

Herzlich Willkommen

Projektphase 2
Camp2Code



Überblick zur Projektphase

Woche 1:

Implementierung einer Fahrbahnerkennung mittels **klassischer Methoden der Bilderkennung**

Woche 2:

Implementierung einer Fahrbahnerkennung mittels **Neuronaler Netze**

Materialien:

Kamera für den Raspberry Pi



Überblick zur Projektphase


- **In jeder Gruppe/Team soll jeweils gemeinsam eine Software entwickelt werden.**
 - Der Quellcode soll angemessen dokumentiert sein.
 - Anwenderfreundliches Nutzerinterface.
- **Finale Präsentation** der Ergebnisse je Gruppe
- **Regelmäßige täglich Treffen** aller Teilnehmer (Progress-Meeting)
- Optionale Sourcecodeverwaltung mit Git
- Optional Nutzerinterface mittels Dash

Camp2Code

Agenda Projektphase 2 – Woche 1

	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
Vormittag	08:30 Uhr <ul style="list-style-type: none"> ▪ Begrüßung ▪ Vorstellung des Lastenhefts 09:30 Uhr <ul style="list-style-type: none"> ▪ Download Repository ▪ Inbetriebnahme RPi ▪ (Softwareinstallation) ▪ Anschluss der Kamera ▪ Test der Kamera 	08:30 Uhr <ul style="list-style-type: none"> ▪ Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> ▪ Funktionen OpenCV 09:30 Uhr <ul style="list-style-type: none"> ▪ Eigenständige Bearbeitung der Aufgaben im Lastenheft 	08:30 Uhr <ul style="list-style-type: none"> ▪ Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> • Progress-Meeting 09:30 Uhr <ul style="list-style-type: none"> ▪ Eigenständige Bearbeitung der Aufgaben im Lastenheft 	08:30 Uhr <ul style="list-style-type: none"> ▪ Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> • Progress-Meeting 09:30 Uhr <ul style="list-style-type: none"> ▪ Eigenständige Bearbeitung der Aufgaben im Lastenheft 	08:30 Uhr <ul style="list-style-type: none"> ▪ Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> • Progress-Meeting 09:30 Uhr <ul style="list-style-type: none"> ▪ Eigenständige Bearbeitung der Aufgaben im Lastenheft
	12:00 – 13:00 Uhr Mittagspause	12:00 – 13:00 Uhr Mittagspause	12:00 – 13:00 Uhr Mittagspause	12:00 – 13:00 Uhr Mittagspause	12:00 – 13:00 Uhr Mittagspause
Nachmittag	13:00 Uhr <ul style="list-style-type: none"> ▪ Eigenständiges Arbeiten in Gruppen (Lastenheft, Projektplanung) 14:00 Uhr <ul style="list-style-type: none"> • Progress-Meeting-Planung 	13:00 Uhr <ul style="list-style-type: none"> ▪ Eigenständige Bearbeitung der Aufgaben im Lastenheft 14:00 Uhr <ul style="list-style-type: none"> • Progress-Meeting 	13:00 Uhr <ul style="list-style-type: none"> ▪ Eigenständige Bearbeitung der Aufgaben im Lastenheft 14:00 Uhr <ul style="list-style-type: none"> ▪ Progress-Meeting 	13:00 Uhr <ul style="list-style-type: none"> ▪ Eigenständige Bearbeitung der Aufgaben im Lastenheft 14:00 Uhr <ul style="list-style-type: none"> • Progress-Meeting 	13:00 Uhr <ul style="list-style-type: none"> ▪ Eigenständige Bearbeitung der Aufgaben im Lastenhefts 14:00 Uhr <ul style="list-style-type: none"> ▪ Zusammenfassung der Woche ▪ Feedbackrunde

Die ersten Schritte der Projektphase 2 – Woche 1

- 1. Download Repository:** https://gitlab.com/u4i/camp2code/-/tree/project_phase_2/
oder Wechsel auf Branch *project_phase_2*
 - Anleitung/Anleitung_Projektphase2.pdf
 - Lastenheft/Lastenheft_Projektphase2.pdf
 - Software (basisklassen_cam.py, basisklassen.py, basecar.py)
 - Opencv/Demo_OpenCV.ipynb

Alternative zur eigenen
Implementierung der Projektphase 1
- 2. Inbetriebnahme und Test des RPi mit Remote Zugriff**
 - Eventuell testen der Klasse BaseCar aus Projektphase 1
- 3. Anschluss der Kamera am RPi und Test** (Anleitung_Projektphase2.pdf)
- 4. Studium und Vorstellung des Lastenheftes** (Lastenheft_Projektphase2.pdf)
- 5. Studium Klasse Camera (basisklassen_cam.py)**
 - Argumente des Konstruktors?, Methoden? (siehe auch Demo_camera.py)
- 6. Projektplanung in den Gruppen**
 - Siehe nächste Seite

Die ersten Schritte der Projektphase 2 – Woche 1

Projektplanung in den Gruppen

- Aufgabenteilung in den Teams
- Kanbanboard? (Git)
- Sourcecodeverwaltung? (Git?)
- Wie soll eine neue Klasse aussehen, welche sich von BaseCar ableitet und die Kamera integriert?
 - Attribute?, Methoden?
- Ideen zur Fahrspurerkennung?
 - ohne deren praktische Umsetzung zu betrachten
- Wie soll die verschiedenen Fahrspurerkennung in die Software integriert werden?

Kurze Vorstellung der Planung und Ideen

OpenCV – Open Computer Vision Library

Wissenswertes:

- Open Source Library in C++
- APIs für u.a. Python/Java etc.
- Anwendungsorientiert und geschwindigkeitsoptimiert
- Beinhaltet verschiedene Algorithmen
 - Von einfachen Algorithmen bis zur Einbindung komplexer ML-Algorithmen
 - Einfache Kantenerkennung
 - komplexes Deep Learning für Objekterkennung
- **Einführung in ausgewählte Funktionen von OpenCV** (Demo_OpenCV.ipynb)



Erste Schritte der Projektphase 2 - Woche 2

- **Projektplanung in den Gruppen**

- Teilaufgaben
 - Zusammenstellung der Trainingsdaten (Image Augmentation?)
 - Erstellung einer Software für das Training (Das Training muss nicht auf dem RPi durchgeführt werden!)
 - Integration in des trainierten CNN in die Software des Modellautos
- Kanbanboard
- ...
- Präsentation der Ergebnisse Freitag

- **Nützliche Tipps/Links:**

- **Beginnt einfach** (wenig Trainingsdaten, ohne Augmentation, einfaches Neuronales Netz, etc...)
- Trainingsbilder sollten vorerst am besten als unbearbeitete Bilder gespeichert werden.
- **Testet die Qualität des trainierten Netzes** mittels eines geeigneten Gütemaßes vor der Applikation am Auto.
- Trainieren eines Neuronalen Netzes mit Keras: <https://www.tensorflow.org/tutorials/images/cnn>
- Speichern und Laden von Kears-Modellen: https://www.tensorflow.org/guide/keras/save_and_serialize
- Image Augmentation: https://www.tensorflow.org/tutorials/images/data_augmentation

Camp2Code

Agenda Projektphase 2 – Woche 2

	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
Vormittag	08:30 Uhr <ul style="list-style-type: none"> Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> Progress-Meeting Überblick Woche 2 	08:30 Uhr <ul style="list-style-type: none"> Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> Progress-Meeting 09:30 Uhr <ul style="list-style-type: none"> Eigenständige Bearbeitung der Aufgaben im Lastenheft 	08:30 Uhr <ul style="list-style-type: none"> Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> Progress-Meeting 09:30 Uhr <ul style="list-style-type: none"> Eigenständige Bearbeitung der Aufgaben im Lastenheft 	08:30 Uhr <ul style="list-style-type: none"> Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> Progress-Meeting 09:30 Uhr <ul style="list-style-type: none"> Eigenständige Bearbeitung der Aufgaben im Lastenheft 	08:30 Uhr <ul style="list-style-type: none"> Stand-up-Meeting der Gruppen 09:00 Uhr <ul style="list-style-type: none"> Progress-Meeting 09:30 Uhr <ul style="list-style-type: none"> Eigenständige Bearbeitung der Aufgaben im Lastenheft
	12:00 – 13:00 Uhr Mittagspause	12:00 – 13:00 Uhr Mittagspause	12:00 – 13:00 Uhr Mittagspause	12:00 – 13:00 Uhr Mittagspause	12:00 – 13:00 Uhr Mittagspause
Nachmittag	13:00 Uhr <ul style="list-style-type: none"> Eigenständige Bearbeitung der Aufgaben im Lastenheft 14:00 Uhr <ul style="list-style-type: none"> Progress-Meeting 	13:00 Uhr <ul style="list-style-type: none"> Eigenständige Bearbeitung der Aufgaben im Lastenheft 14:00 Uhr <ul style="list-style-type: none"> Progress-Meeting 	13:00 Uhr <ul style="list-style-type: none"> Eigenständige Bearbeitung der Aufgaben im Lastenheft 14:00 Uhr <ul style="list-style-type: none"> Progress-Meeting 	13:00 Uhr <ul style="list-style-type: none"> Eigenständige Bearbeitung der Aufgaben im Lastenheft 14:00 Uhr <ul style="list-style-type: none"> Progress-Meeting 	13:00 Uhr <ul style="list-style-type: none"> Präsentation der Ergebnisse der Gruppen 15:00 Uhr <ul style="list-style-type: none"> Feedbackrunde

Beispiel-CNN mit Python/Keras

Diese Netz ist ein Empfehlung für die Bildgröße 480x640 pixel. Eine andere Architektur kann auch gut funktionieren.

```
# Keras Sequential API
model = tf.keras.Sequential(name='BeispielModell')

# Convolution Layers
# elu: Exponential Linear Unit, similar to leaky Relu
model.add(Conv2D(24, (5, 5), strides=(2, 2), input_shape=(100, 200, 3), activation='elu'))
model.add(Conv2D(36, (5, 5), strides=(2, 2), activation='elu'))
model.add(Conv2D(48, (5, 5), strides=(2, 2), activation='elu'))
model.add(Conv2D(64, (3, 3), activation='elu'))
model.add(Dropout(0.2)) # more robustness
model.add(Conv2D(64, (3, 3), activation='elu'))

# Fully Connected Layers
model.add(Flatten())
model.add(Dropout(0.2)) # more robustness
model.add(Dense(100, activation='elu'))
model.add(Dense(50, activation='elu'))
model.add(Dense(10, activation='elu'))

# Output Layer: turning angle
model.add(Dense(1))
```