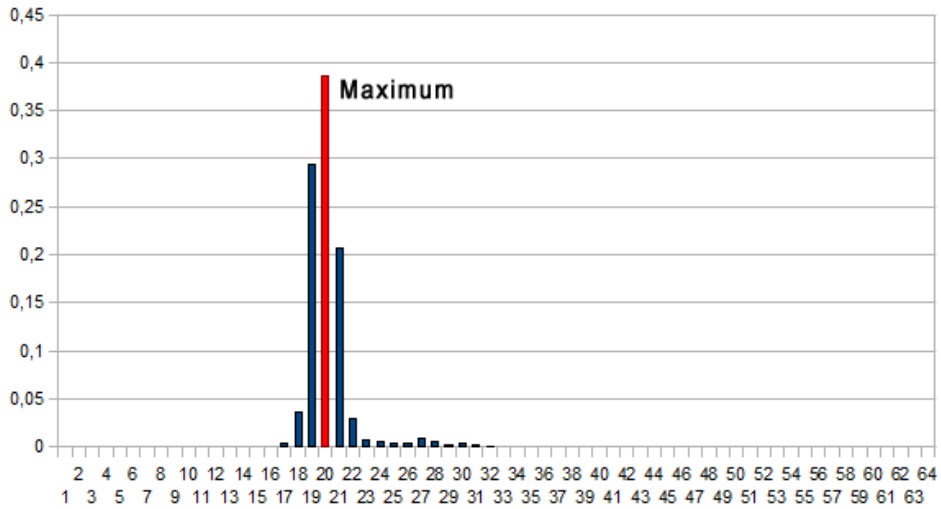
Detektion des dominanten Farbtones

- Berechnung eines Cr-Histogramms von wenigen Samplepixeln
- Quatisierung auf 64 Werte

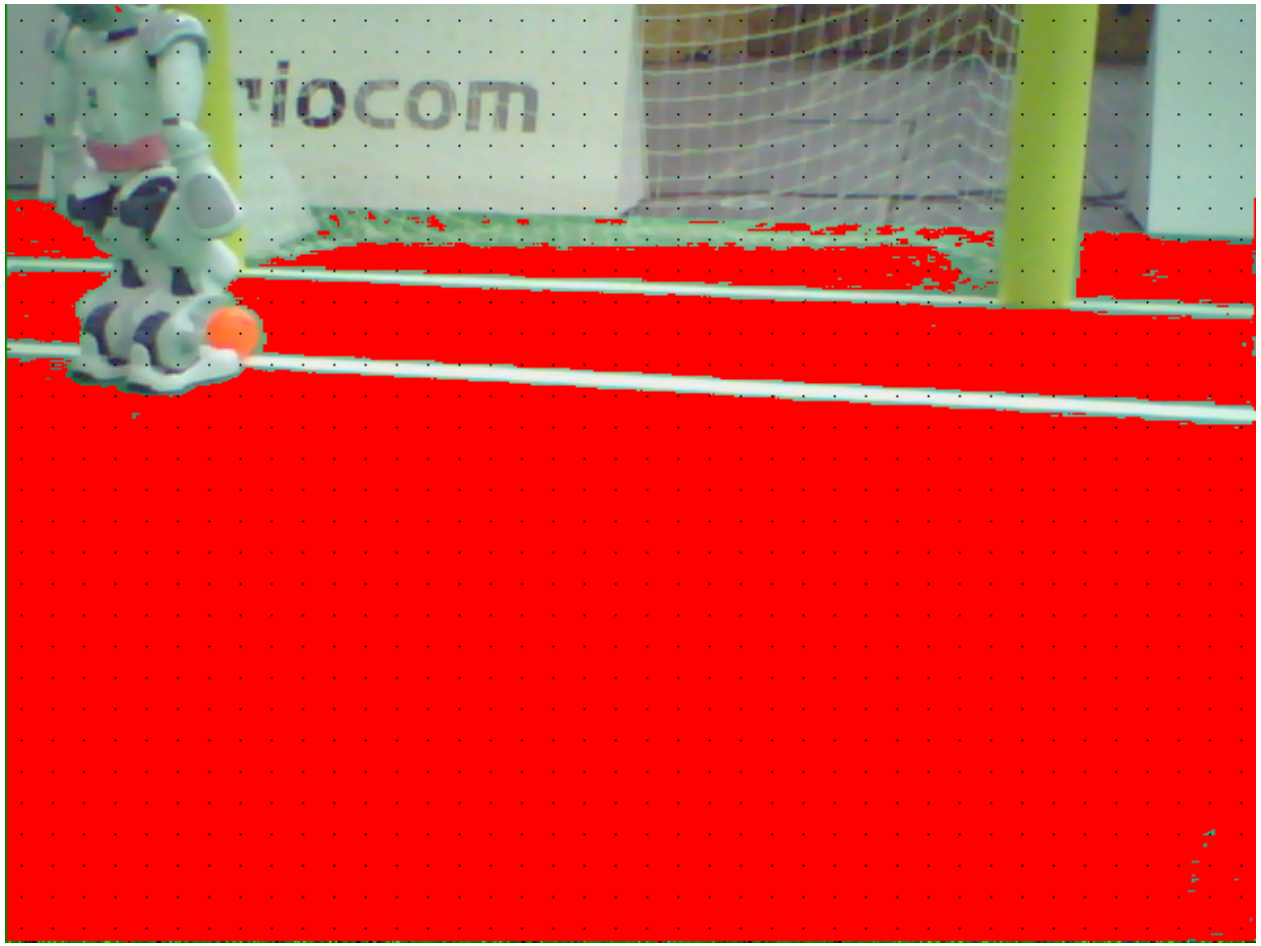


Testdatenbank

- Cr-Wert mit höchstem Wert im Histogramm bestimmen (Cr_{max})

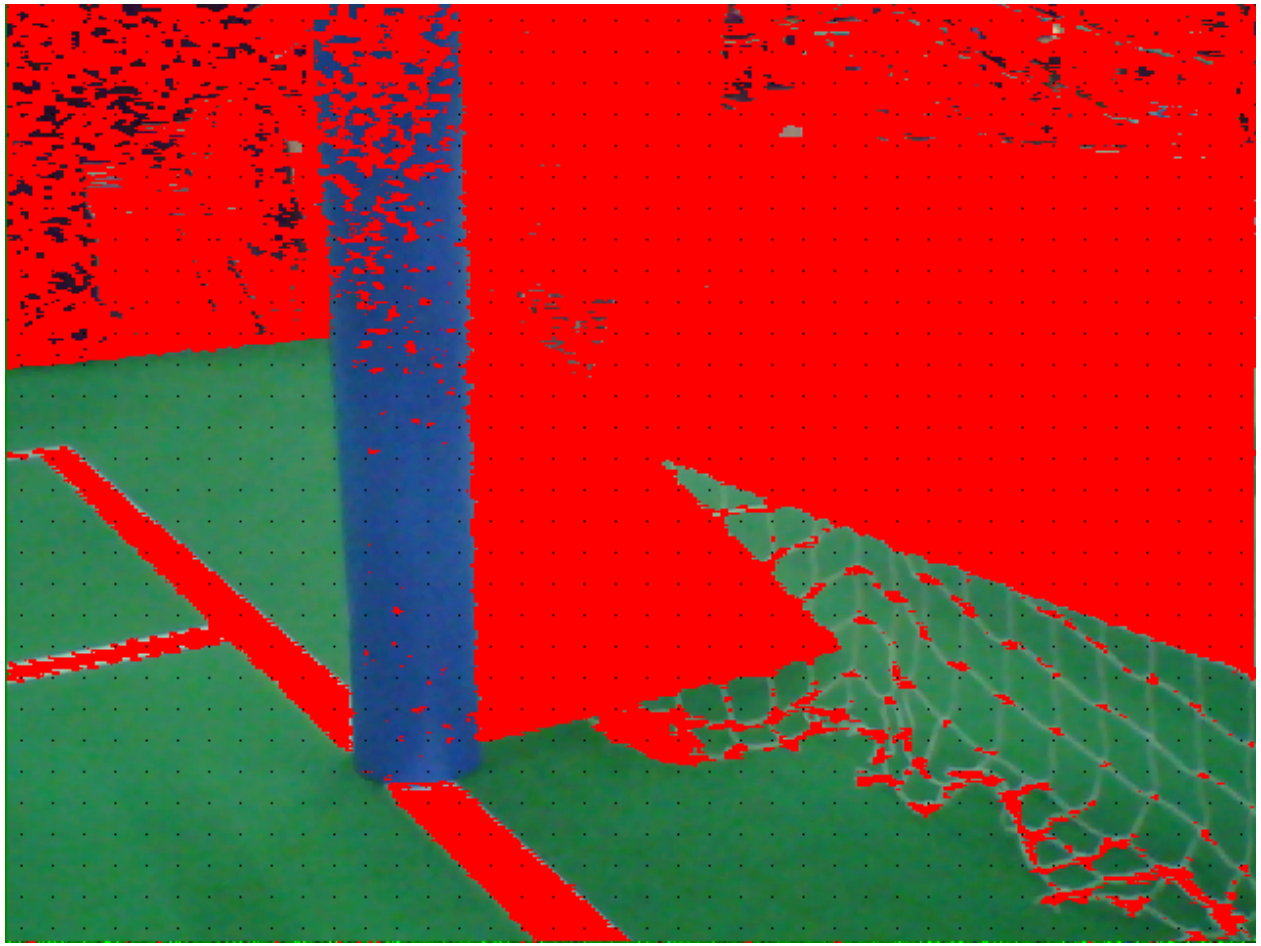


Bodenerkennung

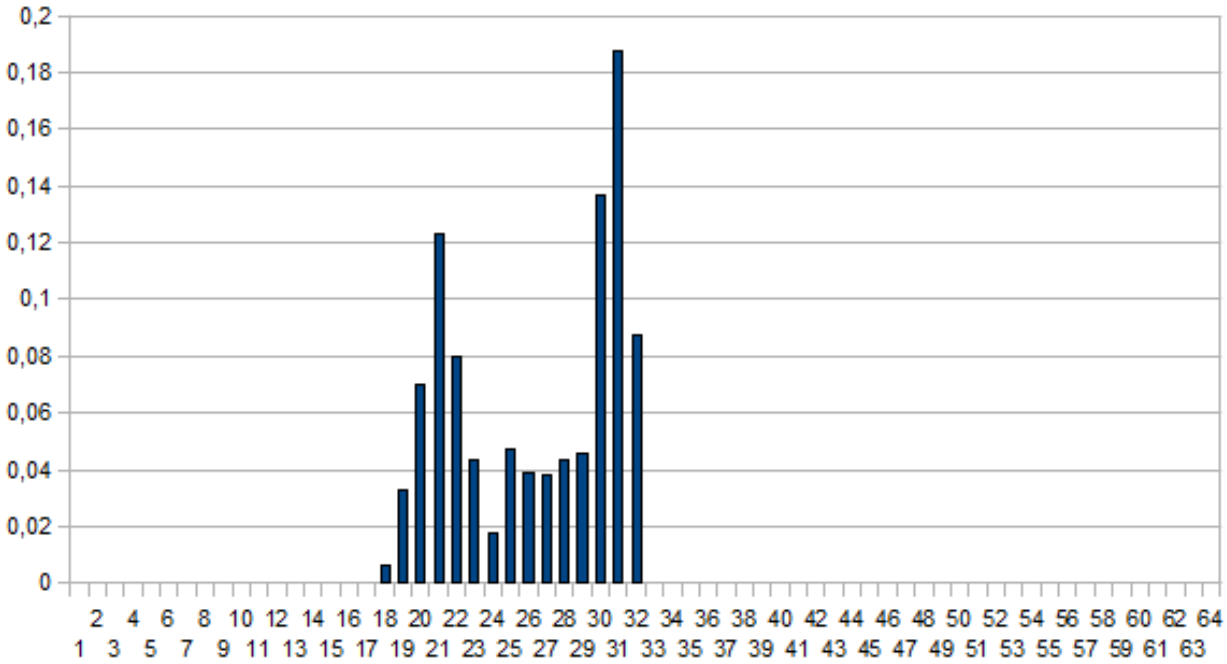


Zweites Beispiel

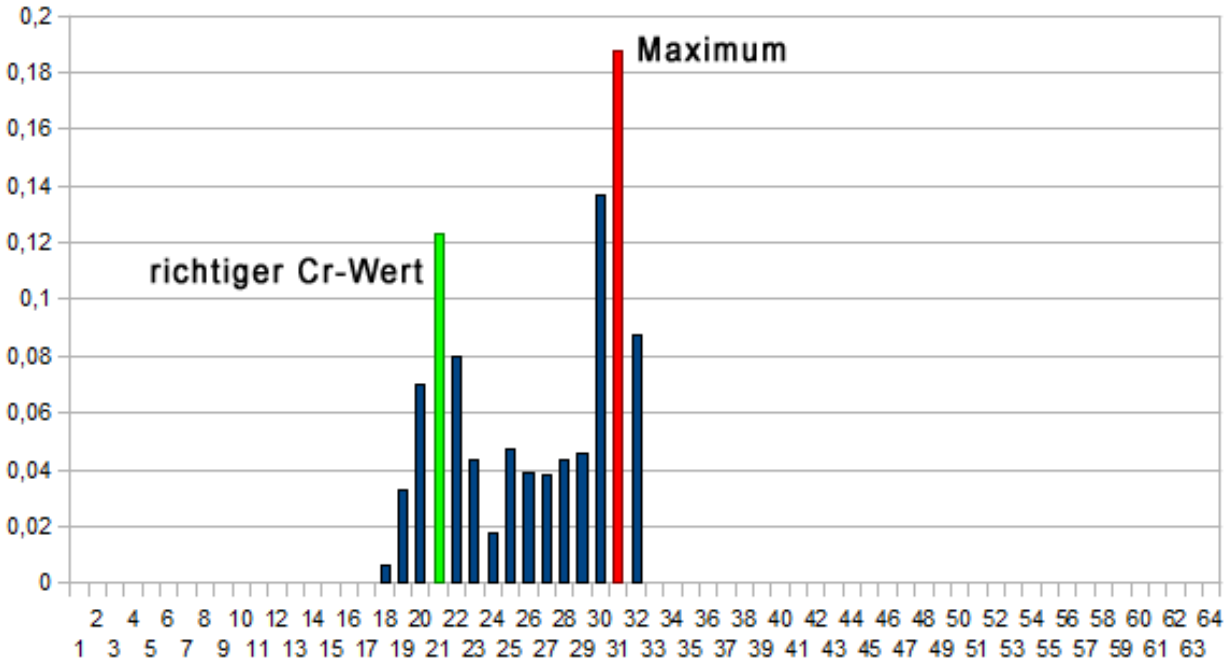
Fehlerkennung



Histogramm


0,2
0,18
0,16
0,14
0,12
0,1
0,08
0,06
0,04
0,02
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64

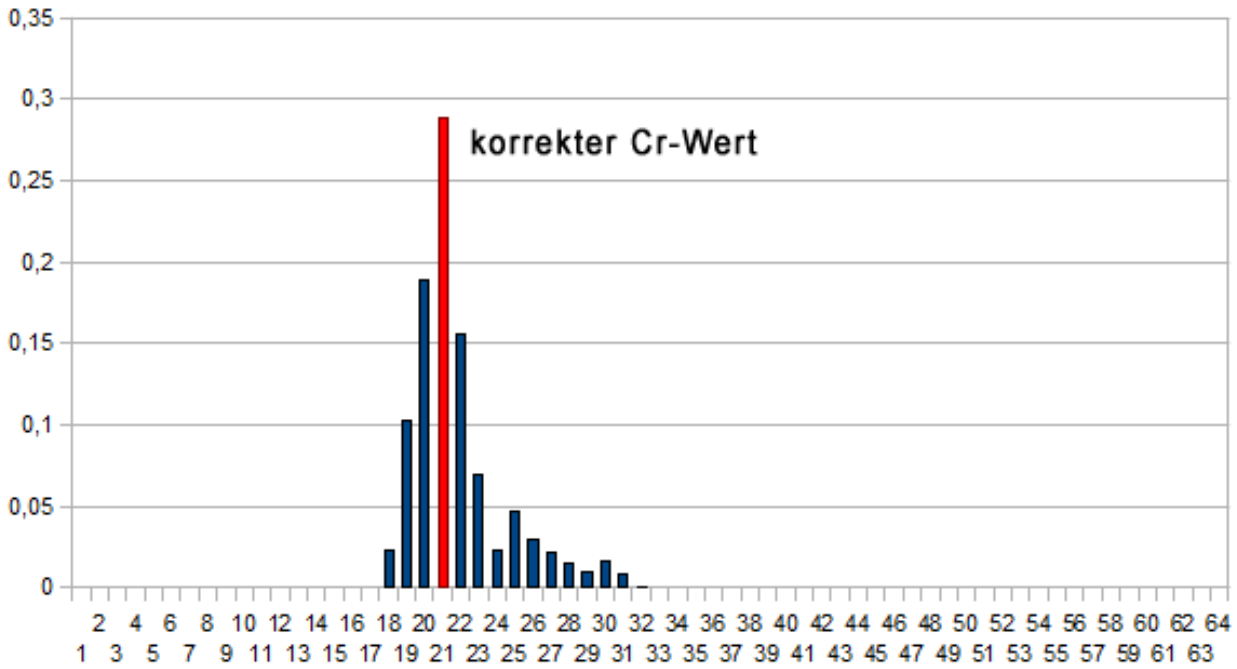
Histogramm


Maximum
richtiger Cr-Wert
0,2
0,18
0,16
0,14
0,12
0,1
0,08
0,06
0,04
0,02
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64

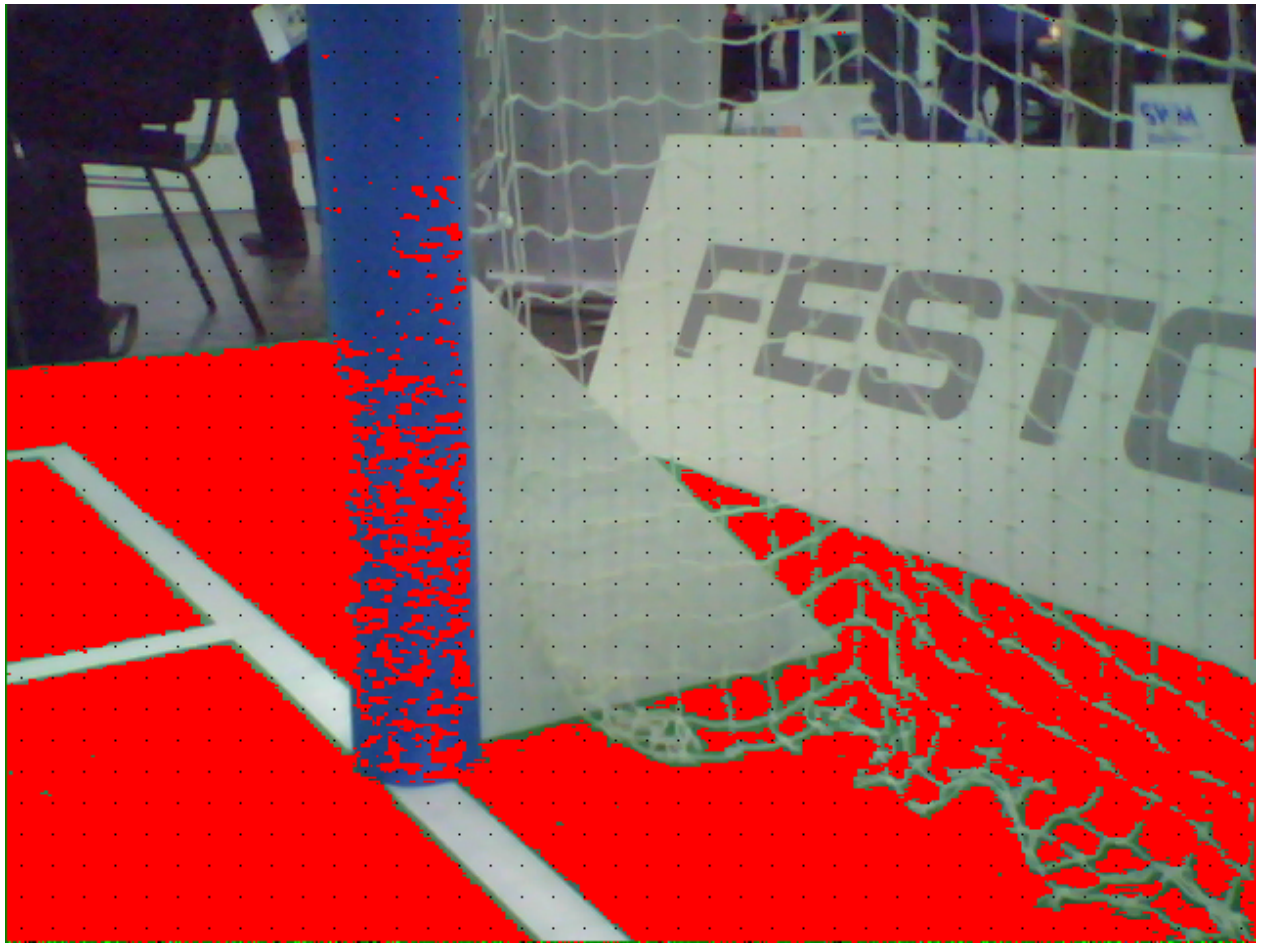
Lösung

- Gewichtung der Histogrammwerte nach Cr-Wert
- Kleinere Werte im Cr-Kanal -> kräftigeres Grün
- Gewichtung: $g = \max(0, 128 - Cr_{xy})^2$

besseres Histogramm



besseres Resultat



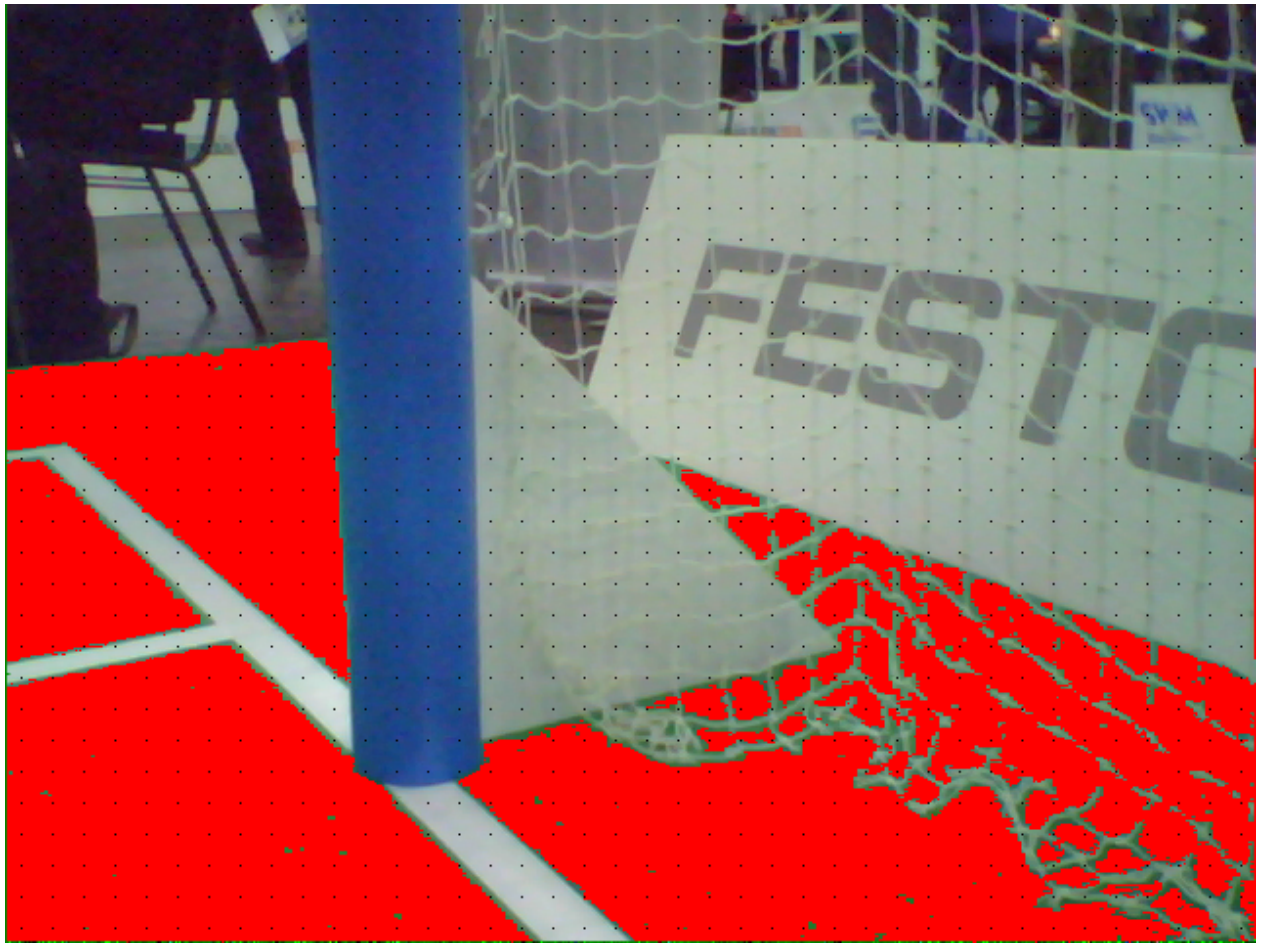
Verwendung von Cb und Y

- bisher: nur Auswertung von Cr-Kanal
- Problem: blaues Tor wird als grüner Boden erkannt
- Lösung:
  - Erstellung der (ungewichteten) Histogramme für Cb- und Y-Kanal
  - Nur Pixel betrachtet, für die bereits $|Cr_{max} - Cr_{xy}| < maxDistance$ gilt
  - Finden von Cb_{max} und Y_{max}

Endgültige Klassifikation von Bodenpixeln

- Grüner Boden, wenn:
  $|Cr_{max} - Cr_{xy}| < maxDistance_{Cr} \bigwedge$
  $|Cb_{max} - Cb_{xy}| < maxDistance_{Cb} \bigwedge$
  $|Y_{max} - Y_{xy}| < maxDistance_{Y}$

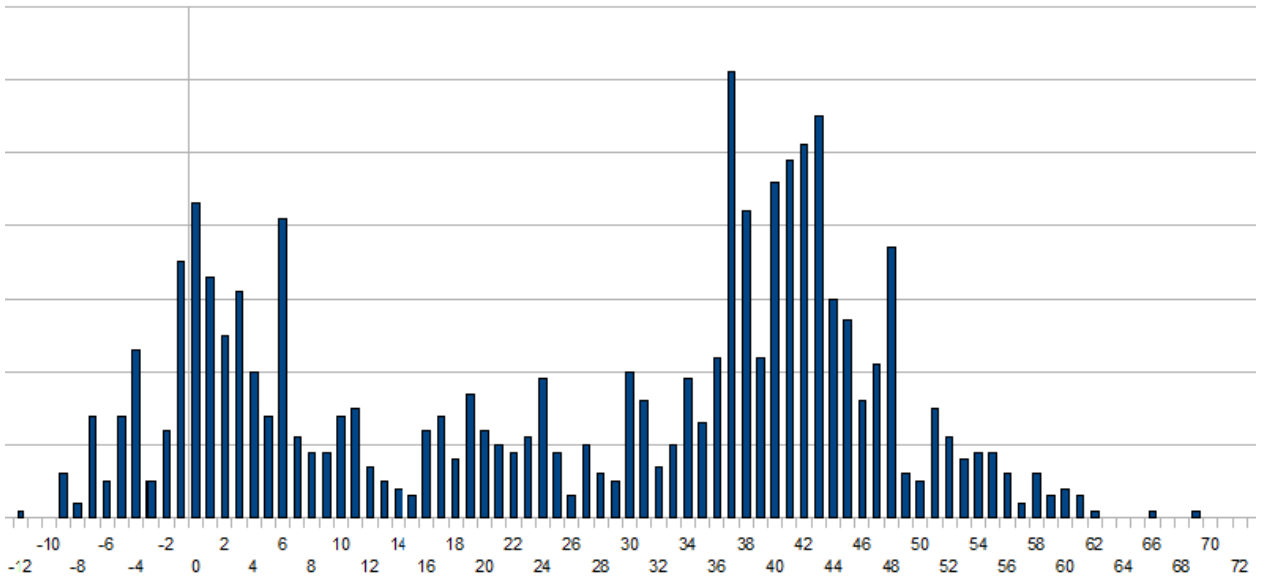
Ergebnis



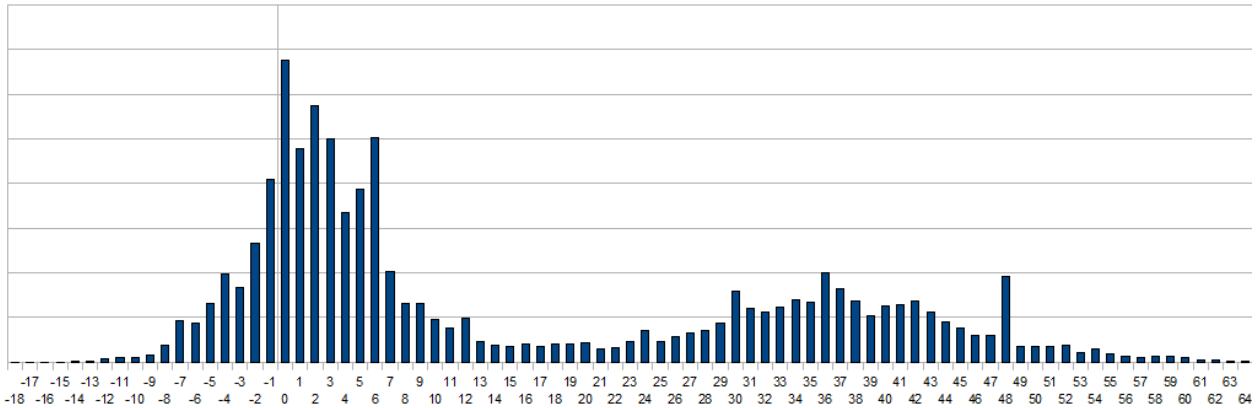
Methoden um maxDistance festzulegen

- Richtwert, um $maxDistance_{Cr}$ einzustellen?
- Idee: Verteilung der Differenzen zu Cr_{max} für alle Pixel des Bildes betrachten

Histogramm der Differenzen



gemitteltes Histogramm der Differenzen

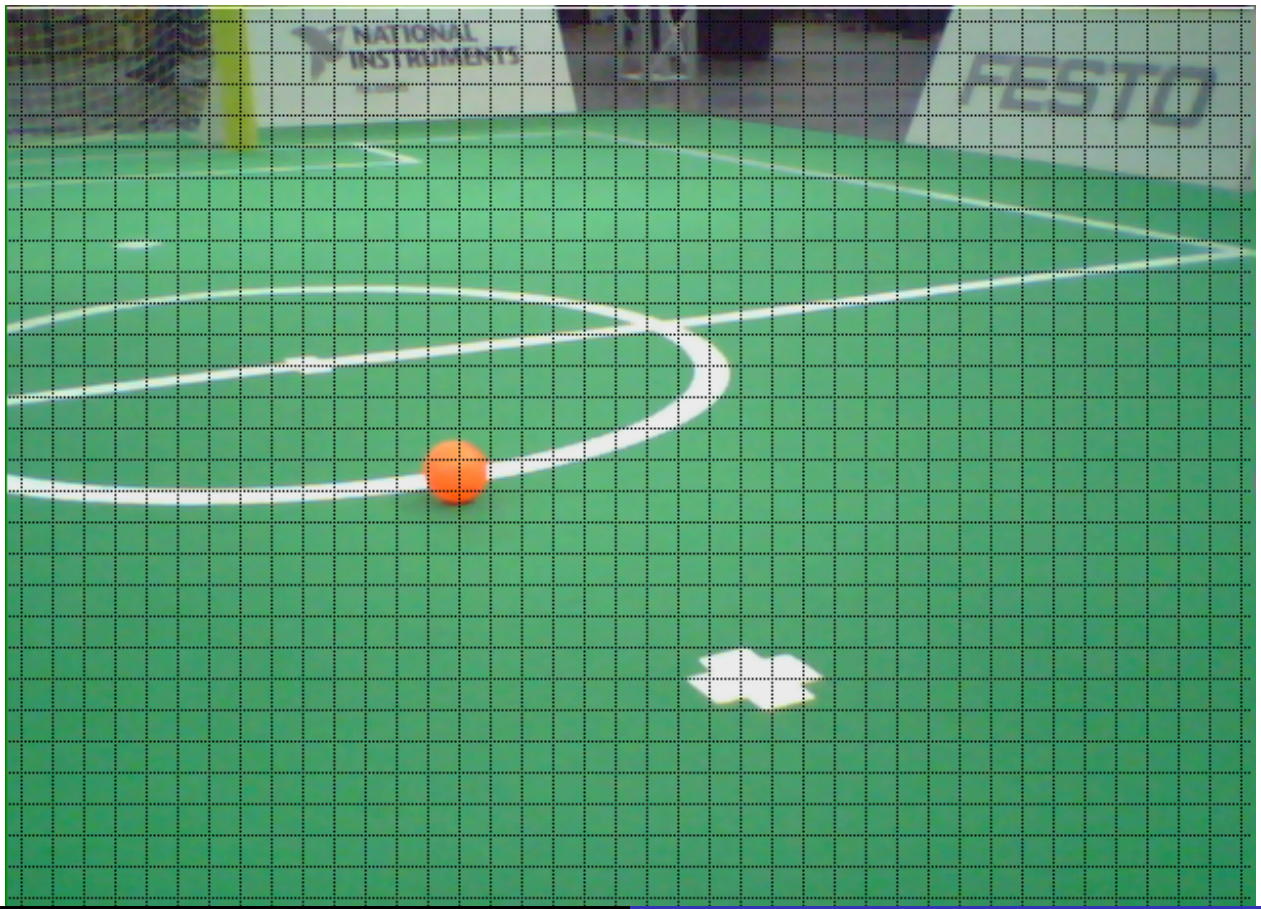


- C_r konstant auf 15 setzen (guter Wert für alle Testbilder)
- Alternative: Schwellwert zur Trennung bestimmen mit Otsu-Schwellwertverfahren (ggf. riskant)

Linienerkennung

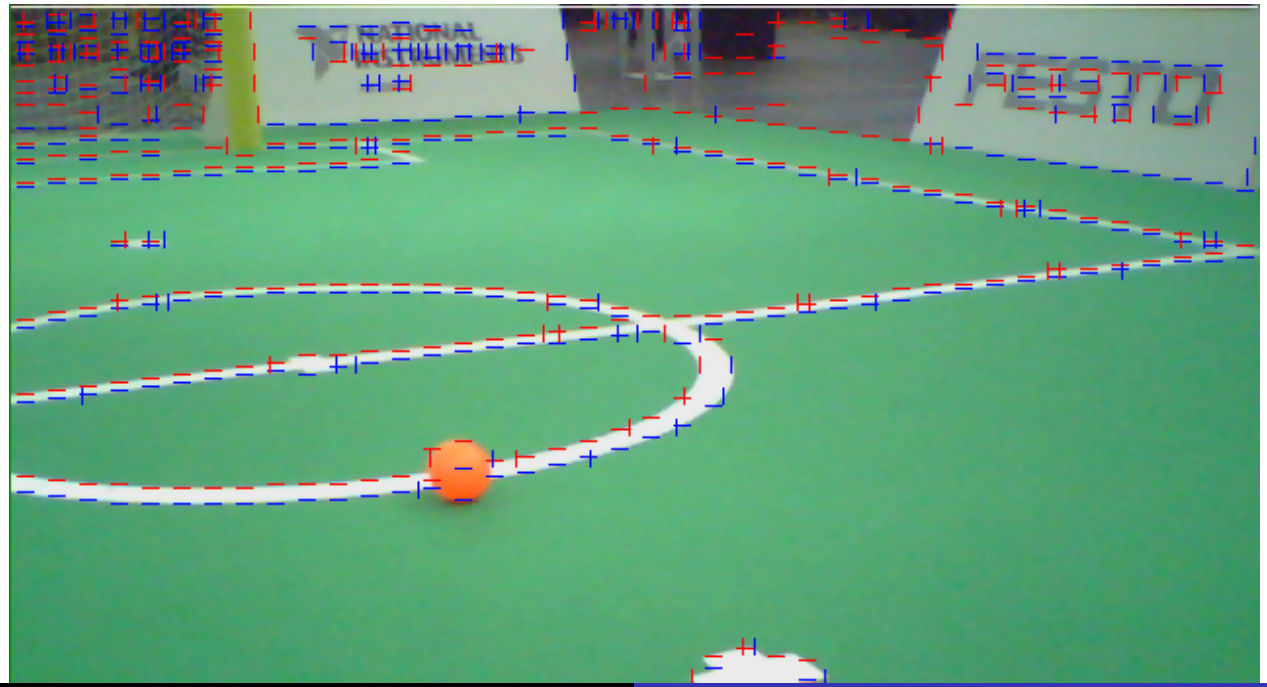
Linienerkennung

- Scanlines (16 px spacing)



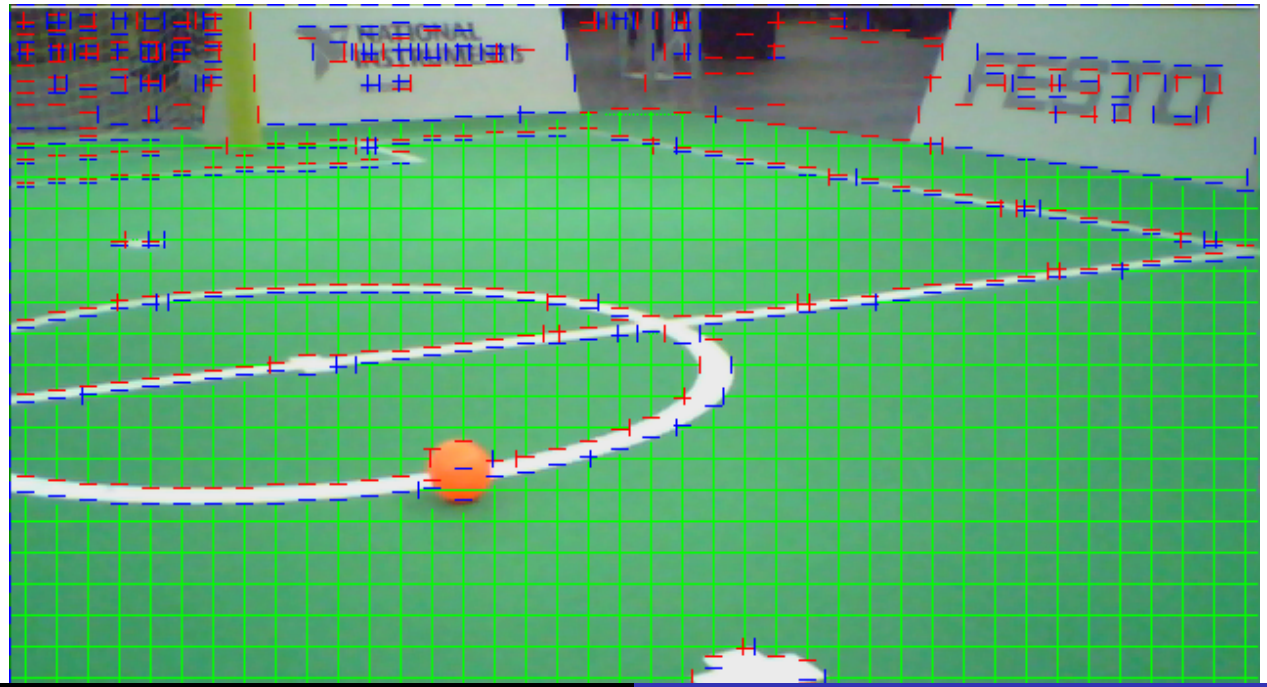
Linienerkennung

- Kantendetektion im Y-Kanal entlang der Scanlines
- Detektor für »grün->heller« und »heller->grün«
- Auftrennung der Scanlines an diesen Kanten:
  - Erstellung von homogenen, Pixelketten



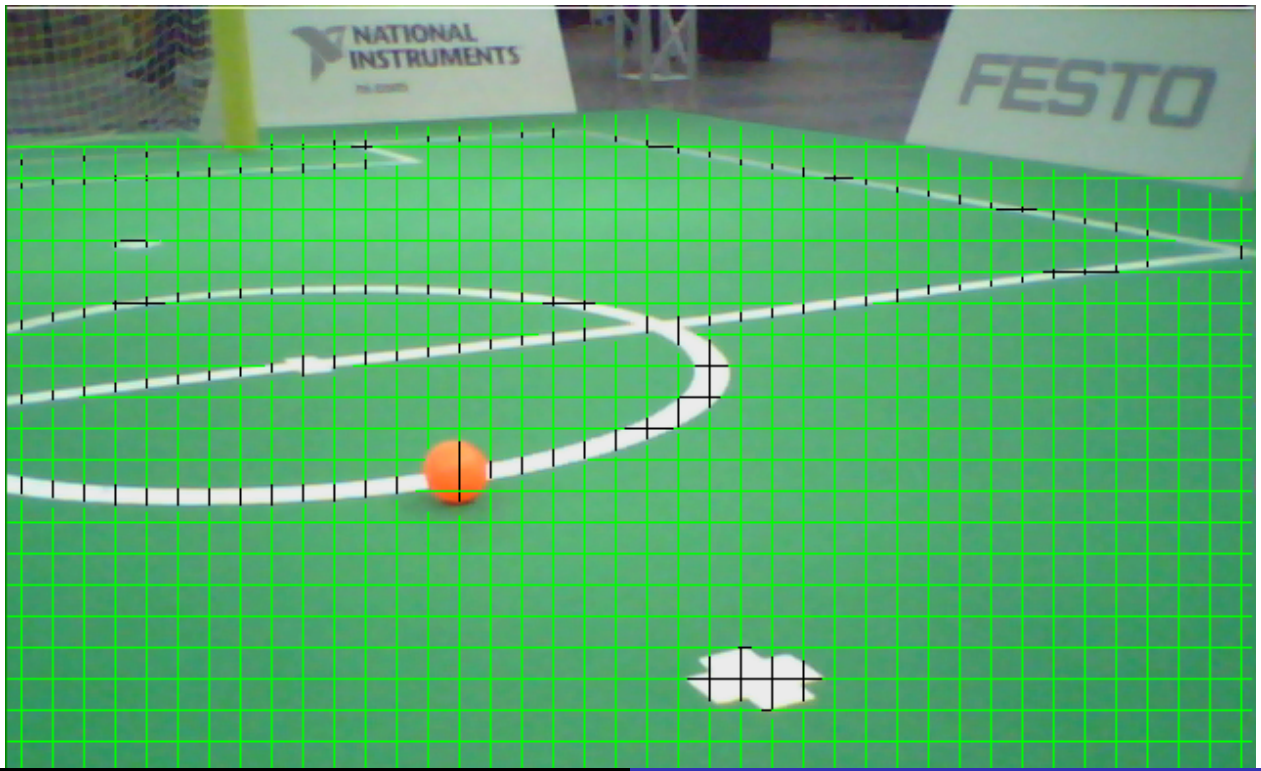
Grünerkennung

- zunächst: Grünsegmente erkennen (da Linie immer zwischen zwei grünen Flächen)
- Alle Pixelketten als »grünes Feld« markieren, für die gilt:
  - mehr als 50% der verketteten Pixel werden als grün klassifiziert



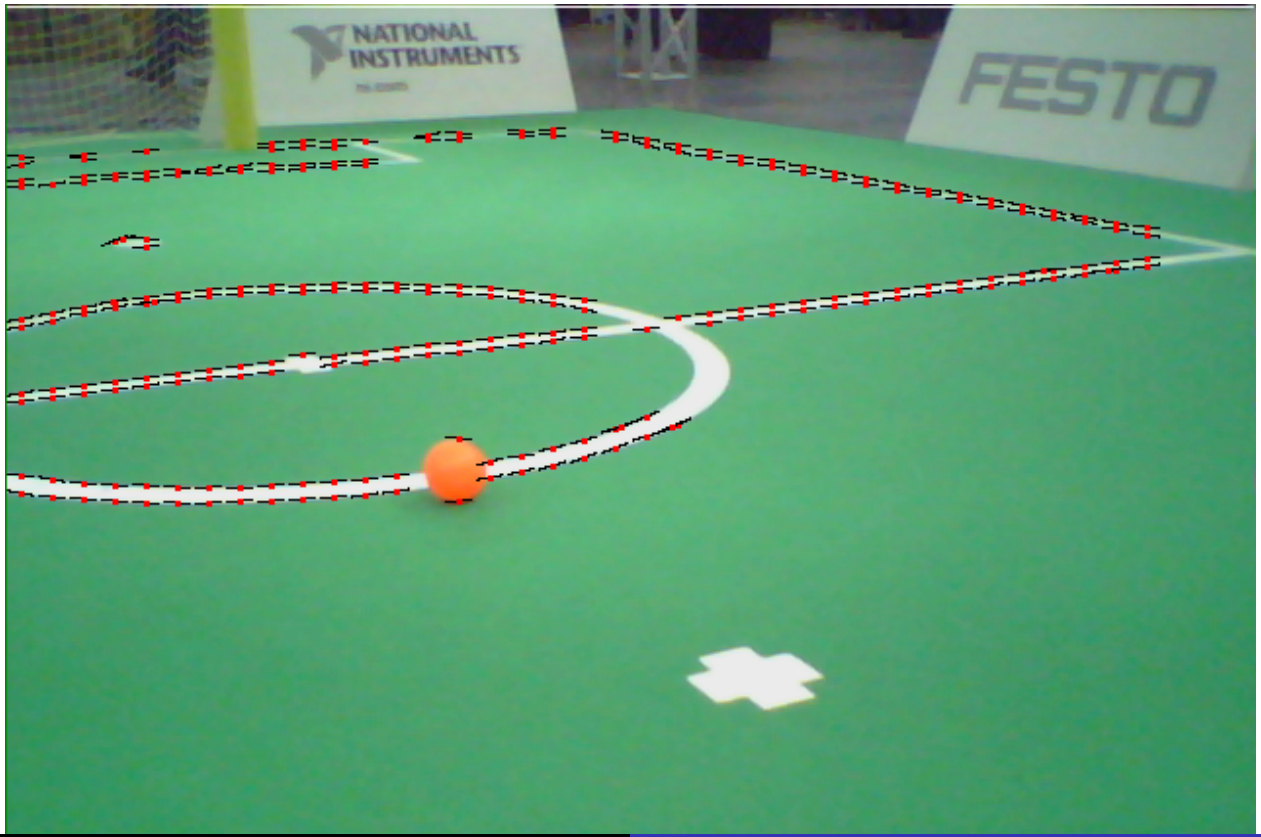
Grünerkennung

- Liniensegmente detektieren:
  - grün-Linie-grün
  - gegensätzliche Gradientenrichtungen der Kanten



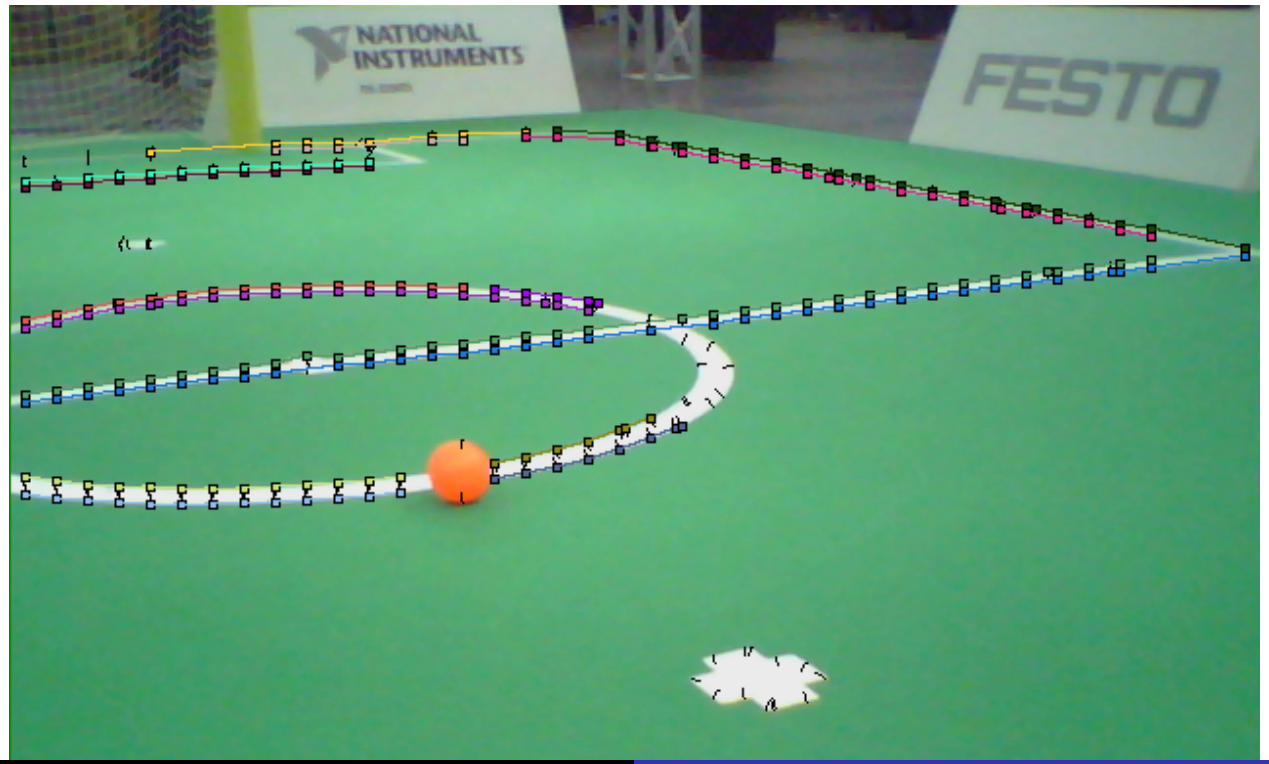
Grünerkennung

- aus einem LinienSegment zwei Linienkanten bestimmen
- jeweils Winkel dieser Kante berechnen



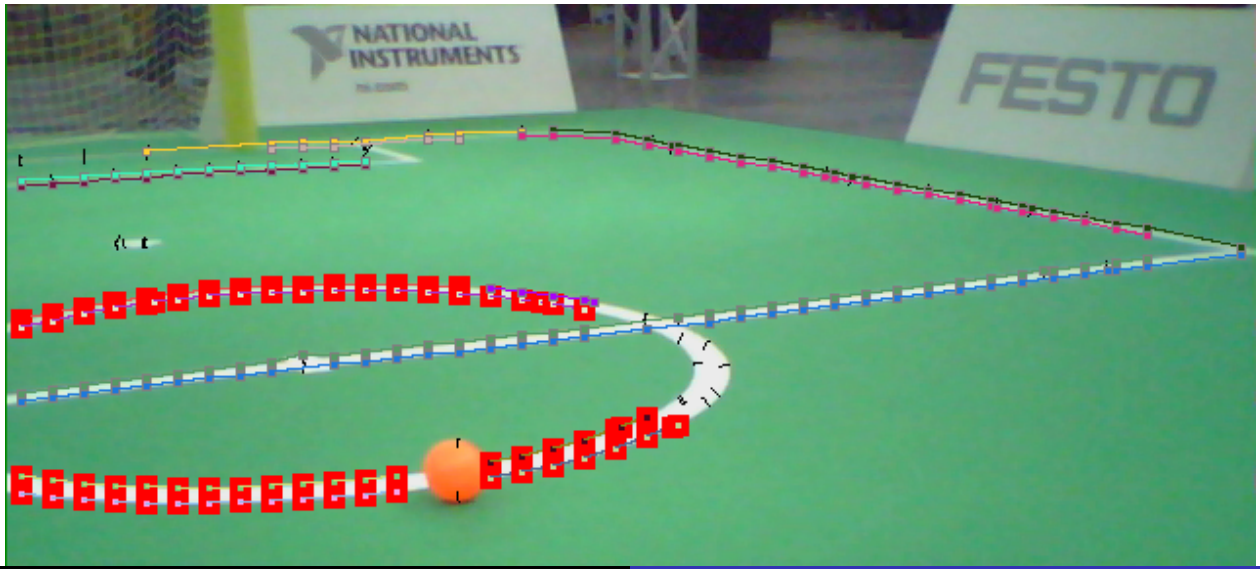
Gruppierung der Liniensegmente

- 1. Durchlauf: Nachbarzuordnung für nahgelegene Liniensegmente gleicher Ausrichtung
- 2. Durchlauf: zusammenhängende Linienstücke zu finden

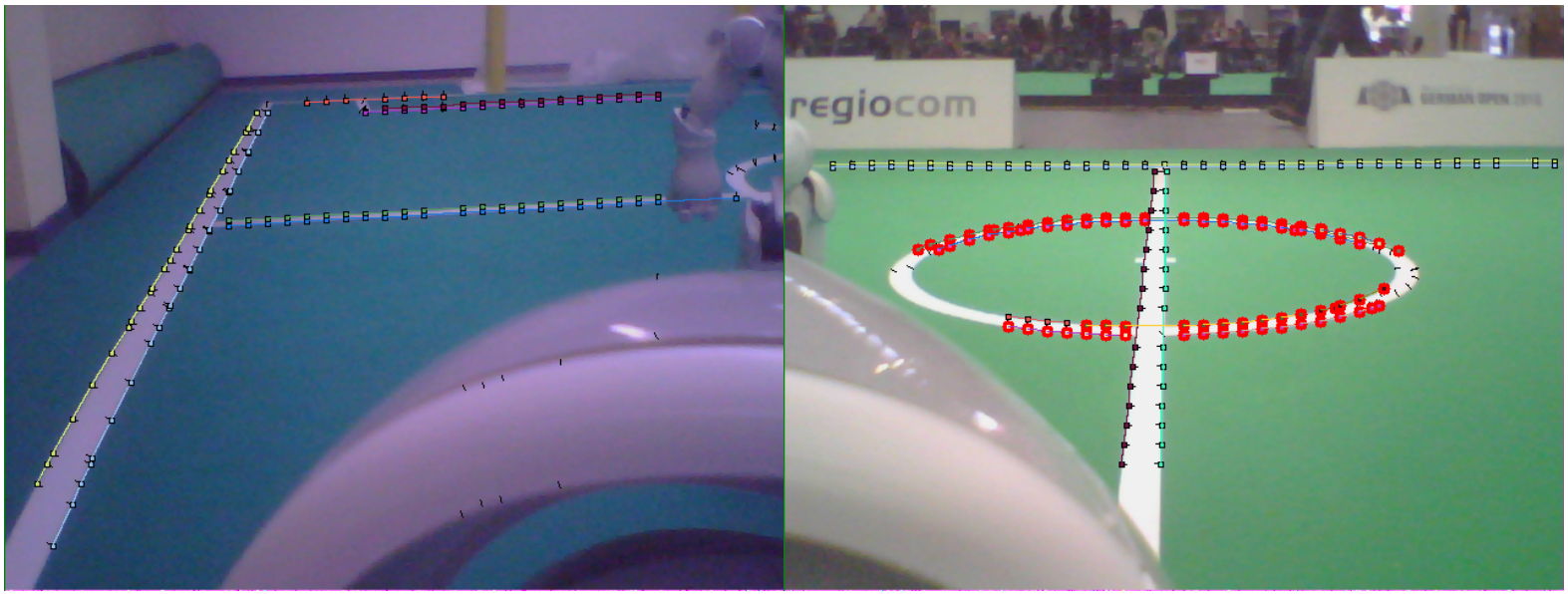


Kreisbogen-Erkennung

- Kreisbogen erkennen:
  - Aufteilung zusammenhängender Segmente in zwei Gruppen (left,right)
  - Ausgleichsgerade für beide Gruppen
  - Winkel der Ausgleichsgerade bestimmen
  - (Winkel größer als Schwellwert -> Kreisbogen)

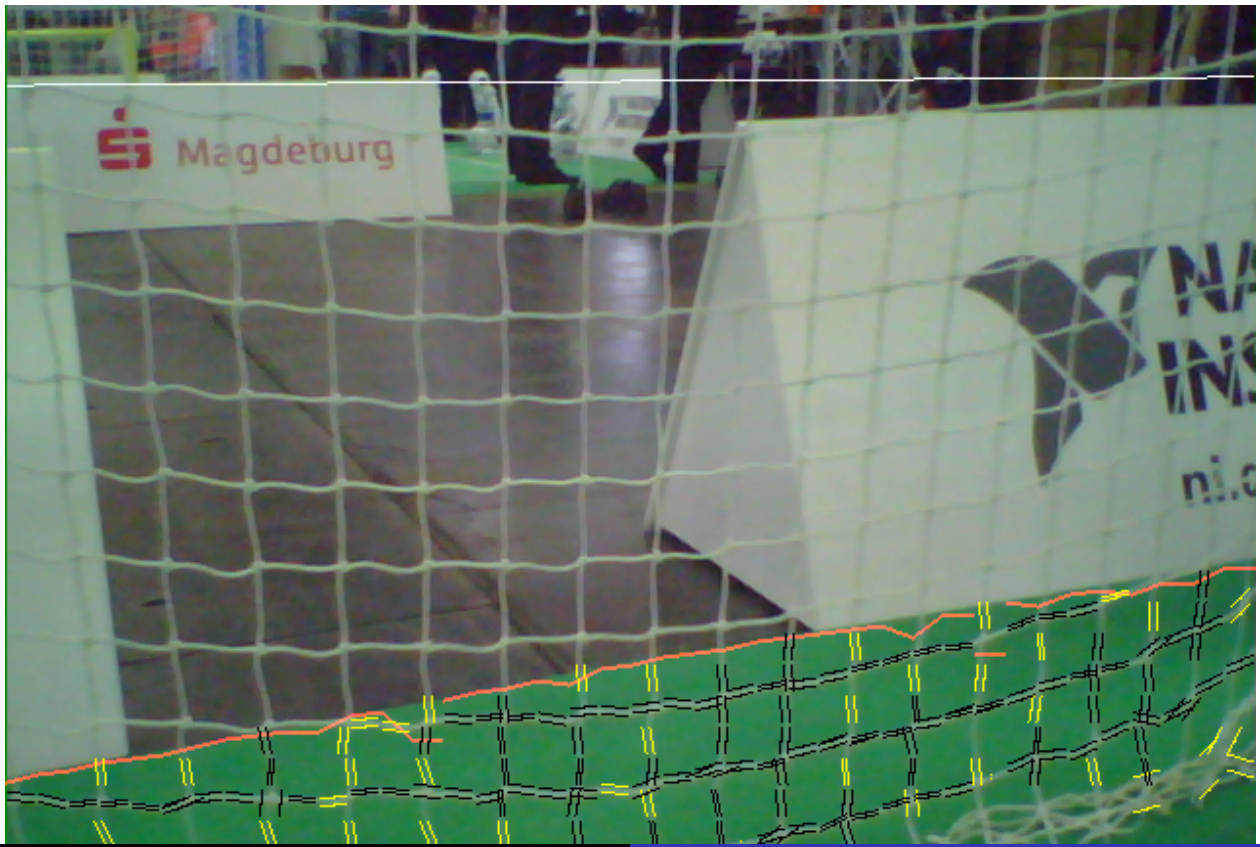


Ergebnisse



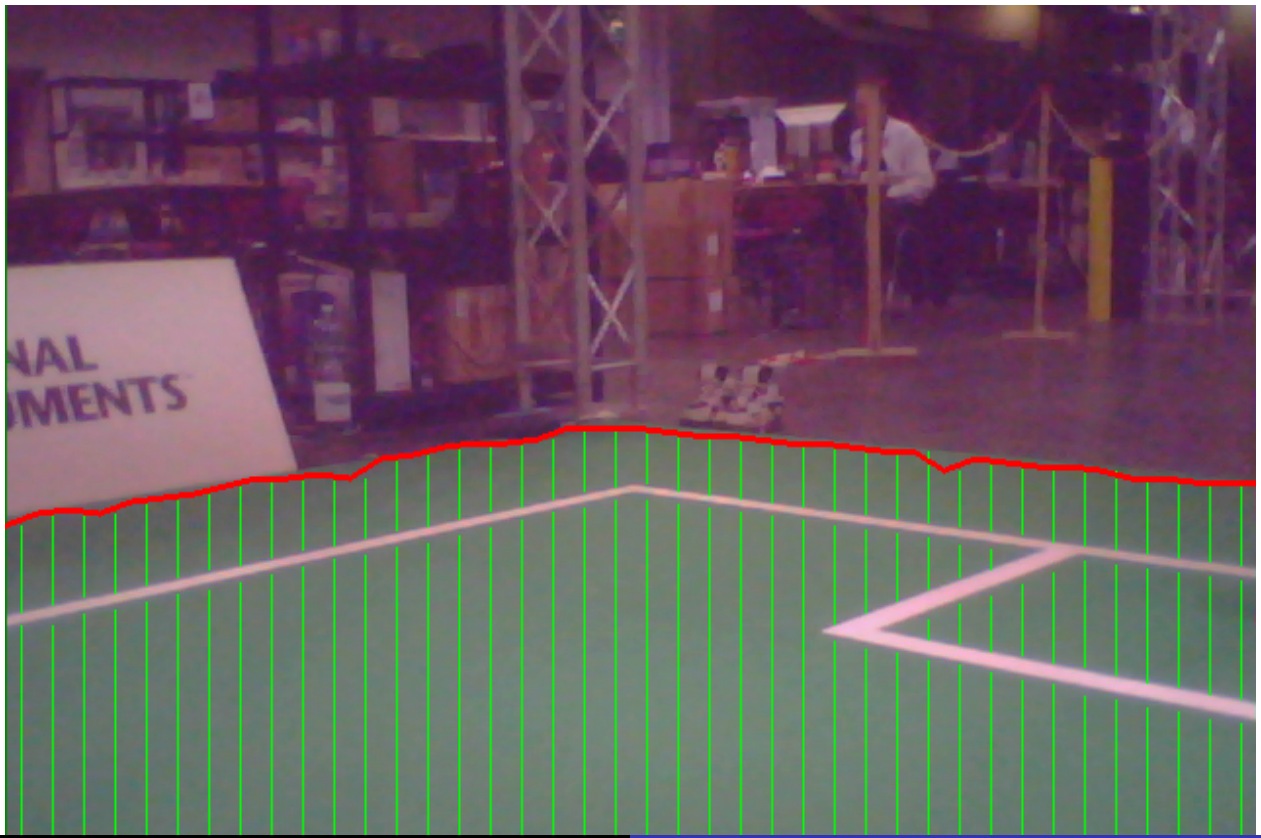
Netz vom Tor

- Zählen von nicht zugeordneten Linien-Segmenten
- Anzahl > Schwellwert > Netz vom Tor



Feldgrenze bestimmen

- Feldgrenze verläuft über den obersten Punkten der erkannten Grünsegmente



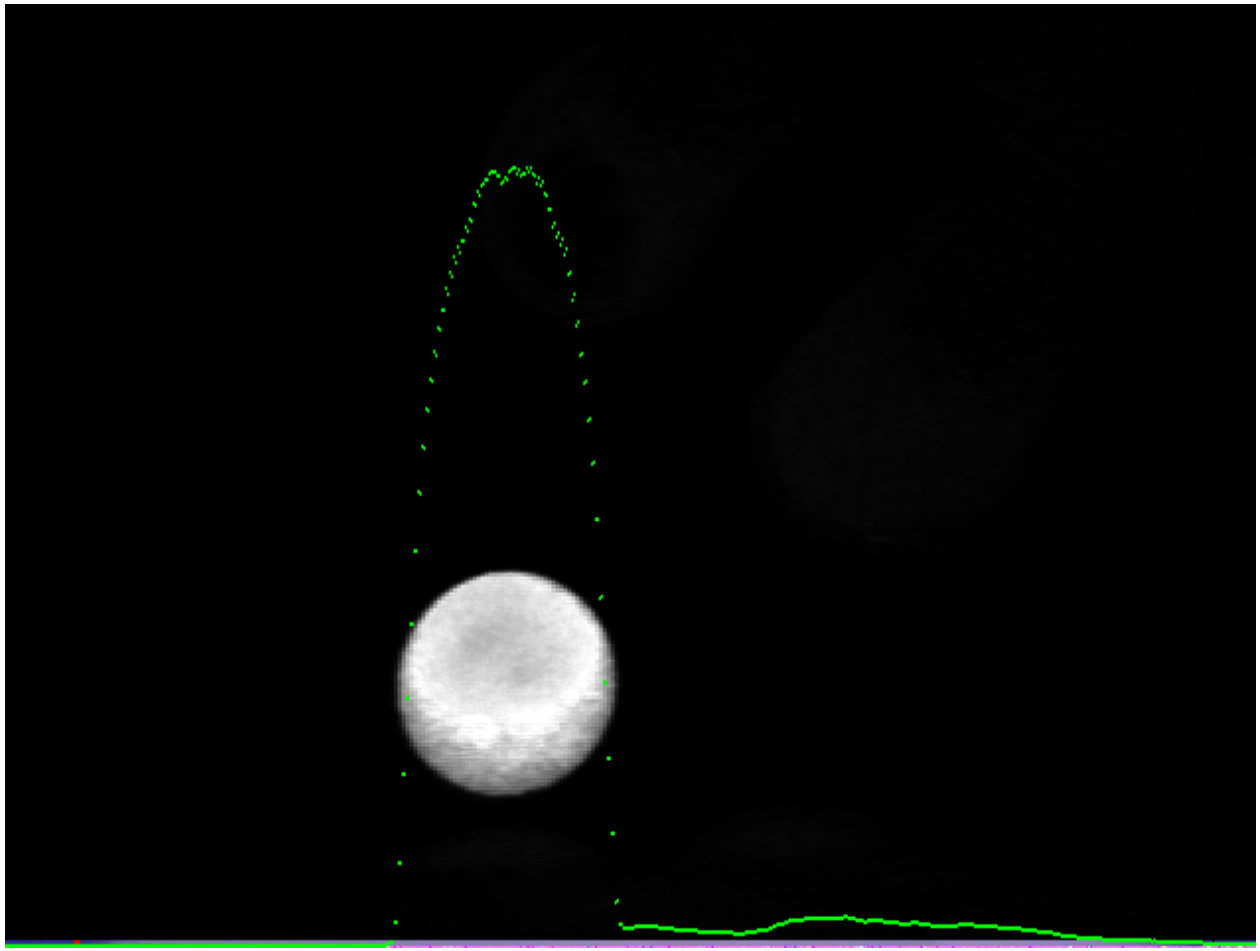
Problem

- Problem: benachbartes Feld in Sicht
- Lösung: Bestimmung möglicher Feldgrenzenpunkte
- RANSAC-Linefitting dieser Punkte -> ergibt maximale Höhe der Feldgrenze

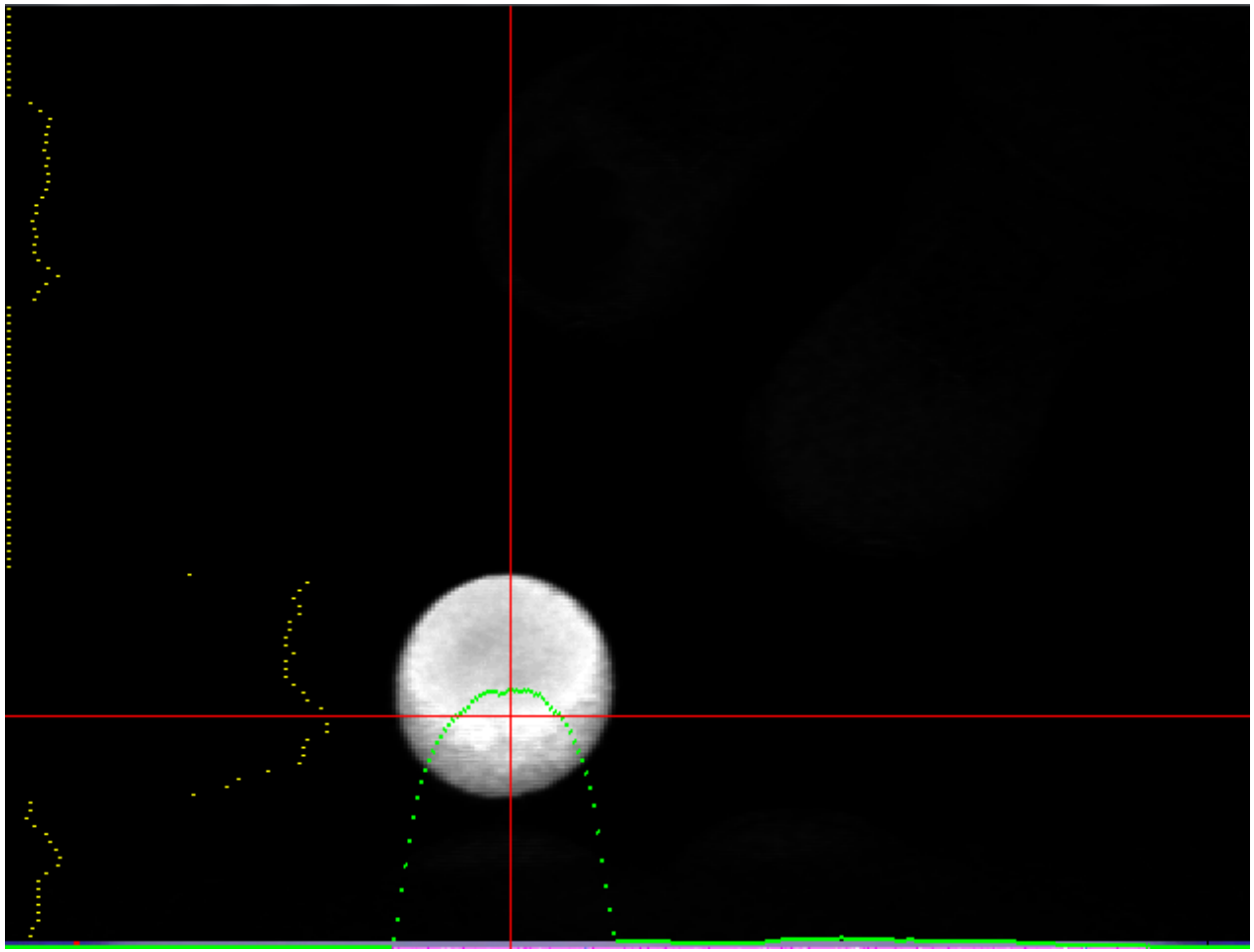
Ballerkennung

Filterung

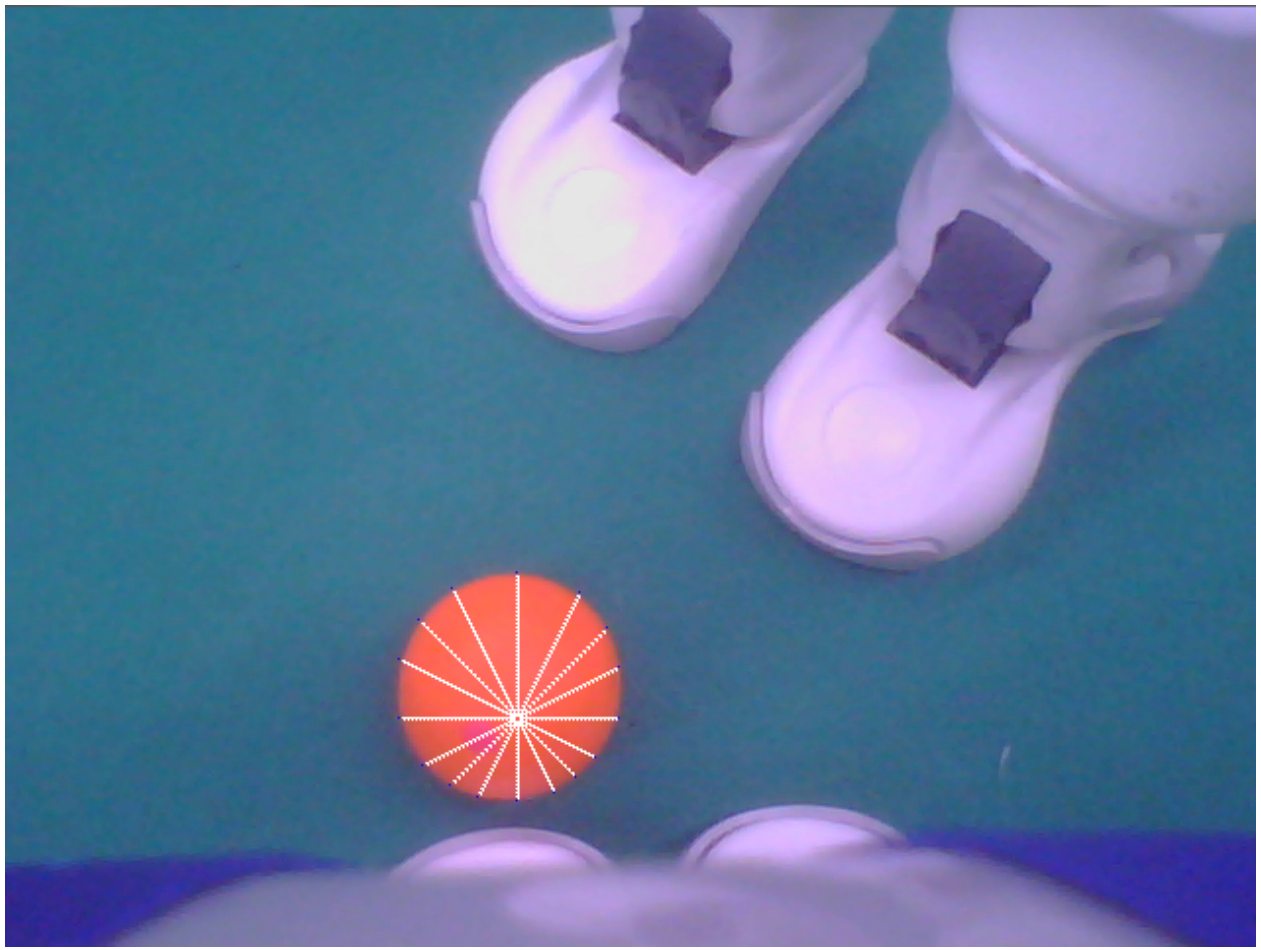
grobe Koordinate berechnen



grobe Koordinate berechnen



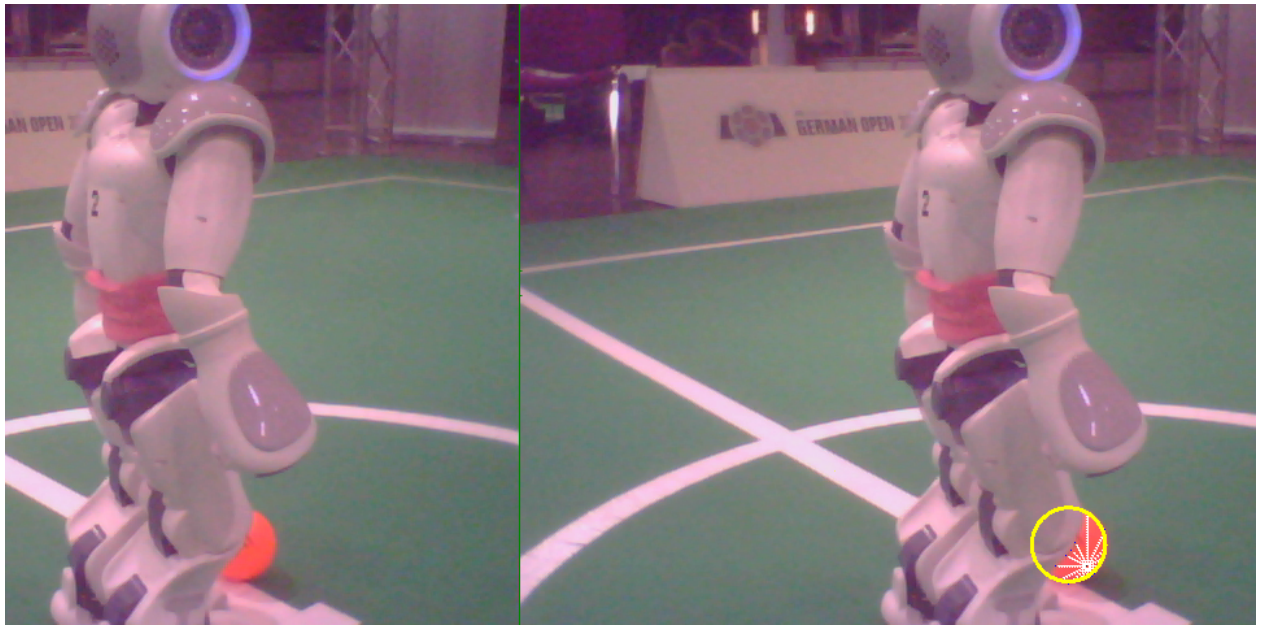
Sternenförmige Scanlines



Kreismatching mit RANSAC

Ergebnisse

Ergebnisse



Torerkennung

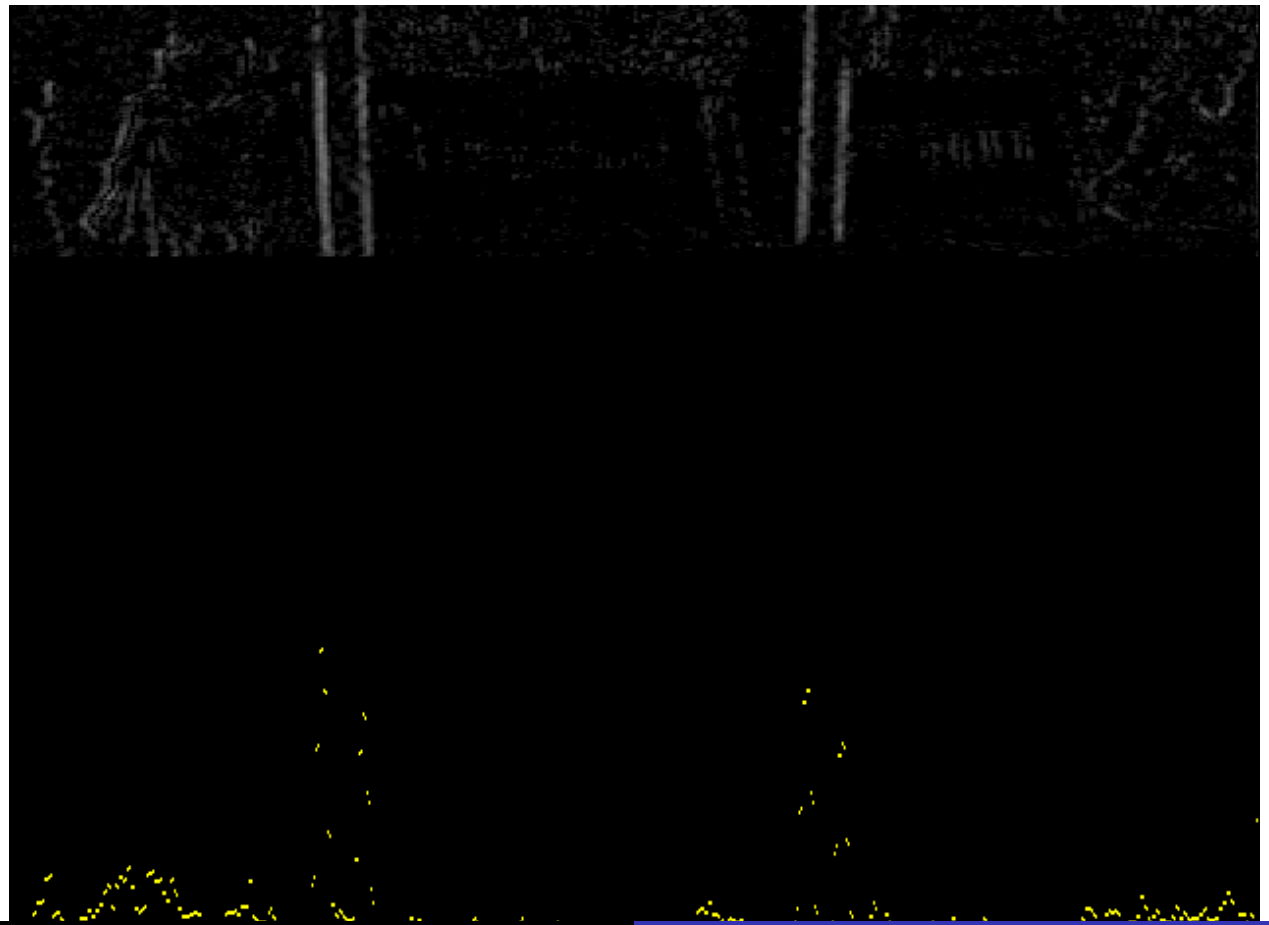
- wo das Tor suchen?
- über der Feldgrenze -> mehrere horizontale Scanlines
- Blau-Gelb -> U-Komponente analysieren

Torerkennung

- horizontale Differenzen benachbarter Pixel berechnen

Torerkennung

- Differenzen aufsummieren

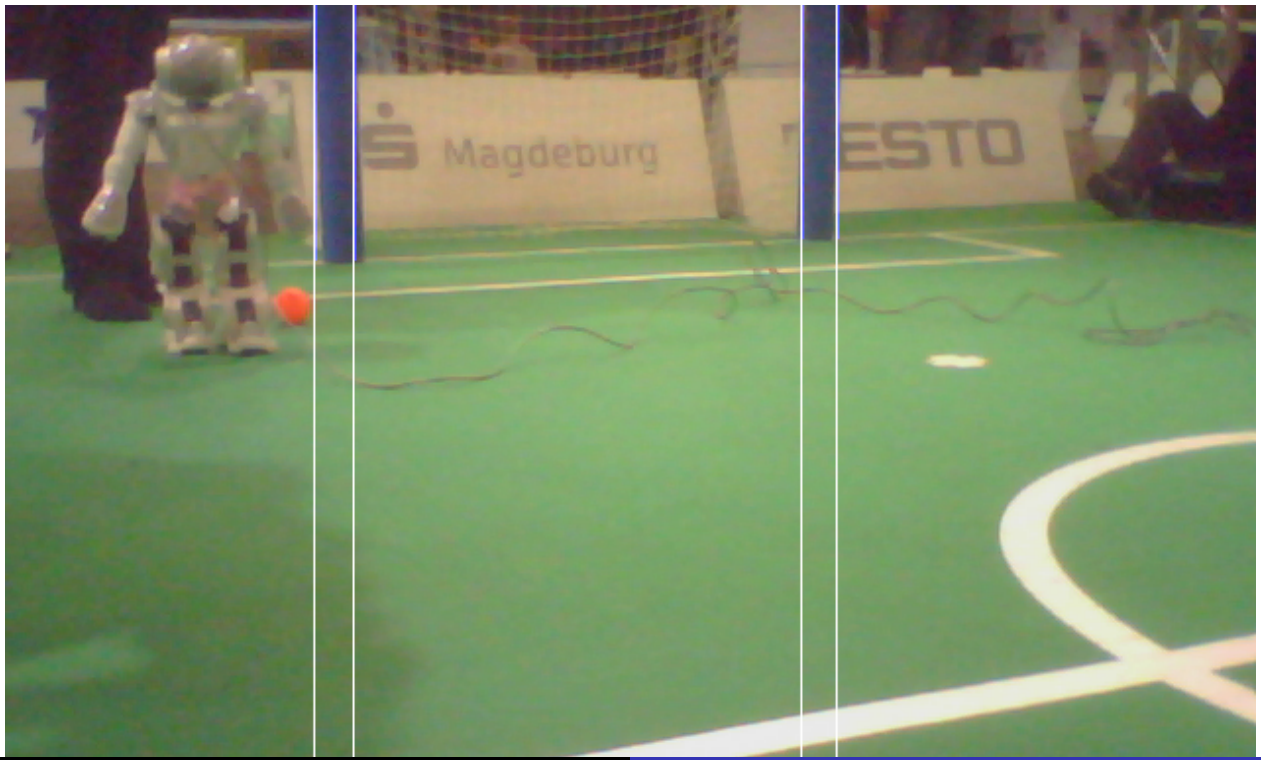


Video Torerkennung

- goalDetection_edgefilter.avi :)

Torerkennung

- non-maximum-suppression, um Torkandidaten zu finden
- Filterung durch Gradientenrichtung, Torpfostenposition, Verhältnis Höhe/Breite



Performance

Algorithmus	Zeit in ms
Feldfarben-Erkennung	< 1 ms
Liniensegment-Erkennung	5-10ms
Linienfusion	1-18ms
Ballerkennung	1-2ms
Torerkennung	3ms
Memcopy 640x480 YCbCr	12ms

- Gesamtzeit (noch) über 33ms, reicht nicht ganz für 30 fps
- Darum: Beim Balltracking keine Linien/Torerkennung

Testdatenbank

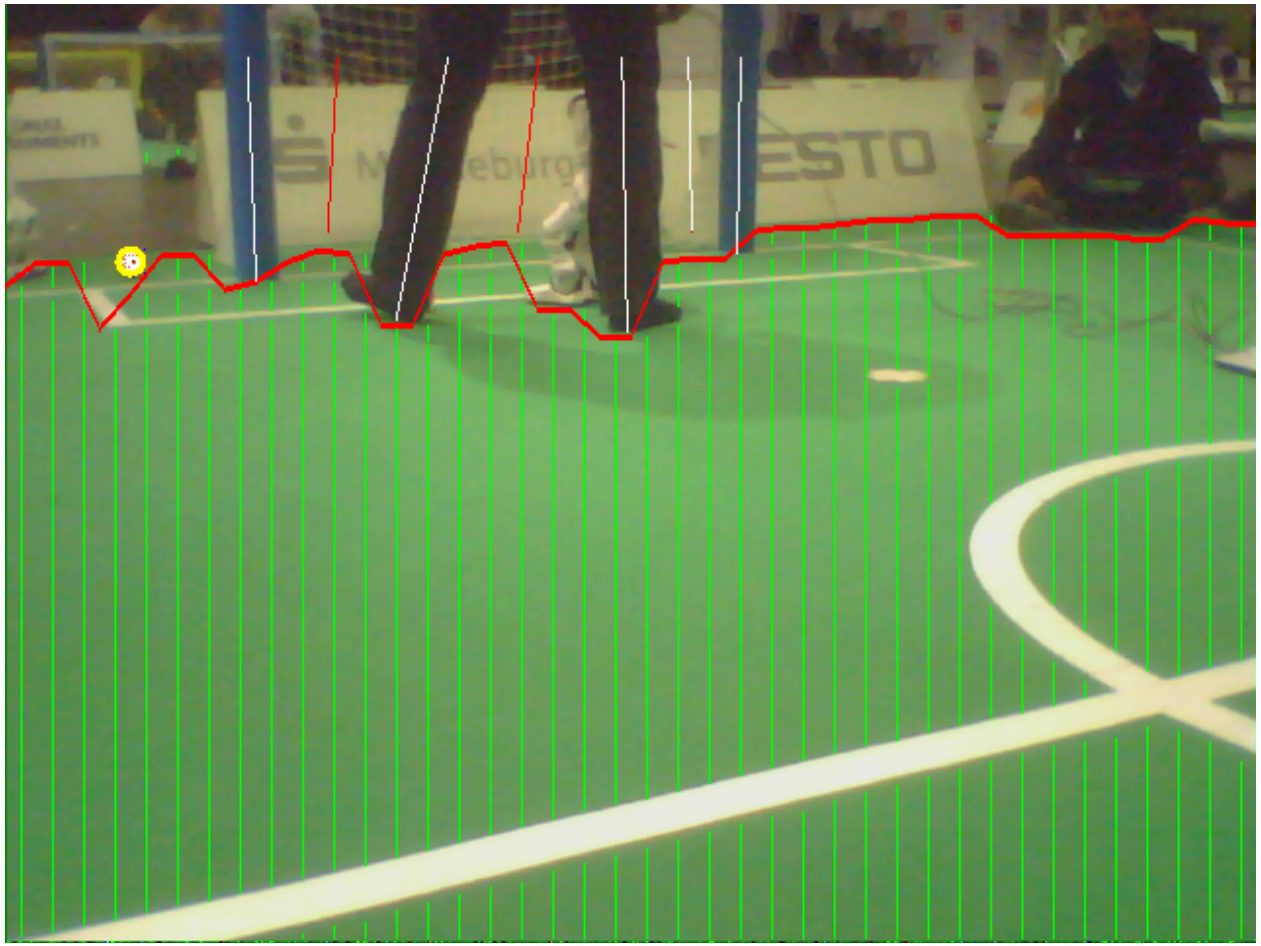
- 600 Bilder mit Information über Ballposition, Torposition, Feldgrenzen und Horizont
- Automatische Tests möglich

Auswertung

Algorithmus	positiv	falsch positiv
Ballerkennung	98,7%	0%
Torerkennung	95,1%	4%

- In Realität: Ballerkennung erkennt trotzdem hin und wieder nicht existente Bälle
- Torerkennung: 4% falsch positiv sind fast ausschließlich Schiedsrichterhosen :)

Hosen



Ende des Vortrages

Fragen?! :)