

DBV-Projekte SS 2019
Themen zum Team HTWK Smart Driving

Inhaltsverzeichnis

1	Dynamische Binarisierung (vergeben)	3
2	Detektion und Erkennung von Verkehrszeichen (vergeben)	3
3	Detektion von Fahrzeugen (vergeben)	3
4	Bestimmung der Orientierung von Fahrzeugen	3
5	Detektion von Blinklichtern im Bild (vergeben)	3
6	Detektion von Bremslichtern im Bild (vergeben)	3
7	Detektion von Personen (vergeben)	4
8	Perspektivische Entzerrung / Inverse Perspective Map	4
9	Segmentierung des Fahrbahnbereiches (vergeben)	4
10	Extraktion der rechten und linken Egospurmarkierung (vergeben)	4
11	Detektion von Kreuzungen (vergeben)	4
12	Detektion von Pylonen (vergeben)	5
13	Schätzen von Entfernung zu einem Referenzobjekt (vergeben)	5
14	Detektion von Parklücken (vergeben)	5
15	Erkennung von Fluchtpunkt und Horizontlinie (vergeben)	5
16	Objekterkennung mit Hough Forests	5
17	Stixel-basierte Objekterkennung	5
18	Bewegungserkennung aus Bildfolgen	6
19	Objektzuordnung in Bildfolgen	6

Organisatorisches

Jede Aufgabe wird in einem Team aus 2 Studierenden bearbeitet.

21 (MIM) + 8 (INM) Studierende ~ 15 Gruppen

Gruppenverteilung und Aufgaben-Zuordnung am 28.5.2019

Bitte bis dahin Gruppen bilden und Zuordnungen absprechen:

- Studierende → Gruppe
- Gruppe → Thema

Teilaufgaben / Vorgehen

- Recherche nach bekannten Ansätzen zur Lösung der Aufgabe
- Beispielbilder für alle Projekte gibt es unter <https://www.imn.htwk-leipzig.de/~schwarz/lehre/ss19/dbv/projekte19>.
Bei Bedarf können Sie selbst noch zusätzliche Bilder mit unseren Modellfahrzeugen erstellen.
Ansprechpartner: Leo Binder
- Umsetzung als ImageJ-Plugin
(Abnahme auf Pool-Rechner)
- Präsentationen (Ansätze, Entwurf, Umsetzung)
- Dokumentation

zusätzliche Verantwortlichkeiten (eine je Team-Mitglied):

- Präsentationen (Zwischenstand, Abschluss)
- Vorführungen (Zwischenstand, Abschluss) und Dokumentation

Voraussetzung zur Prüfungszulassung

für alle Studenten der Gruppe:

1. funktionierendes ImageJ-Plugin,
welches die gestellte Aufgabe erfüllt
Aufruf von Linux-Kommandozeile auf Pool-Rechner,
sinnvolle Kommandozeilen-Parameter
(ohne Nutzer-Interaktion)
2. Dokumentation: je ein Student verantwortlich für
 - Abschluss-Präsentation in Vorlesung am 4.7.2019 (vorletzte Woche)
Inhalt: Aufgabe, Ideen aus Literatur, Methoden, Entwurf, (Zwischen-)Ergebnisse
jede Gruppe 10 min Vortrag
 - Plugin-Vorführungen und Nutzer- und Entwickler-Dokumentation (ca. 1-3 Seiten),
erste Version muss zur Zwischenstand-Vorführung vorliegen:
 - Zwischenstand in den Praktika in KW 25 (20./21.6.2019)
 - Abnahme der Plugins incl. Dokumentationen in den Praktika in KW 26 (27./28.6.2019)

1 Dynamische Binarisierung (vergeben)

Eingabe: Bild (RGB)

Ausgabe: Binärbild mit optimal sichtbaren Fahrbahnl
linien
(weiß, alles andere schwarz)

Anzeige: Bild mit Binärmaske

Problem: verschiedenen Lichtbedingungen, lokale Schatten

2 Detektion und Erkennung von Verkehrszeichen (vergeben)

Eingabe: Bild (RGB)

Ausgabe: (ggf. leere) Liste von Bounding-Boxen aller im Bild erkannten Verkehrszeichen, je mit Verkehrszeichen-Typ

Anzeige: Bounding-Boxen mit Typim Overlay

3 Detektion von Fahrzeugen (vergeben)

Eingabe: Bild (RGB)

Ausgabe: (ggf. leere) Liste von Bounding-Boxen aller im Bild erkannten Fahrzeuge

Anzeige: Bounding-Boxen im Overlay

4 Bestimmung der Orientierung von Fahrzeugen

Eingabe: Bild (RGB)

Ausgabe: Liste von Paaren (Bounding-Box, Orientierung)

Anzeige: Bounding-Boxen mit erkannten Orientierungen im Overlay

5 Detektion von Blinklichtern im Bild (vergeben)

Eingabe: Bild (RGB)

Ausgabe: Liste von Paaren (Bounding-Box, aus / an) für jedes erkannte Blinklicht

Anzeige: Bounding-Boxen mit erkanntem Zustand im Overlay

6 Detektion von Bremslichtern im Bild (vergeben)

Eingabe: Bild (RGB)

Ausgabe: Liste von Paaren (Bounding-Box, aus / an) für jedes erkannte Bremslicht

Anzeige: Bounding-Boxen mit erkanntem Zustand im Overlay

7 Detektion von Personen (vergeben)

Eingabe: Bild (RGB)

Ausgabe: Liste von Bounding-Boxen für jede erkannte Person (Puppe)
Unterscheidung zwischen Erwachsenen- und Kinderpuppen

Anzeige: Bounding-Boxen mit erkanntem Personentyp im Overlay

8 Perspektivische Entzerrung / Inverse Perspective Map

Eingabe: Bild (RGB) und Referenzmodell (Karte)

Ausgabe: Transformationsmatrix zur Abbildung auf Referenzmodell

Anzeige: transformiertes Bild

9 Segmentierung des Fahrbahnbereiches (vergeben)

(vom Egofahrzeug befahrbarer Bereich)

- Segmentierung der Aufnahme
- Kennzeichnung der Fahrspuren (Polygon)

Eingabe: Bild (RGB)

Ausgabe: Mittellinie der Egospur im Bild (geeignetes Format definieren)

Anzeige: Overlay mit farbiger Markierung aller erkannten Fahrspuren mit Hervorhebung der Egospur

10 Extraktion der rechten und linken Egospurmarkierung (vergeben)

(line growing- bzw. kantenbasierter Ansatz)

Eingabe: Bild (RGB)

Ausgabe: Randmarkierungen der Egospur (geeignetes Format definieren)

Anzeige: Overlay mit farbiger Markierung aller erkannten Fahrspurbegrenzungen mit Hervorhebung der rechten und linken Begrenzung der Egospur

11 Detektion von Kreuzungen (vergeben)

Eingabe: Bild (RGB)

Ausgabe: Liste von Bounding-Boxen aller erkannten Kreuzungen und Abzweigungen
mit Angabe von Kreuzungstyp und Orientierung,
(nächste Kreuzung auf der Egospur soll erster Eintrag sein)

Anzeige: Overlay mit farbiger Markierung aller erkannten Kreuzungen

12 Detektion von Pylonen (vergeben)

Eingabe: Bild (RGB)

Ausgabe: Liste von Bounding-Boxen aller erkannten Pylone

Anzeige: Overlay mit farbiger Markierung aller erkannten Pylone (Bounding-Box)

13 Schätzen von Entfernung zu einem Referenzobjekt (vergeben)

Eingabe: Bild (RGB), Maße Referenzobjekt (Pylon)

Ausgabe: Entfernung zum Objekt

Anzeige: Overlay mit farbiger Markierung des Referenzobjektes (Bounding-Box, Abstand)

14 Detektion von Parklücken (vergeben)

Eingabe: Bild (RGB)

Ausgabe: Liste von Paaren (Bounding-Box, frei / belegt) aller erkannten Parklücken

Anzeige: Overlay mit Bounding-Boxen aller und zusätzlicher Markierung aller freien Parklücken

Problem: Verdeckung

15 Erkennung von Fluchtpunkt und Horizontlinie (vergeben)

Eingabe: Bild (RGB)

Ausgabe: Fluchtpunkt und Horizontlinie (geeignetes Format definieren) und Bild mit oberhalb der Horizontlinie gelöschtem (irrelevanten) Bereich

Anzeige: Overlay mit farbiger Markierung von Fluchtpunkt und Horizontlinie

Problem: Kurven, schräge Horizontlinie

16 Objekterkennung mit Hough Forests

(erfordert Einarbeitung)

Eingabe: Bild (RGB)

Ausgabe: Hough Forest

Anzeige: Overlay mit Bounding-Boxen und Typ jedes Objektes

17 Stixel-basierte Objekterkennung

(erfordert Einarbeitung)

Eingabe: Bild (RGB)

Ausgabe: Stixel-Darstellung der Objekte

Anzeige: Overlay mit markierten Objekt-Stixeln

18 Bewegungserkennung aus Bildfolgen

(erfordert Einarbeitung)

Eingabe: zwei oder mehrere Bilder (RGB) von einem Objekt an verschiedenen Positionen

Ausgabe: (relative) Bewegungsrichtung des Objektes

Anzeige: letztes Bild mit Overlay mit eingezeichneten Richtungsvektoren

19 Objektzuordnung in Bildfolgen

(erfordert Einarbeitung)

Eingabe: mehrere Bilder, die je mehrere Objekte zeigen

Ausgabe: Liste von Folgen von Bounding-Boxen (Positionen in der Bildfolge), je mit Objekttyp

Anzeige: letztes Bild mit allen Bounding-Boxen (verschiedene Farben für verschiedene Objekte)