KI ABSICHERUNG
Safe AI for Automated Driving

KI Absicherung – Finale Ergebnissteckbriefe TP2

Version zur Veröffentlichung

Version	1.0
Editors	Dr. Thomas Stauner / BMW Group Dr. Stephan Scholz / Volkswagen AG PD Dr. Michael Mock / Fraunhofer IAIS
Projektkoordination	Volkswagen AG / Fraunhofer IAIS
Fälligkeit	31.12.2022
Erstellungsdatum	28.10.2022

Gefördert durch:
Bundesministerium für Wirtschaft und Klimaschutz
aufgrund eines Beschlusses des Deutschen Bundestages

Dokumenteninformation

Autoren

Karl Amende / Valeo Schalter und Sensoren GmbH

Marzena Franek / Robert Bosch GmbH

Nicolas Gay / DXC Luxoft

Oliver Grau / Intel Corporation

Johannes Guenther / Intel Corporation

Burkhard Guessefeld / Robert Bosch GmbH

Niels Heller / QualityMinds GmbH

Markus Huber / Accenture Song Content Germany GmbH (früher Mackevision)

Lih Ma / DXC Luxoft

Falko Matern / Robert Bosch GmbH

Michael Mlynarski / QualityMinds GmbH

Timo Sämann / Valeo Schalter und Sensoren GmbH

Rene Schuster / DFKI

Thomas Stauner / BMW Group

Bettina Stühle / QualityMinds GmbH

Reviewer

Thomas Stauner / BMW Group

Stephan Scholz / Volkswagen AG

Michael Mock / Fraunhofer IAIS

Karl Amende / Valeo Schalter und Sensoren GmbH

Johannes Guenther / Intel Corporation

Andreas Kölsch / DFKI

Bastian Knerr / QualityMinds GmbH

Kian Saemian / Accenture Song Content Germany GmbH (früher Mackevision)

Ulrich Seger / Robert Bosch GmbH

Kontakt

(in Vertretung für die Projektkoordination)

European Center for Information and Communication Technologies – EICT GmbH

EUREF-Campus Haus 13

Torgauer Straße 12-15

10829 Berlin

Germany

Email: projects@eict.de

Projektwebsite: https://www.ki-absicherung-projekt.de/

Revisionslog

Version	Datum	Kommentar	Autor	Partner
0.1	Bis 21.10.2022	Input auf Confluence	s.o.	s.o.
0.2	21.-24.10.2022	Ausspielen Word, Prüfung und Korrektur Übertrag, Strukturierung und Layout Dokument	Bert Hildebrandt	EICT
0.3	25.10.2022	Formelles Review und Layout	Frank Brauer, Dr. Nikos Papamichail	EICT
1.0	03.11.2022	Finalisierung	Dr. Nikos Papamichail	EICT

Inhaltsverzeichnis

1 AP2.1 - Toolkette für synthetische Daten aufsetzen 7

1.1 E2.1.1 Final: nur projektintern für KI Absicherung verfügbar 7
1.2 E2.1.3 Final: nur projektintern für KI Absicherung verfügbar 7
1.3 E2.1.4 Final: nur projektintern für KI Absicherung verfügbar 7
1.4 E2.1.5 Final: nur projektintern für KI Absicherung verfügbar 7
1.5 E2.1.6 Final: nur projektintern für KI Absicherung verfügbar 7
1.6 E2.1.7 Final: nur projektintern für KI Absicherung verfügbar 7
1.7 E2.1.8 Final: nur projektintern für KI Absicherung verfügbar 7

2 AP2.2 - Corner Cases 8

2.1 E2.2.1 Final: Datenmodell 8
2.1.1 Formal Classification 8
2.1.2 Description of the result 8
2.1.3 Results 8
2.2 E2.2.2 Final: Modelle und Daten inkl. Bewertung/ Ideen für Corner Cases 10
2.2.1 Formal Classification 10
2.2.2 Description of the result 10
2.2.3 Results 10
2.3 E2.2.3 Final: Charta und Dokumentation für die beobachteten Szenarien sowie daraus abgeleitete Ideen für Corner Cases 12
2.3.1 Formal Classification 12
2.3.2 Description of the result 12
2.3.3 Results 12
2.4 E2.2.4 Final: Methodik für die systematische Ableitung und Erzeugung von Corner Cases 14
2.4.1 Formal Classification 14
2.4.2 Description of the result 14
2.4.3 Results 14
2.5 E2.2.5 Final: Modell mit allen relevanten Äquivalenzklassen inkl. Metadaten 17
2.5.1 Formal Classification 17
2.5.2 Description of the result 17
2.5.3 Results 17
2.6 E2.2.6 Final: Corner Case Data Sets 20
2.6.1 Formal Classification 20
2.6.2 Description of the result 20
2.6.3 Results 20

2.7 E2.2.7 Final: Bewertete Corner Cases 23
2.7.1 Formal Classification 23
2.7.2 Description of the results 23
2.7.3 Results 23
2.8 E2.2.8 Final: Bewertete Wissensbasis 27
2.8.1 Formal Classification 27
2.8.2 Description of the result 27
2.8.3 Approach 28
2.8.4 Result 28
2.9 E2.2.9 Final: Strukturierte Anforderungen an die Datengenerierung 28
2.9.1 Formal Classification 28
2.9.2 Description of the result 29
2.9.3 Results 30
3 AP2.3 - Abstraktion von Sensorik 32
3.1 E2.3.1 Final: Definition von vier geeigneten Use Cases und Identifikation der Einflussgrößen auf ein verändertes Sensor-Setup 32
3.2 E2.3.2 Final: Spezifikation der erforderlichen synthetischen Daten 32
3.2.1 Formal Classification 32
3.2.2 Description of the result 32
3.2.3 Result 32
3.3 E2.3.3 Final: Quantifizierung der Auswirkungen der primären Einflussgrößen auf mögliche KPI-Veränderungen 34
3.3.1 Formal Classification 34
3.3.2 Description of the result 34
3.3.3 Experiments and Results 34
3.4 E2.3.4 Final: Finetuning-Ansatz inkl. Quantifizierung notwendiger Menge an Trainingsdaten 42
3.4.1 Formal Classification 42
3.4.2 Description of the result 42
3.4.3 Experiments and Results 42
3.5 E2.3.5: Final: Auf Domain-Adaptation beruhende Ansätze, mit dem die KPI-Werte erhöht werden können 44
3.5.1 Formal Classification 44
3.5.2 Description of the result 45
3.5.3 Experiments and Results 45
4 AP2.4 - Bewertung der Qualität synthetischer Daten 50

4.1 E2.4.1a Final: nur projektintern für KI Absicherung verfügbar 50
4.2 E2.4.1b Final: nur projektintern für KI Absicherung verfügbar 50
4.3 E2.4.1c Final: nur projektintern für KI Absicherung verfügbar 50
4.4 E2.4.2 Final: nur projektintern für KI Absicherung verfügbar 50
4.5 E2.4.3 Final: nur projektintern für KI Absicherung verfügbar 50
4.6 E2.4.4 Final: nur projektintern für KI Absicherung verfügbar 50
4.7 E2.4.5 Final: Wirkkettenanalysen aus der Anwendung gezielt variierter Datensätze für grenzwertige Anwendungssituationen 50
4.7.1 Formal Classification 50
4.7.2 Description of the result 50
4.7.3 Result 51
4.8 E2.4.6a Final: Guideline für die Erzeugung und Anwendung synthetischer Daten für Training und Test von KI-basierten Algorithmen für dedizierte Anwendungen 56
4.8.1 Formal Classification 56
4.8.2 Description of the result 56
4.9 E2.4.6b Final: Guideline für die Bewertung der Übertragbarkeit der Erkenntnisse von synthetischen Daten auf eine reale Anwendung 72
4.9.1 Formal Classification 72
4.9.2 Description of the result 72
5 AP2.5 - Datengenerierung und Noisy Data 73
5.1 E2.5.1 Final: nur projektintern für KI Absicherung verfügbar 73
5.2 E2.5.2 Final: nur projektintern für KI Absicherung verfügbar 73

1 AP2.1 - Toolkette für synthetische Daten aufsetzen

1.1 E2.1.1 Final: nur projektintern für KI Absicherung verfügbar

1.2 E2.1.3 Final: nur projektintern für KI Absicherung verfügbar

1.3 E2.1.4 Final: nur projektintern für KI Absicherung verfügbar

1.4 E2.1.5 Final: nur projektintern für KI Absicherung verfügbar

1.5 E2.1.6 Final: nur projektintern für KI Absicherung verfügbar

1.6 E2.1.7 Final: nur projektintern für KI Absicherung verfügbar

1.7 E2.1.8 Final: nur projektintern für KI Absicherung verfügbar

2 AP2.2 - Corner Cases

2.1 E2.2.1 Final: Datenmodell (zur Veröffentlichung)

2.1.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	*E*
Type of content	*Modell*
Classification level	*INT, LI, PU*

2.1.2 Description of the result

2.1.2.1 Motivation

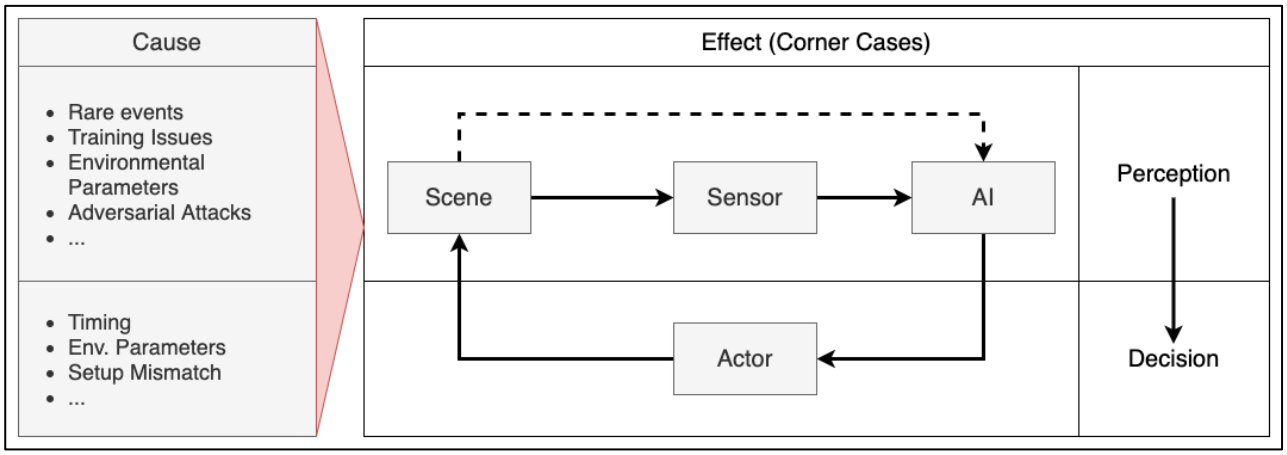
The work was based on how the Operational Design Domain is defined in the project as well as the associated basic context. Besides the creation of a first basic context, further description languages for the definition of the Operational Design Domain were proposed. The goal of the corner case taxonomy is to define different categories of corner cases in order to relate them to the domains Illumination, Occlusion, and Object Detection. Further influences on the corner case taxonomy came from the variation matrix and from a first version of the ground context. In order to make the identified corner cases usable for data production, a data model based on the ontology from TP4 was developed.

2.1.3 Results

Taxonomy

For the design of the corner cases, the first step was to build a taxonomy (ref. E2.2.5). Basically, a corner case is the result of a multitude of following circumstances - these can be erroneous inputs, environmental disturbances, training methods used, or latent disturbances of the input space ("adversarial attacks"). Only corner cases on the perception level, i.e. in the perception of the input to be processed, were considered.

Another result for the present work package was the creation of a JSON schema for the systematic request and generation of data within the request process. This schema was also further iteratively extended to include new insights and, for example, features on request. This was used, for example, within the NCAP requirement process or to request frames for the Leonberg intersection accordingly from the data producers.



Schematic Overview of the Corner Case Taxonomy

Data Model

Another result for the present work package was the creation of a data model for the systematic request and generation of data within the request process. This was used e.g. within the NCAP requirement process or to request frames for the Leonberg intersection from the data producers. The data model is the formal specification of traffic scenarios based on the model of ZwickyBoxes and the ontology from TP4.

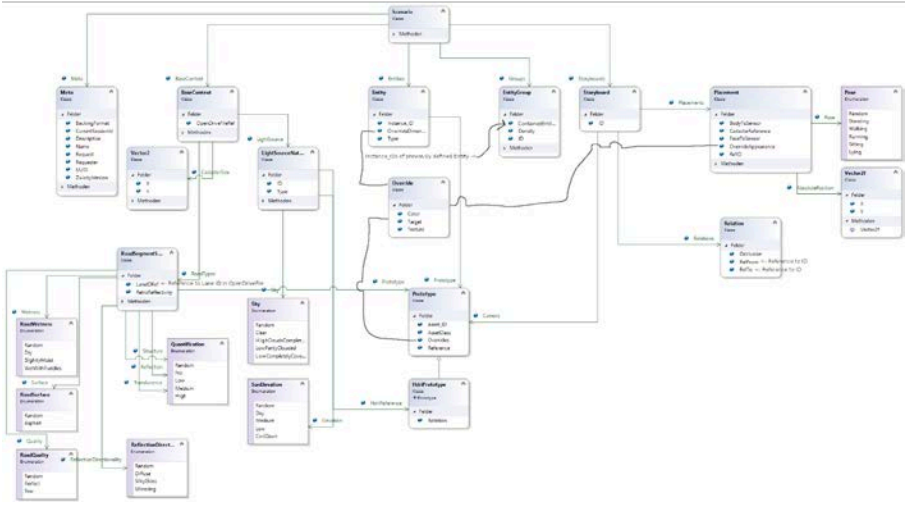
A scenario is a description of a temporal sequence of a traffic situation. Such a sequence is described by a JSON file that follows a specific JSON schema. Thus, the contribution of this WP is to create and refine the schema (which can describe all scenarios) and to provide appropriate software to generate scenario files.

Scenario files can be used as requirements documentation for the data production rendering process, which has been done to some extent. In order to generate scenario files in an automated way, a sample implementation of a scenario generator was realized in C# and provided to the project partners.

A scenario consists of 5 components: Metadata, Ground Context, Entities, Groups, and Storyboard.

- The **Metadata** contains the unique identification of the scenario, informs about the purpose and creator of the scenario, and other information that cannot be seen in the rendered sequence images.
- The **Ground Context** contains global properties such as the street textures, position and types of light sources, the size of the world represented in the scenario, etc.
- The **Entities** describe the actors (mainly cars and pedestrians) that are in the scenario. Entities reference the prototype from the asset catalog and specify changes to this prototype for this scenario.
- **Groups** describe relationships between entities, for example, to describe that entities must not move too far away from each other.

- The **Storyboard** defines each entity and each time the location where this entity should be. It should be noted that the scheme describes a finite set of time points, and thus defines a frame rate for rendering.



Corner Case Classification: Flow Chart

2.2 E2.2.2 Final: Modelle und Daten inkl. Bewertung/ Ideen für Corner Cases (zur Veröffentlichung)

2.2.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Daten*
Group/Cluster	*C*
Type of content	Spezification
Classification level	*INT, LI, PU*

2.2.2 Description of the result

2.2.2.1 Motivation

As AP2.2, we were instrumental in defining the ODD as part of P1/TP4. Several workshops focused on the specification and language boundaries related to corner cases. The basis for this was the already developed corner case taxonomy from E2.2.1, which was created through intensive scientific research as well as expert knowledge of the partners in AP2.2.

2.2.3 Results

The result of this work package was the construction of a corner-case taxonomy with various dimensions, some of which were incorporated into other work packages or are found as preliminary work there.

This structure included various dimensions:

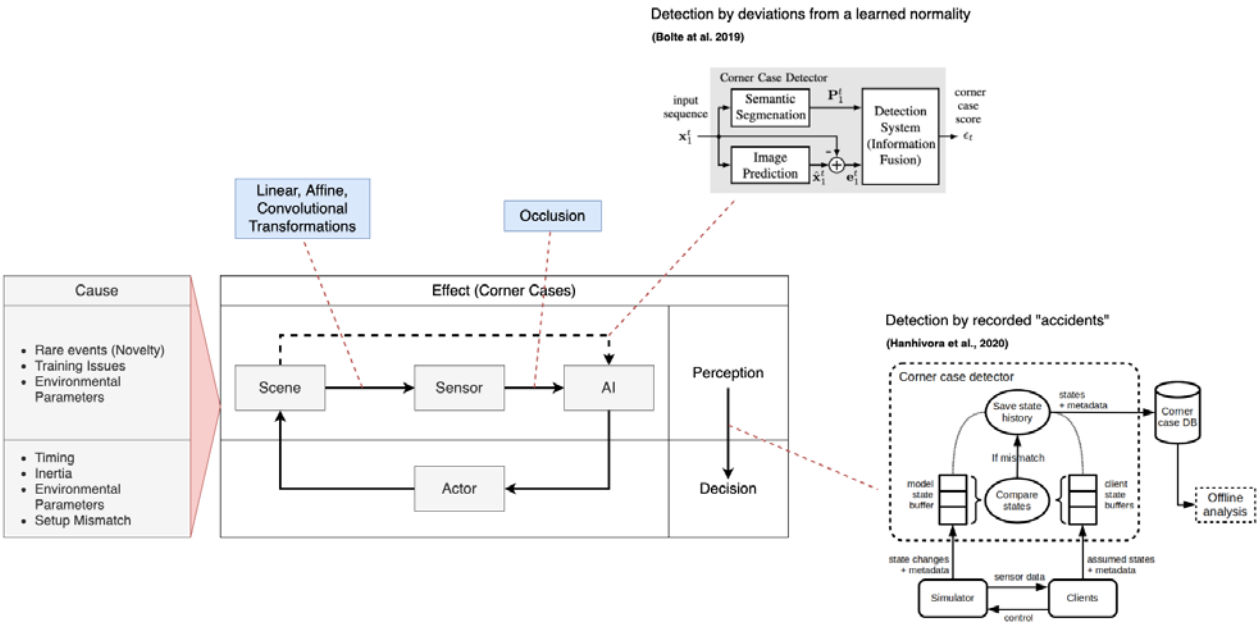
- Scene-Sensor

- Sensor-Intelligence
- Scene Intelligence (SceneAI)
- Corner Case Detectors

Scene Sensor: This category includes corner cases that occur during the recording of the environment by the sensors used and include, in addition to sensor effects, e.g., light or weather conditions with corresponding implications for the input of the AI system. In addition, **sensor intelligence** includes corner cases that occur during the actual perception, i.e., during the processing of sensor data by the AI functionality.

Scene-Intelligence (SceneAI): This category includes cross-scene corner cases that arise in a larger context (i.e., recording sequences). Consequently, it includes situational (semantic) corner cases. Independently of the KI Absicherung project, DLR captures real-world trajectory data at a research intersection in Braunschweig, Germany. It was found that corner cases can also be present in this trajectory data. Therefore, a dataset of trajectory data was prepared in parallel to the synthetic scenes and images mainly used in the project. This was not shared in the project but could be used by DLR for their analyses in the other results of AP2.2 (E2.2.4, E2.2.6).

Last but not least, another approach was to identify "**corner case detectors**" that can automatically recognize corner cases. For this purpose, approaches e.g. from Scene Design were adopted, whereby it is possible for the AI system to identify a relevant predictor for a corner case, e.g. due to the occlusion of relevant objects. Detailed documentation can be found here.



Schematic illustration of the corner case taxonomy with examples of corner cases (blue) and a corner case detector.

2.3 E2.2.3 Final: Charta und Dokumentation für die beobachteten Szenarien sowie daraus abgeleitete Ideen für Corner Cases (zur Veröffentlichung)

2.3.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	*E*
Type of content	*Methodik*
Classification level	*INT, LI, PU*

2.3.2 Description of the result

2.3.2.1 Motivation

The goal of this AP is to perform an exploratory data collection and analysis in real-world environments searching for corner cases. In this context, alternative data sources for corner cases are identified. Continental created a structured questionnaire for the identification of corner cases and DFKI detected rare human poses in available datasets using AI.

2.3.3 Results

2.3.3.1 Contribution of Continental:

One goal of the collection of corner cases was to generate a preferably complete catalog of corner cases, to ensure the completeness and coverage of different areas of the domain and ensure traceability. To create a clear structured catalog a uniformly structured description for the generation of corner cases is needed. In our case, there are also some preliminary requirements like the classification into the given taxonomy or the specific context of an urban pedestrian scenario. With the usage of a structured catalog form the decision on suitable or unsuitable corner cases should be easier and also the search for a concrete case can be much more efficiently solved.

To facilitate the systematic description of the corner cases, a structured questionnaire was created. This checklist should cover the basic preconditions for each corner case, e.g. is it part of the basic context of AI Validation (urban pedestrian crossing), how many people does the corner case include, or in which area of the taxonomy can the distinct corner case be placed in. It should also prevent corner cases to lack important information in their description.

The first part of the questionnaire is concerned with the verification of the context to ensure the collection of suitable corner cases and to avoid the collection of unnecessary or not fitting scenarios. It also covers the properties of the pedestrians and the environment because pedestrians are the central artifact for us and are one source of a possible erroneous behavior (e.g. through occlusion or clothing). The environment of the scene can be the second source of erroneous behavior of the AI system and therefore lead to a corner case situation. This can be the case if the light conditions are unfavourable and thus make a decision difficult or distinct weather conditions have not been part of training and are rare outliers (e.g. flooded area).

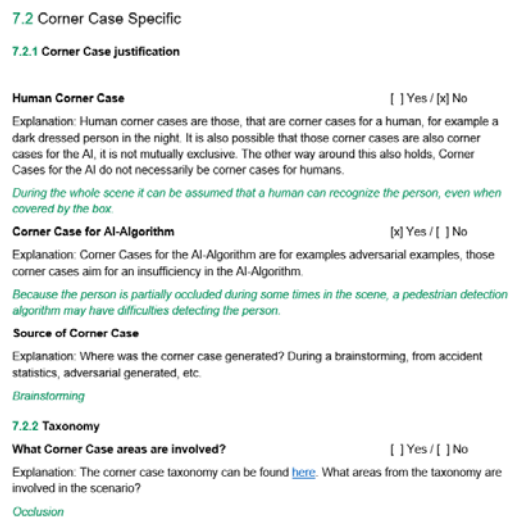
The next big part of the questionnaire is the classification of the corner cases to first, distinguish between a human corner case and an AI corner case, second, the taxonomy categorization, and third, the criticality. The distinction between human corner cases and AI corner cases is important to show that there are differences between those two. AI corner cases can for example include sensor or hardware errors or can be adversarial attacks etc. A human corner case could be a sudden event for which the reaction speed of a human is not sufficient. To make sure of a complete coverage of the taxonomy and to classify corner cases w.r.t. the taxonomy, the questionnaire also includes a taxonomy part. This can be used as tags for each corner case, where a collection of all suitable taxonomy labels (it can be more than one because there can be overlapping scenarios) is possible and subsequently the tags can be a base to further filter for specific categories (e.g. Sensor, weather condition or Perception-Decision, AI actor).

The last question of this part is an assessment of the criticality of the scenario. This follows as part of AI-related questions, like error type (false negatives/positives), assessment w.r.t. functional insufficiencies and neural network-specific safety concerns. The questionnaire also includes a collection of possible solutions and the collection of related safety-relevant metrics for the corner case entry, which can also be filled during or after development and enhances traceability for the whole process.

The link to the safety argumentation is done with the linking to hazards, system safety goals, and safety requirements from TP4.

It concludes with catalog references where related or similar corner cases can be collected.

To evaluate the usage of the template, an exemplary execution and entry were done as a sample. This includes a pedestrian carrying a box which results in occlusion of the pedestrian and possible false negatives. The questionnaire leads thoughts in a structured way e.g. by revealing differences in visual perception of human validation engineers and a typical AI function for computer vision. It also describes possible data acquisition experiments to mitigate data related safety concerns.



7.2 Corner Case Specific

7.2.1 Corner Case justification

Human Corner Case [ ] Yes / [x] No

Explanation: Human corner cases are those, that are corner cases for a human, for example a dark dressed person in the night. It is also possible that those corner cases are also corner cases for the AI, it is not mutually exclusive. The other way around this also holds, Corner Cases for the AI do not necessarily be corner cases for humans.

During the whole scene it can be assumed that a human can recognize the person, even when covered by the box.

Corner Case for AI-Algorithm [x] Yes / [ ] No

Explanation: Corner Cases for the AI-Algorithm are for examples adversarial examples, those corner cases aim for an insufficiency in the AI-Algorithm.

Because the person is partially occluded during some times in the scene, a pedestrian detection algorithm may have difficulties detecting the person.

Source of Corner Case

Explanation: Where was the corner case generated? During a brainstorming, from accident statistics, adversarial generated, etc.

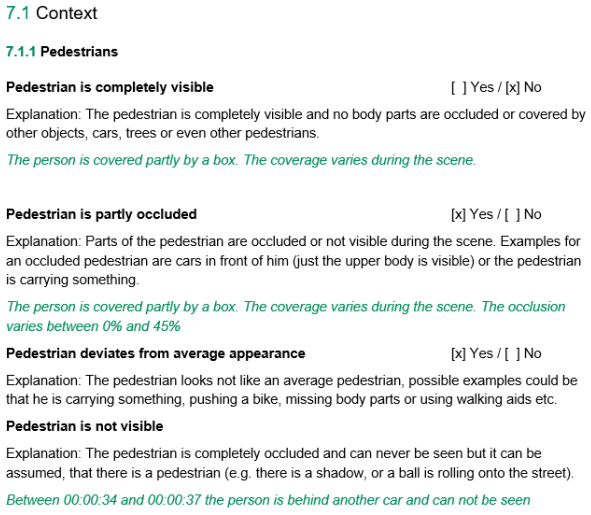
Brainstorming

7.2.2 Taxonomy

What Corner Case areas are involved? [ ] Yes / [ ] No

Explanation: The corner case taxonomy can be found here. What areas from the taxonomy are involved in the scenario?

Occlusion



7.1 Context

7.1.1 Pedestrians

Pedestrian is completely visible [ ] Yes / [x] No

Explanation: The pedestrian is completely visible and no body parts are occluded or covered by other objects, cars, trees or even other pedestrians.

The person is covered partly by a box. The coverage varies during the scene.

Pedestrian is partly occluded [x] Yes / [ ] No

Explanation: Parts of the pedestrian are occluded or not visible during the scene. Examples for an occluded pedestrian are cars in front of him (just the upper body is visible) or the pedestrian is carrying something.

The person is covered partly by a box. The coverage varies during the scene. The occlusion varies between 0% and 45%

Pedestrian deviates from average appearance [x] Yes / [ ] No

Explanation: The pedestrian looks not like an average pedestrian, possible examples could be that he is carrying something, pushing a bike, missing body parts or using walking aids etc.

Pedestrian is not visible

Explanation: The pedestrian is completely occluded and can never be seen but it can be assumed, that there is a pedestrian (e.g. there is a shadow, or a ball is rolling onto the street).

Between 00:00:34 and 00:00:37 the person is behind another car and can not be seen

Example Applications of the Template

For a systematic approach to collect, describe and catalog corner cases, a uniform description is necessary. A questionnaire like the contributed one would be suitable to also enable 'non AI

engineers' or people from other areas to collect corner cases and describe them in an easy and understandable way. The form and the length of such a description can be changed, depending on the domain and tasks, but the description needs to be connected to each corner case in the database to ensure comparability and traceability. It also can contribute to the data collection itself, by giving an overview of already covered cases or cases that lack example data.

2.3.3.2 Contribution of DFKI:

The initial idea was to analyze "in-the-wild" sequences of dangerous traffic scenes (or accidents) to derive rare situations and rare human poses for the generation of corner cases. Towards this end, a set of short sequences has been collected and LCR-Net++ (the only available 3D human pose estimator back then) has been used to estimate the human pose of pedestrians in the scene. However, LCR-Net++ turned out to be not reliable enough for this task/these scenes. Instead, the observations helped to derive limiting factors for person detection/human pose estimation.

2.4 E2.2.4 Final: Methodik für die systematische Ableitung und Erzeugung von Corner Cases (zur Veröffentlichung)

2.4.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	*E*
Type of content	Methdik
Classification level	*INT, LI, PU*

2.4.2 Description of the result

2.4.2.1 Motivation

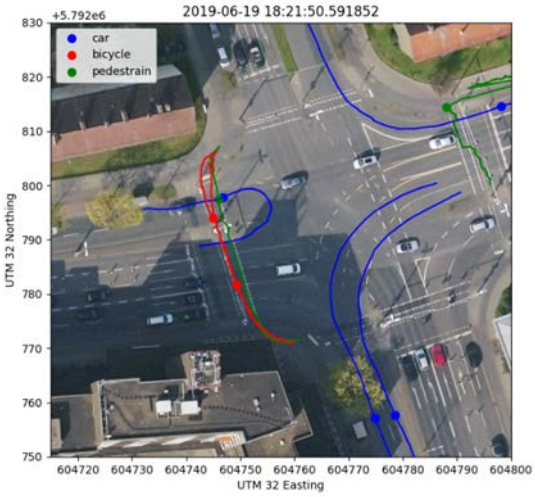
The main goal was to find and use alternative data sources for corner cases. To this end, systems for continuous delivery of corner cases were developed. These processes were automaterd as much as possible so that the number of corner cases increased over the course of the project and to increase the quality of the training data of the AI algorithms. DLR gathered and evaluated trajectory data, VW and DFKI performed various types of sensitivity analyses on various features, and QualityMinds performed an extensive literature review.

2.4.3 Results

Contribution of DLR

The aim of the research was to systematically derive and generate corner cases. For this purpose, we built onto the results from E2.2.2. and used synthetic data generation to test initial hypotheses. DLR began with a literature review of corner cases and existing taxonomies. This was used to record the scientific state of the art in order to build on it in the subsequent work. We developed a method for the automated detection of corner cases in trajectory data and tested it on the DLR trajectory data set from E2.2.2 so that traffic-related corner cases and critical situations could be identified in the dataset. Based on the taxonomy from the E2.2,5, a

rule violation was used (no-turn rule for traffic coming from the west), which was investigated on the basis of the real ground context of the Leonberg intersection. Here, vehicles cross the path of pedestrians and cyclists without permission and thus create a situation with increased relevance for the protection of driving functions. Since not all road users adhere to this rule, DLR has data on this rule violation. In this situation, the interaction between the motorized road users performing the rule violation and the pedestrians whose pathway they are crossing was specifically investigated. For this purpose, established methods of traffic conflict engineering such as the calculation of the surrogate safety measure "Post Encroachment Time" (PET) were used. This allowed 8,982 interactions to be identified. After applying a user-defined threshold, it was determined that seven of the interactions could be described as "critical". The methodological derivation of corner cases was further continued and extended, including ghost effects and pose recognition. For the safeguarding of driving functions, situations with a low negative PET are particularly relevant. In these situations, the VRU reaches the intersection first and is on the future trajectory of the vehicle at this time. The situation with the most critical negative PET value of -1.44 seconds was extracted from the data set as an example and the position data of the road users were detected in this situation as shown in the figure below. This procedure can also be implemented for other situations within the project or an even more extensive data set in order to create a data set of real traffic situational corner cases within the framework of E2.2.6.



Example of observed trajectories on a crossing.

Contribution of VW

H1: Each data point inferred into a DNN has different properties, which makes it "easy" or "hard" for the DNN to detect. By knowing these properties, one could know in advance if a DNN is going to fail or not for a specific data point. The approach for testing the above hypothesis was threefold (visual observation, image processing techniques, training a monitoring classifier). First of all, in order to find the failure criteria for a DNN, the objects the DNN failed to detect are distinguished from the successfully detected ones. This process refers to visual observation. Each group of detected/non-detected objects is compared against each other to find the possible correlation among them that might have led to such a success/failure. In a second approach, statistical analyses were conducted based on information extracted by classic image processing techniques and by employing different annotations / meta-information, such as for instance segmentation, LiDar information, etc. Based on this information generated, nine

criteria were studied which are divided into two groups: Geometrical properties (Size, Distance, Height-to-Width Ratio, Crowding of Objects, Complexity of Scenes) and Pixel Value Properties (Occlusion, Difference of Intensity, Edge Strength, VOG scores of images). Furthermore, classifiers were trained and monitored. These were trained based on the failed/passed results from the main 2D object detection model. In this way, the classifier will learn to predict if the main DNN will fail to detect the objects in an image or not. In order to analyze the correlation between each criterion and the DNN performance (box IoU), LOESS regression as a smoothing method was applied. For these classifiers, three main detection tasks have been defined accordingly: 2D object detection, Semantic instance segmentation, and Keypoint detection. It has been observed that there is a correlation between each of the aforementioned criteria and the DNN performance.

Contribution of DFKI

The results of E2.2.2 and E2.2.3 led to two hypotheses that we have investigated throughout the course of the project (H2, H3). **H2:** People of different sizes in the image are recognized differently. Especially scales that rarely occur in the training dataset can form corner cases. **H3:** Hidden skeleton keypoints are more poorly recognized by pose estimators than visible keypoints. From these hypotheses, an overarching theory was formed: Size matters. The scale of a pedestrian in the scene has an impact on the performance (i.e. extreme-scale/size → corner case). To validate the hypothesis regarding different recognition of different sizes, we have used EfficientDet as a person detector on the CityPersons and WiderPersons datasets and evaluated the performance with respect to the relative size of the bounding box to the entire image. In short, the hypothesis is confirmed. There is a significant impact of the person's relative size on the detection performance (smaller is more difficult). Also, the distribution of relative sizes on these datasets shows that really small or large persons are rare. For the other hypothesis of "Occlusion matters - Occluded keypoints (human body joints) are more difficult to detect in human pose estimation" the hypothesis was validated by an evaluation of occluded and non-occluded joints on the MPII and LSP datasets using StackedHourglassNetwork. The results show, first of all, that there is a significant discrepancy in the detection performance between occluded and non-occluded joints (i.e. severe occlusions → corner cases in terms of performance). A large part of our work was then dealt with mitigation of these phenomena by using synthetic occlusions during training. As a result, the network trained with occlusions performed better in training than the network trained without synthetic occlusions. Interestingly, there was not only an improvement in the hidden keypoints but also in the visible keypoints.

Contribution of QM

The definition of "corner cases" varies significantly in literature. Thus, a literature review was done on the terminology of "corner case". The identified definitions ranged from outcome-centric corner cases (e.g. detection failed, was erroneous) to input-centric definitions (e.g. multitude of combinations of wrong / rare combinations of input). Additionally, a literature review of existing methods for automatic generation and detection of corner cases was also performed. A literature review on existing examples of corner cases based on a taxonomy of traffic situations that are relevant for the assessment of communication processes between automated vehicles and other road users was presented as well.

2.5 E2.2.5 Final: Modell mit allen relevanten Äquivalenzklassen inkl. Metadaten (zur Veröffentlichung)

2.5.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Dokument*
Group/Cluster	*E*
Type of content	*Modell*
Classification level	*INT, LI, PU*

2.5.2 Description of the result

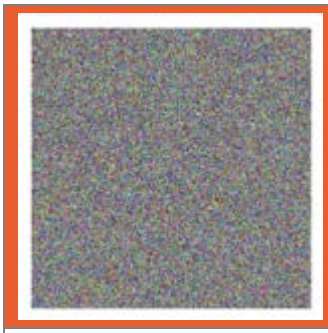
2.5.2.1 Motivation

The goal of this AP is to organize corner cases in equivalence classes, to ease data generation, corner case evaluation, and detection. Continental used a Neuron Coverage method for such a classification. QualityMinds and DLR built a Corner Case Taxonomy representing different parts of the AI-system and its surrounding. In this taxonomy, a Corner Case Class is characterized by the Interaction between system parts which cause the corner case.

2.5.3 Results

Contribution of Continental:

The collection of corner cases can be an exhausting process, and even if a lot of corner cases can be collected by hand, an automatic approach would be much faster and therefore also cheaper. In this work package, we were concerned with the classification of corner cases in equivalence classes. First paper research in the direction of corner case classification and automatic corner case generation led to neuron coverage as one method. The experiments on neuron coverage were in two different directions: 1. Can neuron coverage be used to find equivalence classes of corner cases or scenarios? 2. Can neuron coverage be used to crawl different datasets and automatically find corner cases? As a starting point, we used neuron coverage on a toy dataset for classification tasks and investigated the activations. Class-specific activations could be found and therefore we decided to also run it on the simulated data from AI Validation. For this, we ran the SSD (release 2) for bounding box generation and tracked the activations for each layer. Unfortunately, there were no specific patterns observed and we decided to go for the basic categorization of three classes: Average activation, (significant) below average activation, and (significant) above average activation. This can be used for example to decide which cases probably need human verification. We continued with experiments for the automatic collection of corner cases. There we also first used the toy dataset for a classification task and picked the outliers wrt. their deviation of the average activation.



Example of a test image that was added to the toy dataset
to be found by the automatic activation verification (it had significantly lesser activations than the average)

The reason for the lack of class-specific patterns or other patterns can be that at this time the performance of the SSD-r2 release was at around 60% mAP and therefore not comparable with the performance on the toy dataset we used for the first experiments (95% accuracy). Also, the task difference (classification vs. segmentation) can be a reason for the missing patterns. Since the three basic classes also suggested going in the direction of outlier detection, we continued with experiments for the automatic collection of corner cases. In the toy dataset, we were able to detect some outliers only by using the deviation of the average activation. This suggests that the huge amount of data available from different sources (e.g. Youtube, dashcams, computer vision challenges, ...) can be used to automatically crawl for corner cases or rare activation patterns. This does not mean including the exact data from those sources, but it may lead to new corner cases for the synthetic data generation process or new test data.

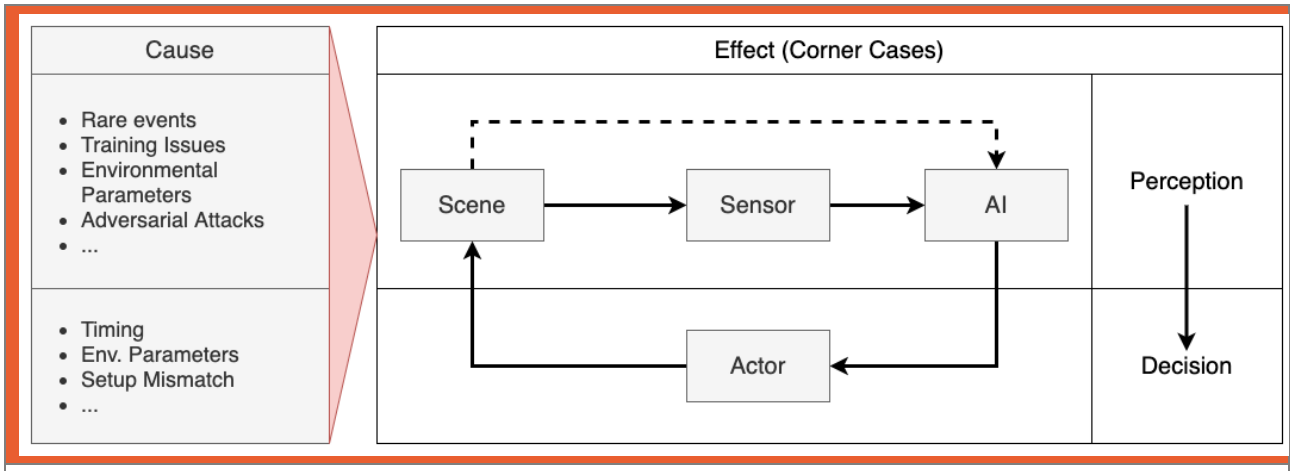
Contribution of QualityMinds (combined with input from DLR):

The creation of the corner case model is an iterative process (see Fig 1). First, we precipitate the expert knowledge of existing models and data incl. evaluation/ideas for corner cases, which led to categorization and documentation of corner case taxonomy. In parallel, we conducted an extensive literature review to identify comparable approaches for categorizing corner cases. Discussions yielding from project meetings and dedicated workshops, in which representatives of TP1, TP3, and TP4 participated, we were able to extend and refine the taxonomy by all context elements that are necessary for the derivation of corner cases. This resulted in a Corner Case Taxonomy and Corner Case Catalog (the full version of the corner case catalog including corner case images with their respective DNN inference has been attached to the appendix).

Corner Case Taxonomy:

A model has been developed (see Fig. 1) for defining corner cases. According to this model, a corner case is initially a consequence caused by a variety of circumstances. For example, *causes* (e.g. rare inputs) can lead to erroneous behavior (Corner Cases) because they do not occur in the data set at all or in too low a frequency. Not infrequently, such outliers are even deliberately removed from the data set in order to improve the performance of the model. Other causes such as environmental disturbances, the training methodology used, or latent disturbances of the input space (adversarial attacks) can lead to corner cases. In addition to perception, there are also corner cases in decision-making, but these have not been considered further. In the categorization of corner cases, various system components are taken into account, represented by nodes in the model (Fig. 1). The Scene node represents the

world/situation that has relevant features for the execution of the AI functionality. The Sensor node represents the types of sensors used to capture these features. The AI node represents the entire AI functionality, i.e. perception, monitoring, and decision-making. Finally, the Actor node represents the control of the car by the AI functionality. This model pursues the thesis that corner cases are to be sought primarily along with the connections between the nodes. For perception, these would be the connections between Scene-Sensor, Sensor-Intelligence, and Scene-Intelligence.



Recognition and generation of corner cases from the literature.

To further refine the Corner Case Taxonomy, DLR contributed to it by proposing an extension of the taxonomy to include traffic-situational corner cases. This subsection of the taxonomy was subdivided into "Rule Violation" and "Atypical behavior" so that the traffic-situational corner cases can be better distinguished. From there on, rule-violations and a-typical behaviors to further added to refine the Corner-Case-taxonomy ("traffic situations"). Additionally, the impact of various light corner cases on the performance robustness is expected to be very high which has to be proven by appropriate tests. The light-induced difference in object appearances is a matter of fast and extreme change which implies that dynamic processes have to be taken into account as well. This result has further impact and synergies to other working packages, e.g. building of the dataset in order to take into account different dynamic light situations via frame to frame variations and dark scenes with low contrast.

Corner Case Catalog: The Corner Case catalog is based on the already mentioned definition of a "'Corner Case" that has been provided in previous working packages. From there on, it has been semantically structured into "Type of failure" (e.g. distortion or semantic shift), "Impact by" (e.g. Sensor or weather) and parameters belonging to the impact (e.g. temperature or dust on lense for "sensor"). Based on this classification tree, the derivation of pictures based on this structure has been conducted and a corner case catalog has been built up. Additionally, model inference of the trained DNN (model = deeplabV3plus_kia_tranche3_bit-ts_weights.pth) was run on these corner case images yielding interesting results compiled in the distinct results report.

2.6 E2.2.6 Final: Corner Case Data Sets (zur Veröffentlichung)

2.6.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Daten*
Group/Cluster	*C*
Type of content	*Spezifikation*
Classification level	*INT, LI, PU*

2.6.2 Description of the result

2.6.2.1 Motivation

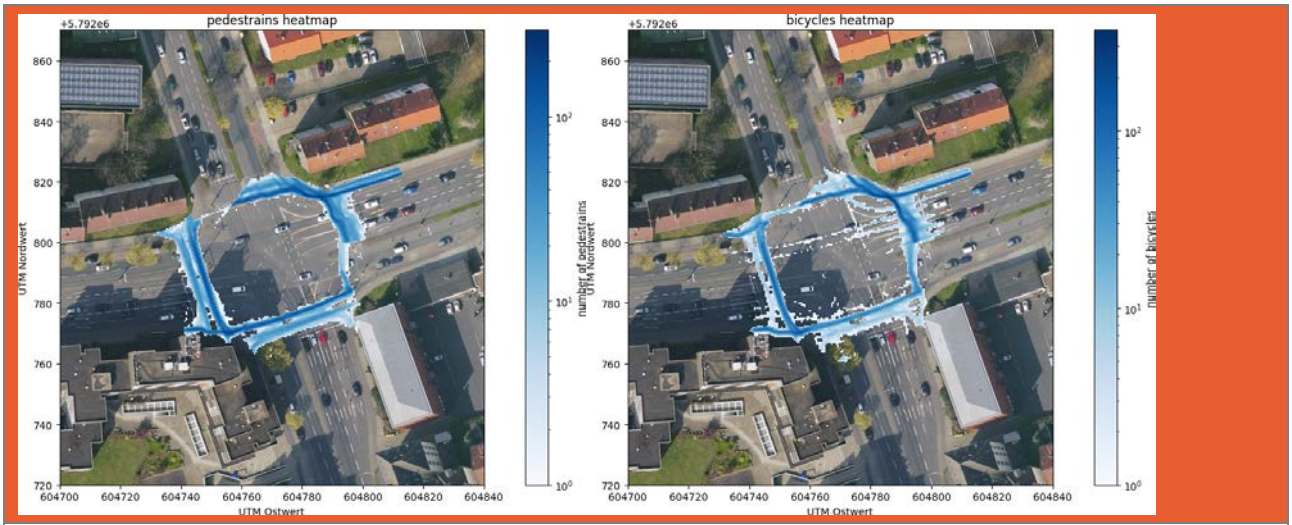
Cross-partner approach to data generation

In AP2.2, we use an iterative approach to corner case generation. The design process first results in the E2.2.5 Model with all relevant equivalence classes, including metadata, using the methods developed within E2.2.4. For this purpose, a consolidation system for the creation of corner cases in the project was developed (see E2.2.1). Thus, if the input is an abstract description of the corner cases to be created, then this is concretized using a formal language based on the ontology from AP4.1 and recorded in JSON format. This description is set as a data requirement in JIRA and refined using acceptance criteria (see E2.2.9 - Structured Requirements for Data Generation). After data generation in AP2.5, the corner case data sets for AP2.2 will be made available and applied in experiments. We have received feedback from the project partners to make the synthetic data more varied. To do this, we collected data requirements from across the project, clustered them, and used the Pairwise Testing method to derive the maximum expression of variation. This extract was then used to generate a so-called storyboard for the data request, which specifies logical sequences in the form of scenarios for the data producers.

2.6.3 Results

Contribution of DLR

To create corner case datasets, it should be determined how large the proportion of pedestrians and bicyclists in a dataset of an urban intersection is. Only by knowing which distributions are normal, abnormal distributions can be identified as such for the corner case datasets. For this purpose, real trajectory data from the DLR research intersection was again analyzed.
The first step was to calculate the general percentage of pedestrians/bicyclists on all objects in the dataset. In the studied dataset, the total number of road users varied from about 200,000-330,000 road users per week. The percentage of bicyclists ranged from 6.1-6.5%, and the percentage of pedestrians ranged from 5.2-5.8%.
Furthermore, one day of the data set was analyzed in more detail by dividing the research intersection into a grid of 0.5x0.5 meter cells and analyzing the number of different road users per cell. This allowed us to establish distribution across the intersection and identify areas where pedestrians/bicyclists were less likely to be present than others. The following figures illustrate our measurements.



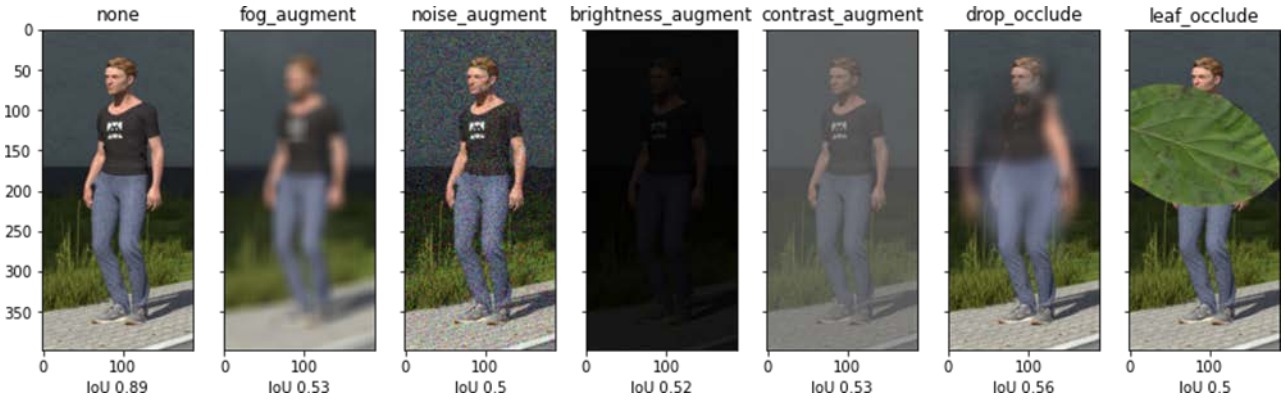
Pedestrian distribution (left) and bicycle distribution (right) at the research intersection.

Contribution of QualityMinds

One approach to generating corner case datasets is augmentations, i.e. algorithms applied to an image to change its superficial properties (example: distorting an image by statistical noise). It is to be expected that some augmentations create corner cases in the sense of the definition used in E2.2.7 (corner cases are inputs that produce an unexpectedly bad result): For example, if statistical noise in the image disturbs the AI, but this disturbance is not expected, this is a corner case according to the definition.

The following augmentation algorithms were used: Fog_augment adds a blur effect to the image with adjustable strength, implemented by the Python Automold library. Noise_augment adds normally distributed random values to the color values of the image, the variance of which can be varied to control the effect size. "Brightness_augment" and "contrast_augment" are provided by the Python PIL ImageEnhance module, each of which allows adjustment of target values (0 brightness results in a perfectly black image, and 0 contrast results in a monochrome image with the average brightness of the original). Drop_occlude and leaf_occlude are new implementations that add drop-effects or overlay the image with images of leaves. In both cases, effective placements and sizes can be adjusted, although effect size had the strongest effect on AI performance.

The following figure shows examples of augmented images. The effect size was calibrated to reduce the performance of the AI by about the same amount each time.

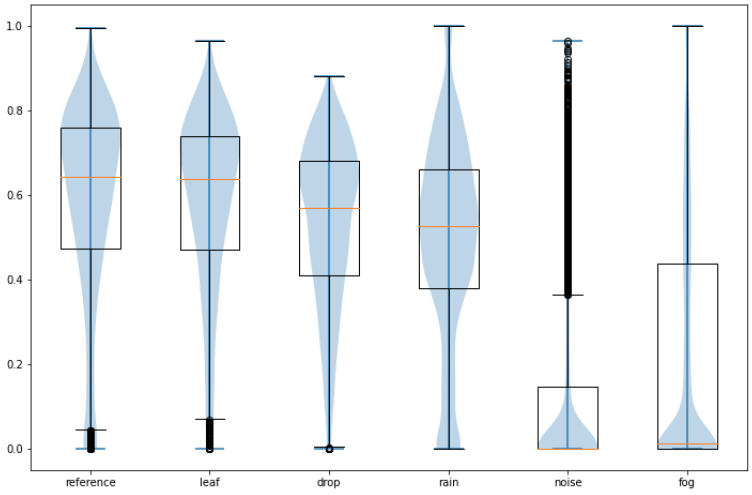


Examples of the application of augmentation: The first image from the left is the original. The captions indicate the performance of the AI on this image (IoU).

In the first run, sequences were augmented and the resulting rates of corner cases were examined using the model described in E.2.2.7. First, it was shown that all augmentation approaches are suitable for generating corner cases, and second, that they had different effects on the performance of the AI (see the following two figures).

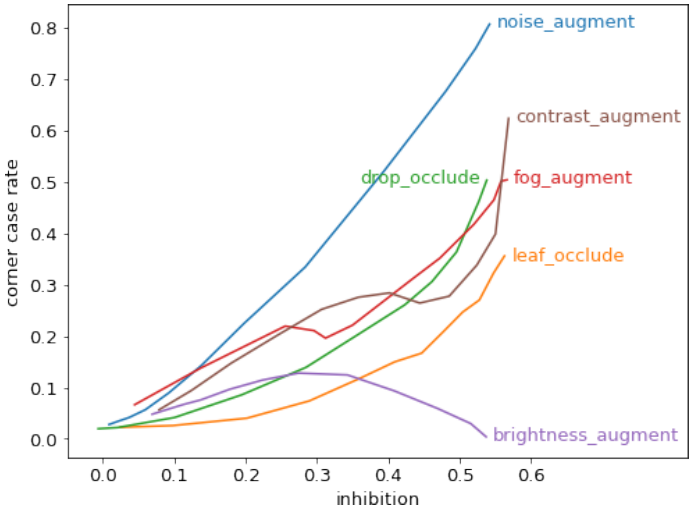
	cc_rate	odds_ratio	p_value
reference	0.023237	1.000000	1.000000e+00
leaf	0.069982	3.163005	6.885890e-31
drop	0.066760	3.006961	2.270483e-18
rain	0.061988	2.777824	2.753105e-72
noise	0.687864	92.632068	0.000000e+00
fog	0.527533	46.933299	0.000000e+00

Corner case rates (cc_rate) and risk ratios (odds_ratio) for different augmentations. The origin corner case rate (reference) is 0.02, that of noisy images (noise) is 0.68.



Performance of the AI (in IoU) for different augmentations. "reference" is the unprocessed dataset.

To correct for this effect, we have applied trial series of augmentations with increasing effect sizes. Statistical noise produces the highest corner case rates for this AI and corner case model, while leaf_occlusion produces the lowest corner case rates.



Corner case rates as effects become stronger (resulting in stronger performance inhibition).

2.7 E2.2.7 Final: Bewertete Corner Cases (zur Veröffentlichung)

2.7.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Dokument*
Group/Cluster	*D*
Type of content	*Evaluation*
Classification level	*INT, LI, PU*

2.7.2 Description of the results

2.7.2.1 Motivation

The goal of this AP is to produce machine-readable descriptions of corner cases, and to evaluate examples of such corner cases. The DLR described Corner Cases using observed pedestrian trajectories, and produced respective frames for evaluation. The DFKI performed as categorization of Corner Cases for prioritization. Valeo produced a meta data model describing corner cases. QualityMinds designed an iterative method to find corner cases based on several performance-inhibiting features.

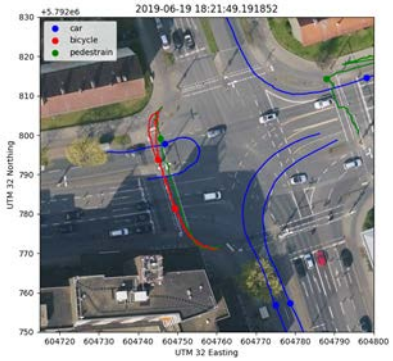
2.7.3 Results

Contribution of DLR:

The goal was to provide acquisition and provision of ground truth data (e.g. for modeling purposes or quality assessment of synthetic data).

The data on the traffic situation corner cases, which could be detected using the methodology developed in E2.2.4, was previously only available internally at DLR in the form of trajectories. With the help of our approaches, the most critical situations were transferred from the trajectories of the dynamic objects from the DLR intersection to the Leonberg intersection. This allowed the data to be translated into synthetic scenes and generated for the project. From the

scene, 107 variations were created, which are now available in tranche 7 sequence 96 frames 5464-5571. Our work thus demonstrates, how a traffic situational corner case can be synthetically recreated.

1. Original scene from the real data from DLR on the intersection in Braunschweig	2. Transfer to the Leonberg intersection: visualization of the differently requested variants	3. Visualization of the scene from the vehicle's point of view
		

Szene aus Realdaten von DLR Kreuzung in Braunschweig

Contribution of DFKI:

The goal was to identify meaningful variations and thus prioritize the corner cases. We performed a systematic analysis of the effects of variation of corner cases as well as changes in the knowledge base using the algorithms from AP1.3, AP1.4, and AP1.5.

Contribution of Valeo:

Based on the corner cases identified in AP3.5, an analysis of potential metadata describing these corner cases was performed. The identification of the corner cases is based on a difference in DNN outputs based on different modalities (LiDAR and camera). The metadata can be divided into two categories:
i) Global metadata: These include, for example, the number of lanes, the speed limit, the road type, the environment (city, highway, country road, etc.), and others.
ii) Local metadata: These are based on a visual analysis of the scene and include the object localization in the vehicle coordinate system, the present lighting conditions, the object type (car, van, pedestrian, etc), as well as low-level features computed by a dimension-reduction algorithm (DNN encoder) and the weather. Automated recording and storage and processing of local and global metadata for recordings is part of the result and can be used to record new corner cases. Work building on this was done in AP2.2.8.

Contribution of QualityMinds:

We formulated a methodology that involves developing and applying an iterative scheme to identify corner cases (see Fig 1.) The results are based on the definition of corner case as "unexpectedly poor performance of the AI". By applying the method to the KI-A data set and a semantic segmentation AI, a total of eight features were found, which can be divided into three groups: quantitative, perceptual, and situative features. To find such features, an iterative process was defined in that each iteration consists of four phases: *Exploration phase* (analysis of data set e.g. via plotting data), *hypothesis formulation phase* (specific performance constraint is assumed, e.g. instance size), *experimentation phase* (selection of concrete features and evaluation if feature is new and has significant impact) and the *result compilation phase* (applying the newly found selection rule together with all old rules to create a new corner case dataset).

Extensive occlusion analyses were performed depending on the type of occlusion (e.g. occlusion occurring due to car, vegetation, human, etc.), and the amount of occlusion. Further, time analysis was done for consecutive frames and variation of occlusion versus performance was measured, which led to the quantification of the correlation present between the two. The definition of corner cases was thus expanded to scenarios that represented unexpected behavior e.g. positive correlation between occlusion and performance. Additionally, a tool was developed to identify instances that are similar depending on selected features. Corner cases were identified as instances that were highly similar, yet with significantly large differences in performances.

2.8 E2.2.8 Final: Bewertete Wissensbasis (zur Veröffentlichung)

2.8.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Dokument*
Group/Cluster	*D*
Type of content	*Evaluation*
Classification level	*INT, LI, PU*

2.8.2 Description of the result

Valeo:

- The work was based on the subworkpackage 2.2.7 by performing an analysis of the metadata of found corner cases. The determined metadata (local and global) were incorporated into an automation process for future recordings, so that they are automatically recorded, stored and processed. With the help of the automated recording of local and global metadata, specific corner cases can be recorded during recordings. Suitable recoding scenes can be determined by analyzing the recorded metadata. Example: If the found corner cases occur more frequently at certain speeds, a certain motion blur might be the cause of the corner cases. Consequently, recordings would be performed more often at higher speeds. With this approach, a direction is given as to which criteria are important for the recordings and thus the scenes to be recorded are strongly restricted. To improve the predicitve performance of the DNNs, they are fine-tuned with the recorded data. If supervised algorithms are used, they have to be labeled beforehand. The labeling process as well as the fine-tuning and re-analysis of the DNN outputs with respect to the corner cases was outside the scope of this work package.

DFKI:

- Systematische Analyse von Effekten der Variation von Corner Cases sowie Änderungen in der Wissensbasis anhand der Algorithmen aus AP1.3, AP1.4, AP1.5. Ziel ist es sinnvolle Variationen zu identifizieren und so die Corner Cases zu priorisieren.

VW:

- Definition von Äquivalenzklassen aus Sicht der Sensitivitätsanalyse
- Spezifikation von Metadaten basierend auf den Testergebnissen der Sensitivitätsanalyse
- Anhand der Testergebnisse werden diejenigen Szenarien mit den relativ schlechteren Testergebnissen analysiert auf domänenspezifische Unterschiede zu den „besseren“ Szenarien und Gemeinsamkeiten untereinander identifiziert. In weiteren Testreihen werden Verdachtsfälle geprüft und Eigenschaften, die Corner Cases auslösen, werden entsprechend klassifiziert.

2.8.3 Approach

Die Wissensbasis wird sowohl durch die Analyse der E2.2.2 Modelle und Daten inkl. Bewertung/ Ideen für Corner Cases, des E2.2.1 Datenmodell als auch zentrales Corner Case Modells mittels Corner Case Taxonomie erstmalig erstellt. Das Modell sowie dazugehöriges Wissen sollte iterativ mit Hilfe von Hypothesen Testing überprüft und bewertet werden. Siehe Hinweise darauf im Konsolidierungssystem. Wie in E2.2.7 können derzeit keine Corner Case Daten generiert werden (höchste Priorität Trainingsdaten) und deswegen findet auch keine Bewertung der Wissensbasis. Als Workaround bedienen wir uns in den AP2.2 Experimenten derzeit eigens angelernter Netze (bspw. von den Partnern die DFKI oder Continental oder allgemein verfügbaren Datensätzen wie dem A2D2 von Audi).

2.8.4 Result

QM:

- Nutzung von Status quo-Bewertungsmethoden (Eintrittswahrscheinlichkeit, Kritikalität, Reaktion des zu testenden Systems, etc.) für die Bewertung der Corner Cases und Wissensbasis angelehnt an projektinternen KPIs
- Konzeption einer einer Bewertungsmethode und Durchführung der Bewertung nach der Datengenerierung, dem Training sowie Test auf den DNNs

→ Picture from HACE Paper

→ Rerun the CC model on new AI performance (Niels Heller Namrata Gurung)

Lessons Learned:

It was observed to be difficult to consolidate all needed requirements. Additionally, the definition of "good performance" in a standardized project-wide environment was difficult e.g. defining thresholds when only a semantic level description is present, is a bit arbitrary. It was noticed that in order to request data in the absence of a thorough Corner Case definition while simultaneously setting up DNNs for research, is rather counterintuitive. We realized that corner case identification is a continuous iterative process, wherein a solid knowledge base should be set up first. We also found discrepancies in the conception world versus the measuring world, with the absence of proper tracking which led to several repetitive analyses which could've been avoided. Additionally, the side of data-management could be improved upon significantly e.g. accessibility to data, computation of features, requesting data, decision-making on the data structure, and communication thereof to the users (changing data structure on code-level should be avoided as users build data-processing packages with the old versions of data structure).

2.9 E2.2.9 Final: Strukturierte Anforderungen an die Datengenerierung (zur Veröffentlichung)

2.9.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Dokument*
Group/Cluster	*C*

Criteria	Classification according to VHB
Type of content	*Anforderung*
Classification level	*INT, LI, PU*

2.9.2 Description of the result

2.9.2.1 Motivation

The dataset was created by iteratively producing data tranches, extending tool capabilities and assets, and moving to the next iteration while simultaneously prioritizing and refining the user's requirements. This approach was necessary to quickly implement first data tranches and allowing users to try out the data whilst still improving on data quality in the meantime. Learnings have been adapted to the following tranches accordingly to further improve data quality. Requirements have been matched with discussions within the P1-Process and users from other work packages of the project. Documentation has been conducted via confluence pages in order to collaborate further with all necessary project partners. Thus was ensured to build a data set with strong relations to actual requirements within the project and further learnings about requirements from user's perspectives were applied to improve the data continuously. This concerns slicing of scenes e.g. regarding pedestrian or vehicle distribution and instance variation, as well as unique requirements like assets which have not been included in the training data ("out-of-distribution assets") to experiment with detection for assets unknown to the neural network.

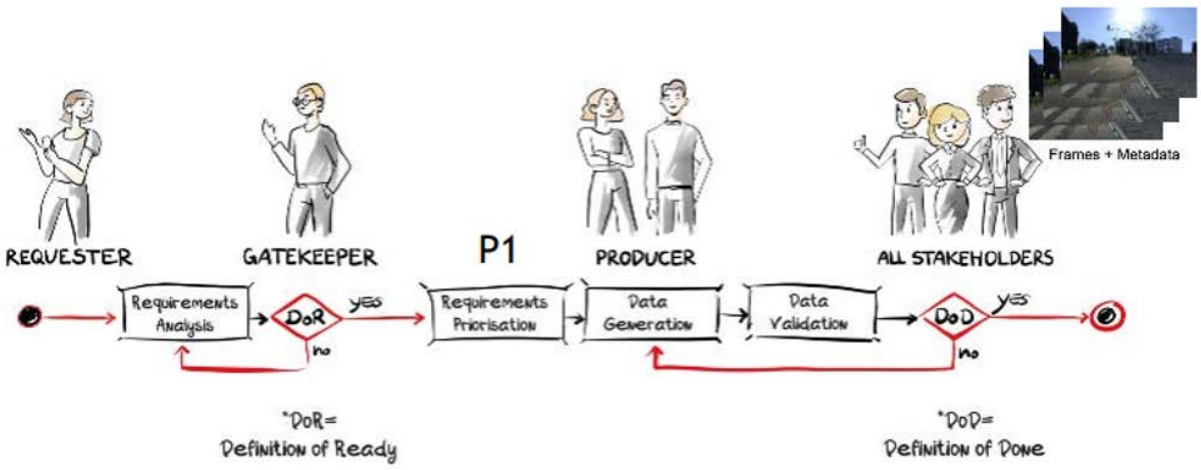


Fig. 1: Process for Requirement Engineering and publication of the respective data deliveries.

Requirements, acceptance criteria and further details have been documented via gitlab first. In June 2021, relocation of the ticketing system from gitlab to Jira has been conducted with several implications to the requirement management process, e.g. completeness of old tickets when moving from gitlab to Jira. For exact and automated data requirements, a JSON format has been developed and used within the project in order to further automate and increase the speed of the requirement process.

2.9.3 Results

The result of this approach was thirteen different data deliveries from both data producers (Mackevision and BIT-TS) split up into nine tranches with over 300.000 frames being produced. Altogether, the dataset has several different areas of focus which can be seen in Table 1.

The start phase mainly had a focus on preparing for large-scale production. Production of data has been according to the ontology that has been developed, in the project, including both public libraries as well as self-created assets - the latter to satisfy specific requirements e.g. regarding pose animation or asset attributes. Likewise, scenes have been expanded iteratively with new tool features from the different data producers to increase variance and complexity. For instance, frame-to-frame variations were introduced early in the project in order to further boost variance. These variations concerned multiple features within the scene design which are toggled accordingly to the user's needs. Toggeled features include change of pedestrian's clothing colors, presence, and orientation of pedestrians, sky, and sun position. Later tranches introduced more highlights e.g. nightly scenes according to light scene design. A full list of highlights can be seen in table 1.

Tranche	Added Features	Data Producer
1, 2	Preparatory measures for large scale data production - Tranches not part of the published dataset	BIT-TS, Mackevision
3	Introduction of HDR images for variation in lighting situations (dark to very bright)	BIT-TS
	Further ramp-up for large scale production	BIT-TS
4	Frame-to-frame variations	BIT-TS, Mackevision
	Meta information on AssetIDs	BIT-TS, Mackevision
	Bodypart segmentation	BIT TS
	Introduction of procedural sun model	Mackevision
5	Integration of sensor noise (postprocessing)	BIT TS
	Introduction of procedural clouds	Mackevision
	Ground truth for pose estimation	Mackevision
	Meta information on occlusion	Mackevision
6	Environmental effects: wetness, sun glare	Mackevision
	Out-of-distribution assets	Mackevision

Tranche	Added Features	Data Producer
	Data for different camera sensor parameters	Mackevision
7	Data generation with camera and LiDar sensor models using physically based rendering with OSPray	BIT TS
	Data for different LiDar sensor parameters	BIT TS
	Environmental effects: fog, vignetting	Mackevision
	Meta information on AnimationID	Mackevision
8	Introduction of Night scenes incl. artificial light	Mackevision
9	Focus on contrast experiments (background and instances) and introduction of new materials	Mackevision

In order to allow for easy use of the data, the data of all relevant tranches, fixes and enrichments of project partners is packaged as "Latest and Greatest" release.

3 AP2.3 - Abstraktion von Sensorik

3.1 E2.3.1 Final: Definition von vier geeigneten Use Cases und Identifikation der Einflussgrößen auf ein verändertes Sensor-Setup (zur Veröffentlichung)

is contained in E2.3.2 below.

3.2 E2.3.2 Final: Spezifikation der erforderlichen synthetischen Daten (zur Veröffentlichung)

3.2.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	*C*
Type of content	*Specification*
Classification level	*INT, LI, PU*

3.2.2 Description of the result

In order to analyse effects of modified sensor setup, data is necessary that conforms to the use cases of the modified sensor setup. This result specifies the properties of this data. In particular it specifies the different sensor setups that we use. In order to be able to study the effect of changes in the sensor setup in isolation we decided that the data for modified sensor setup is not from entirely different data sequences, but from the same base sequences. So for every parameter change, e.g. wider field-of-view (FoV), the data is generated using the same base sequences thus including the same situation, objects etc.. This allows to specifically investigate the given parameter change and get accurate insights of the effect for a given change to the performance.

3.2.3 Result

The following camera parameter variants have been requested:

- default (1.4m mounting height, 1920x1280 resolution, 60° horizontal field of view, for further details see E1.2.1 Final: Definition der Referenz Bildsensormodelle und ihrer Verbaupositionen)
- horizontal field of view of 100°
- camera mounting position +1m, i.e. 2.4m height
- lower resolution of 960x640
- lower color resolution of 8 bit (computed from default data in a post-processing step)
- greyscale (computed from default data in a post-processing step)

In order to have independent frames for training and testing two sequences are required, with at least around 2000 frames for training. The data was delivered within Tranche 6 of Mackevision, sequences 66, 67, 77-82.

Source: Mackevision

The follow lidar parameter variants have been requested:

	lidar_ideal	lidar_realistic	lidar_variant2	lidar_variant3	lidar_variant4
horizontal FoV	180°	133°	180°	180°	180°
vertical FoV	50°	10.5°, complex scan pattern	75°	25°	50°
number of APDs in group	4	4	4	4	4
number of APD groups	20	4	20	20	10
mounting position	2 variants: • rooftop, middle, with sensor pitch of -11.45° (i.e. sensor is oriented towards the ground). Lidar Ids in BIT TS Tranche 7delivery: lidar017, lidar033, lidar625 • front bumper, middle, sensor pitch of 0° Lidar Ids in BIT TS Tranche 7 delivery: lidar018, lidar034, lidar634				

Similar to the camera, we require at least around 2000 frames for training and additional frames for testing. The data for the above lidar variants was delivered in Tranche 7 of BIT TS. The drives of two different vehicles through the scene can be used to split training and test data, thereby yielding around 2000 frames for each set and each variant in sequence 1008.

Note that lidar data prior to BIT TS Tranche 7 is based on the parameters of a Velodyne HDL-64E, with the lidar sensor in the same position as the camera.

3.3 E2.3.3 Final: Quantifizierung der Auswirkungen der primären Einflussgrößen auf mögliche KPI-Veränderungen (zur Veröffentlichung)

3.3.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	*D*
Type of content	*Evaluation*
Classification level	*INT, LI, PU*

3.3.2 Description of the result

The effect of the primary influence factors from sensor setup changes as defined in E2.3.2, like changes in the resolution, are qualitatively and quantitatively analyzed. For the analysis, experiments are conducted by the project partners with different AI functions from TP1 and different sensor setups.

3.3.3 Experiments and Results

3.3.3.1 BMW experiments and results

One of our goals in AP2.3 is to evaluate the effect of sensor parameter changes on DNN performance and to examine how domain adaptation can close the domain gap and improve performance. E2.3.3 provides the intial setup. We selected the DeeplabV3 and train it with data tranche 4 with the project data split 3*. The baseline is the performance on the test set of this data split, a mIoU of 0.74. Using inference on the Mackevision sequences 77, 78 and 79, which are sensor parameter variants of sequence 66, we obtain the following performance results:

camera parameter variant	mIoU
default (cross evaluation on tranche 6 seq 66)	0.65
increased horizontal field of view of 100°	0.64
higher camera mounting height +1m	0.62
lower resolution of 960x640	0.64
lower color resolution of 8bit	0.60
greyscale	0.50
24 colors	0.52
12 colors	0.48

Except of changes in the color resolution, all other variants only have a small effect in comparison to the default performance on tranche 6 sequence 66. The performance on this

sequence is, however, considerably lower than the performance on the test set of tranche 4. This can be explained by a domain gap between tranche 4 and 6. In particular, that sequence contains more assets, more frame variations and lens flare.

The high loss for smaller color resolutions can be explained by a loss of information, but also by strongy modified appearance. Note that greyscale and 8bit color resulution both use 8bit for information encoding, but greyscape performs worse that using only 24 colors.

The small differences between the field-of-view and the sensor position variant may have their reason in cropping of the data as augmentation for training and in the fact that no objects were supressed in the training of the semantic segmentation algorithm, in difference to typical training setups for 2D bound box algorithms.

In E2.3.4 we continue the analysis of fine tuning and domain adaptation approaches to improve the performance for the modified sensor parameters.

3.3.3.2 Bosch experiments and results

The goal of this sub-workpackage is to measure the effect of changing camera setup on the KPI's of pedistrian detection. The data for training and evaluation has been taken from KI-A project synthetic generated images. Image data has been taken from two rendering pipelines of Mackevision (MV) and BIT Technology Systems (BT-TS). The image data represents the use-cases and default setup of the camera sensor.

The following is an ovierview of the data used for training and evaluation:

- BIT_TS: images rendered by the pipeline from the company BIT_TS . All images produced in using this pipeline are in default sensor setup
- Mackevision: images rendered by the pipeline from the compnay Mackevision. Tranche_06 includes sequences that are used for evaluation of the use-cases. Other sequences are produced in the default setup. use-case 4 (Color) and Use Case 2 (Resolution) can be constructed from any default setup.

The figure below gives an overview of the data coming from both rendering pipelines.

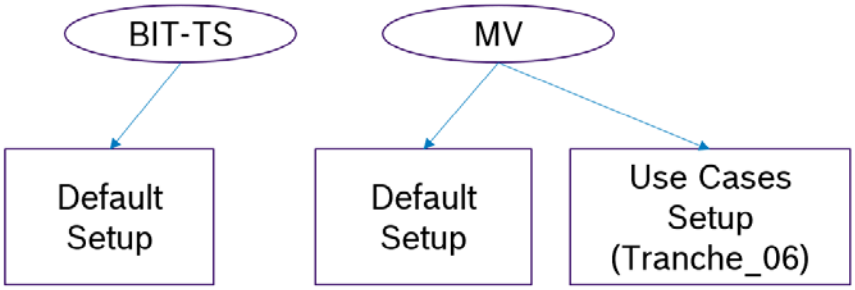


Figure: overview of the data used by Bosch for measuring the effect of changing the sensor setup

The splits for train/val/test has been taken from the official KI-A data split

3.3.3.3 Baseline Experiment Design and Results

Goal of baseline experiments is to measure performance of object detection for different use cases with relative to default. To achieve this goal a set of experiments have been designed. Ex_Baseline_00/01 are aimed to train/validate and test on splits for each rendering pipeline data (MV and BIT_TS). Faster RCNN is used as detection model.

The following tabels show the results of training/testing experiments where each experiment has a name, training data, test data and mean average percision of pedestrian detection. The name of the experiment begins with MV or BIT for the pipeline followed optionally by tranche number, the sensor setup and finally the color space.

Experiments Ex_Baseline_02_xx are designed to test the trained model on MV default setup data on each of the use-cases.

The following is a summary of the experiments' design and the results.

Exp_Name	Train Data	Test Data	mAP
Ex_Baseline_00	MV_default_rgb	MV_default_rgb	0.72
Ex_Baseline_01	BIT_default_rgb	BIT_default_rgb	0.73

Figure: Baseline pedestrian detection performance of each of the two datasets from MV and BIT-TS

Exp_Name	Train	Test	mAP
Ex_Baseline_02_01	MV_default_rgb	MV_tr06_default_rgb	0.62
Ex_Baseline_02_02	MV_default_rgb	MV_tr06_FOV100_rgb	0.43
Ex_Baseline_02_03	MV_default_rgb	MV_tr06_lower_resolution_rgb	0.59
Ex_Baseline_02_04	MV_default_rgb	MV_tr06_default_gray	0.34
Ex_Baseline_02_05	MV_default_rgb	MV_tr06_Hight+1m	0.60

Figure: Pedestrian detection tested on each of the use-cases for different camera sensor setup

The results of the experiments Ex_Baseline_02_xx shows the following:

- The performance tested on lower resolution setup is minimally affected by the change in camera setup. It is important to note that the evaluation protocol ignores small objects which have height<8 pixels. The quality of detection of the rest of objects has been minimally affected.
- The performance changing camera height (height +1m) is only 2 percent worse than the default setup. One explanation of this is that the appearance of the pedestrians is not much affected due to changing camera setup. In order to measure the effect of the

changing in camera height more data is need in which the camera height is larger than 1 meter (for example camera on truck)

- Changing the color space to grayscale has large effect on the pedestrian detection performance due to loss of appearance information.
- The analysis of evaulation of FOV100 was not possible due to many label errors, where whole label is missing or the occusion estimation has large value which is not correct by visual inspection

Ex_Baseline_03_xx shows the domain gap between the two rendering pipeline (MV and BIT_TS) in which the models trainined on one dataset are tested on the other dataset and the training and test data are in default setup.

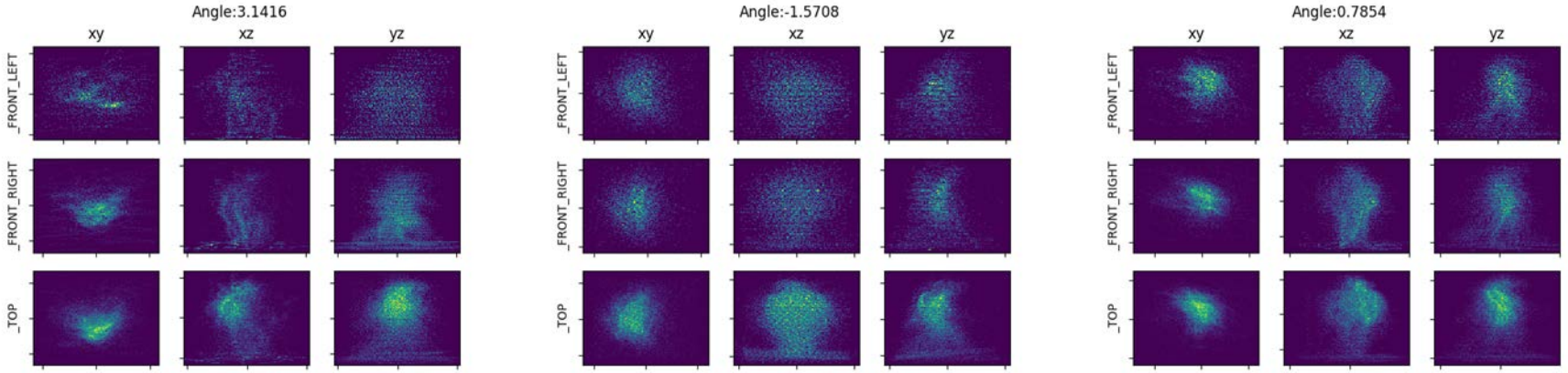
Exp_Name	Train Data	Test Data	mAP
Ex_Baseline_03_01	MV_default_rgb	BIT_default_rgb	0.49
Ex_Baseline_03_02	BIT_default_rgb	MV_default_rgb	0.43

Figure: Measuring domain gap between the two rendering pipelines in terms of pedestrian detection performance

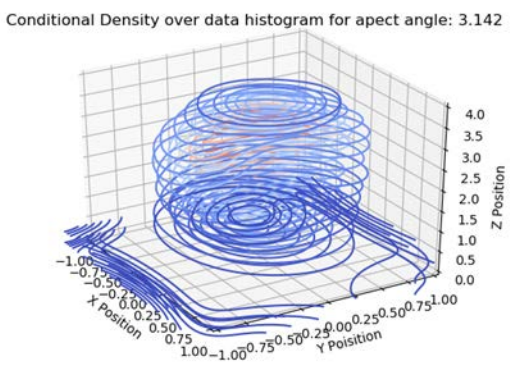
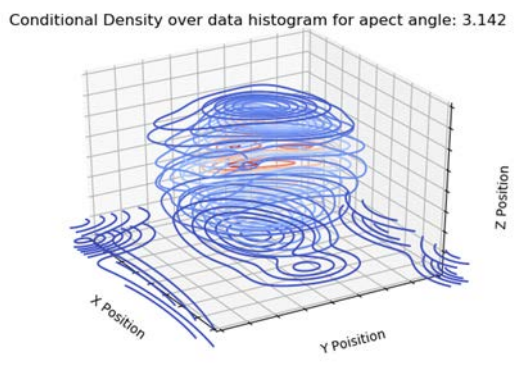
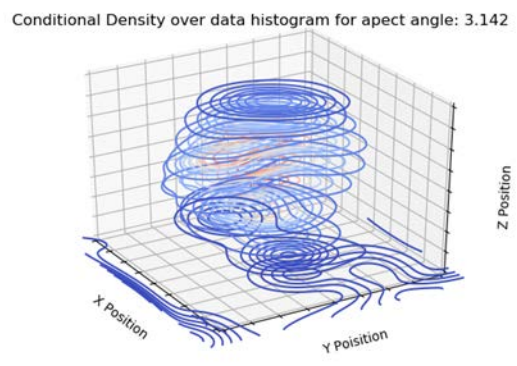
3.3.3.4 Valeo experiments and results

The aim of this work is to analyze the effect of the primary influence factor of the mounting position of lidar sensors on the distribution of measurements on the target. We used public datasets with lidar sensors mounted on the roof as well as on the height of the bumper to accumulate data in a shared scale independent coordinate system.

This data was used to fit a general distribution on the targets, for which we used a Gaussian mixture model, which allows for a comparison of the target's domain, specifically in the form of a likelihood function where measurements are generated on the target. During the accumulation biases in the data have been perceived based on the relative view from the sensor on the target. These have been removed by partitioning the viewpoints into angular sections shown in the following images.

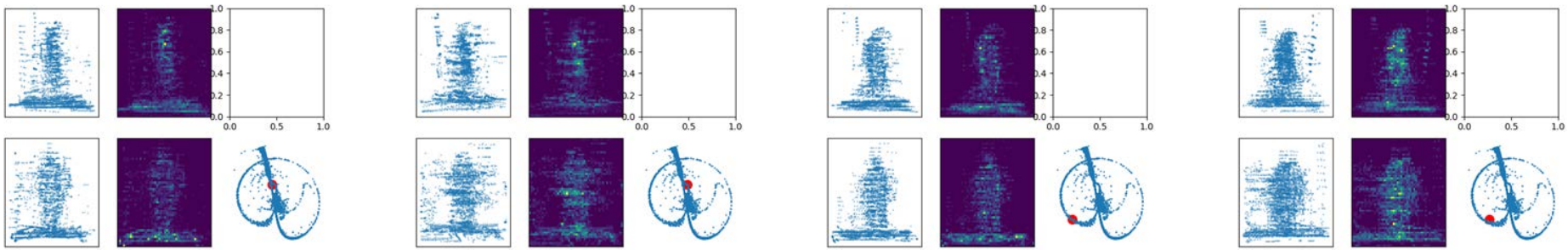


We used the data to train conditioned mixture components in the space of the accumulated coordinates. The model is trained with dimensions x, y, z, aspect angle and we illustrated normalized slices through the likelihood for an exemplary conditioned aspect angle in the following Figures:

Top:	Front_Left:	Front_Right:
		

Based on this accumulation we realized that clustering the data into different leg poses would be desirable, especially regarding free limb movement of arms and legs.

We started to investigate possible clustering methods of the data and aimed for the use of self organizing spring graph based approaches to represent similar groups of frames.This results in a fully connected graph with weights that represent the score in a point cloud metric between the frames. The nodes are then self organizing based on an iterative approach forming structures based on their similarity. In our setup we aimed for the use of the Chamfer distance which is commonly used in our field of research. A set of ~10000 frames was then used to provide a larger graph on which we investigated the frames by applying a nearest neighbor accumulation on specific positions in the graph. We provide examples of the given accumulation of data points in the following figures based on a k nearest neighbor accumulation with k=20:

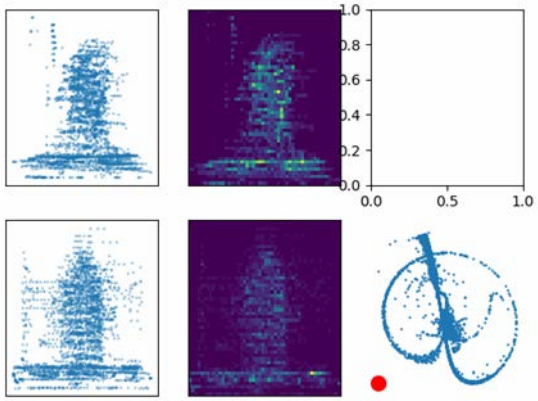


3.3.3.5 Valeo Result

We performed a comparative analysis for the mounting position of lidar sensors as primary influence factor on the measurement distribution of pedestrians.

We provided an approach for the analysis and comparison based on a shared coordinate system in lidar sensors and investigated the potential approaches for structuring the data in domains. We found that lower mounted lidar systems provide more structure on the lower limbs, expressed in a dilution of the likelihood function, while the roof mounted lidar predominantly provides measurements on a pedestrian's upper body. For the less rigid body parts such as the legs the distribution indicates a large spatial spread. Thus, the likelihood of seeing the legs is reduced due to the spatial spread, even though it is a feature that gets better resolved.

We further investigated an approach to structure the lidar frames, which unfortunately did not provide a satisfying result. While locally a strong similarity of the point clouds is visible no larger usable measurement structure in the limb position was found in the inspection of the graph.The following animated gif shows a grid based coverage over the given space to show the general structure of the graph:



Eventhough, this approach provides no global structure, future approaches in organizing the data could be pursued. The data could be organized using another distance function or a variational autoencoders and structure analysis in the embedding space could be performed.

3.4 E2.3.4 Final: Finetuning-Ansatz inkl. Quantifizierung notwendiger Menge an Trainingsdaten (zur Veröffentlichung)

3.4.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	*D*
Type of content	*Evaluation*
Classification level	*INT, LI, PU*

3.4.2 Description of the result

As result of experiments the amount of data is analysed that is needed to move from an AI function trained with data of the original sensor setup to a new version of the AI function for the new sensor setup that show similar performance as the original function, but w.r.t. the new sensor setup. (The required data is data for the new sensor setup. It is used to retrain the AI function.)

For the analysis, selected versions of AI functions of TP1 that are trained on data with the standard sensor setup are retained with data of the new sensor setup. Data amount is varied in order to analyze which amout is required to reach a performance similar to the original setup.

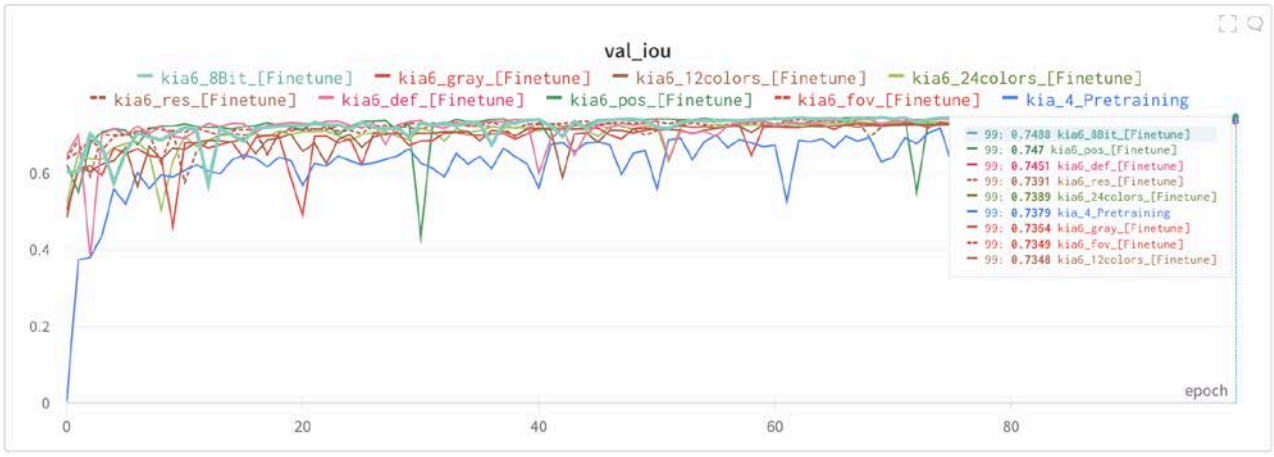
3.4.3 Experiments and Results

3.4.3.1 BMW Experiments and results

Building on the experimental setup of E2.3.3 with DeepLabV3 we examine the benefit of the following domain adaptation methods for increasing the performance on the target domain with the modified sensor parameters:

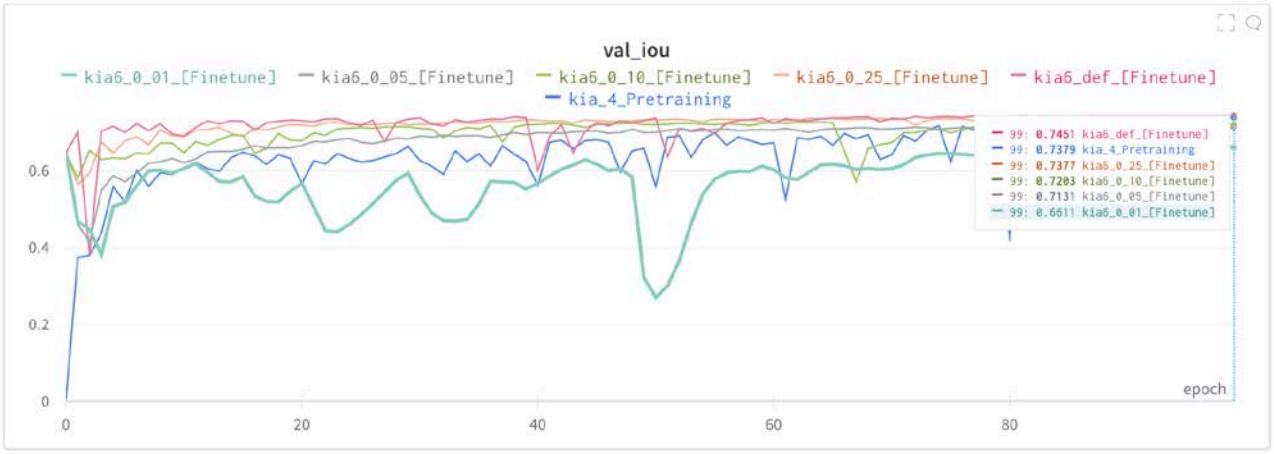
- fine tuning on the target domain (supervised)
- CCSA - classification and contrastive semantic alignment method (weakly supervised): S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, “Unified deep supervisedomain adaptation and generalization,” 2017.
- FTAdaptSegNet (supervised): FTAdaptSegNet is an extension of AdaptSegNet that we invented in order to allow the mechanism to use labels for target data to the degree to which they exist.
  Reference for AdaptSegNet (unsupervised): Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker, “Learning to adapt structured output space for semantic segmentation,” 2020.

Note that we do not experiment with all the domain adaptation methods that we consider in our work in AP2.4.3, because the unsupervised approaches cannot compete with the supervised ones for the amout of data that we consider. Using the Mackevision sequences 80, 81 and 82 for training with the domain adaptation approaches and 77, 78 and 79 for testing (depending on the variant), we obtail the following result:



The chart shows, that with fine tuning on the camera parameter variant DeepLabV3 for each variant reaches a performance that is roughly equal to the performance after training and testing on tranche 4, which was 0.74 mIoU. AdaptSegNet and CCSA do not reach the performance level of fine tuning in our experiments.

In order to study the necessary amount of training data we conducted further experiments with only 1% / 5% / 10% / 25% of the labeled training data using finetuning. The results are as follows:



The performance drop for finetuning with just a few images 20 (1%), 100 (5%), 200 (10%), 500 (25%) is relatively small. In comparison to the training with almost 2000 images the performance dropped by 3.2% for the training on 100 images, for 200 and 500 images even less. An expectably bigger drop of 8.1% for the 20 images-experiment was determined. Still approaches like AdaptSegNet or CCSA didn't outperform the finetuning approach. Further experiments on domain adaptation approaches were conducted. AdaptSegNet and CCSA were performing much worse than the simple finetuning baseline. Eventually, hyperparameter tuning could bring some effects, but the approaches tended to be instable and vanishing gradient issues were appearing what made hyperparameter tuning quite difficult.

3.4.3.2 DFKI Experiments and results

DFKI has conducted experiments in terms of 2D Human Pose Estimation with varied sensor settings. Since the project data commonly presents multiple persons per scene, the task is formulated as multi-person human pose estimation, i.e. keypoints (19 joints of the human body) are predicted for each visible person within a range of 50 m. Furthermore, the two main sensor

settings which are considered in these experiements is the default one (setting A) and a variation in sensor position (setting C), in detail the camera is shifted by 1 m upwards. The exact data used for the experiments are the sequences 67, 81, 66, and 78 of Mackevision for setting A and C and training and testing, respectively. A StackedHourglassNetwork [A. Newell, K. Yang, J. Deng. "Stacked hourglass networks for human pose estimation." *ECCV,* 2016] is used and initially trained on each setting individually. In a second step, fine-tuning from setting A to C is performed with varying amount of data of setting C. Each model is then evaluated on both settings. In all cases, the training is always fully supervised. The training objective as well as the evaluation metric is the mean squared error (MSE) on the heatmaps of all keypoints. The following table summarizes the results:

	Test	A	C
Epochs	Train	MSE	MSE
9	A	0.00325	0.00515
12	C	0.00541	0.00320
3	A then C (25 %)	0.00418	0.00336
3	A then C (50 %)	0.00443	0.00329
2	A then C (75 %)	0.00472	0.00326
5	A then C (100 %)	0.00465	0.00314

From these numbers we can conclude that there is similar difference from one setting to the other and vice versa. Obviously, the results on the data that was used for training are better than on the other setting. Further, fine-tuning with the entire data of setting C reaches a comparable level of accuracy as with training on setting C directly (even slightly better). However, it does not surpass those numbers significantly, which indicates that there is not positive forward transfer of knowledge from setting A to setting C when fine-tuning. The opposite effect can be observed: After fine-tuning on setting C, the task is better solved on setting A than with a model that was solely trained on setting C. We can assume a positive backward transfer here. With respect to the amount of data required for fine-tuning, we can state that a significantly lesser amount of data from setting C (25 %) is sufficient to reach a reasonable level of accuracy on this setting after fine-tuning. In fact with increased amount of data for fine-tuning, the forgetting of setting A increases steadily with only marginally positive impact for setting C. Lastly, it is evident that fine-tuning itself requires a significant less number of iterations compared to full training on setting C until convergence.

3.5 E2.3.5: Final: Auf Domain-Adaptation beruhende Ansätze, mit dem die KPI-Werte erhöht werden können (zur Veröffentlichung)

3.5.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Code*

Criteria	Classification according to VHB
Group/Cluster	*E*
Type of content	*Method*
Classification level	*INT, LI, PU*

3.5.2 Description of the result

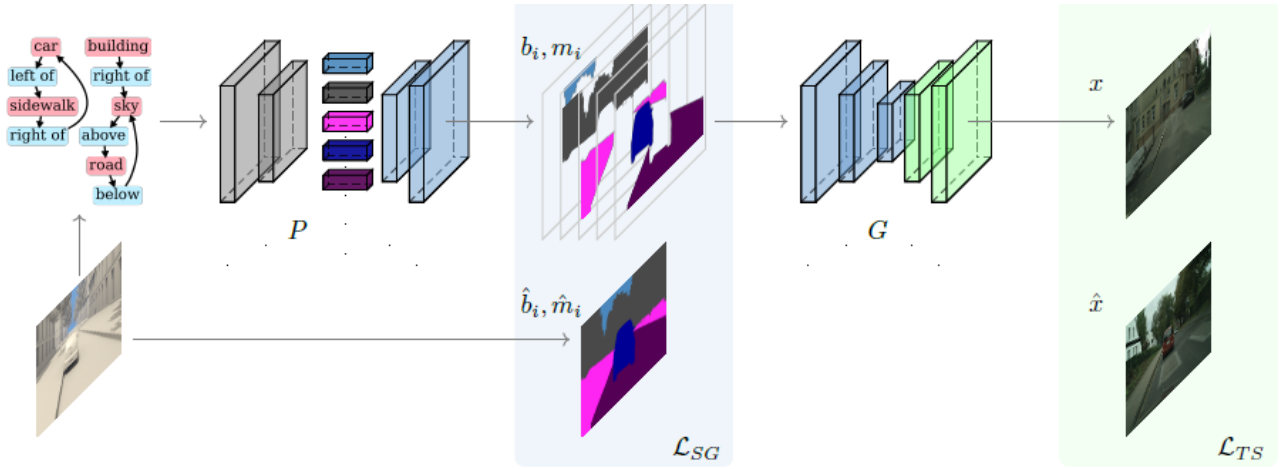
Alternative methods from the field of domain adaptation are implemented and analyzed. Domain adaptations offers means (1) to modify the training proccess such that the differences between source and target data are compensated and (2) to adapt data from one domain to a different one. In our case, one considered difference between source and target data is the difference caused by different sensor setups.

3.5.3 Experiments and Results

3.5.3.1 BMW experiments and results

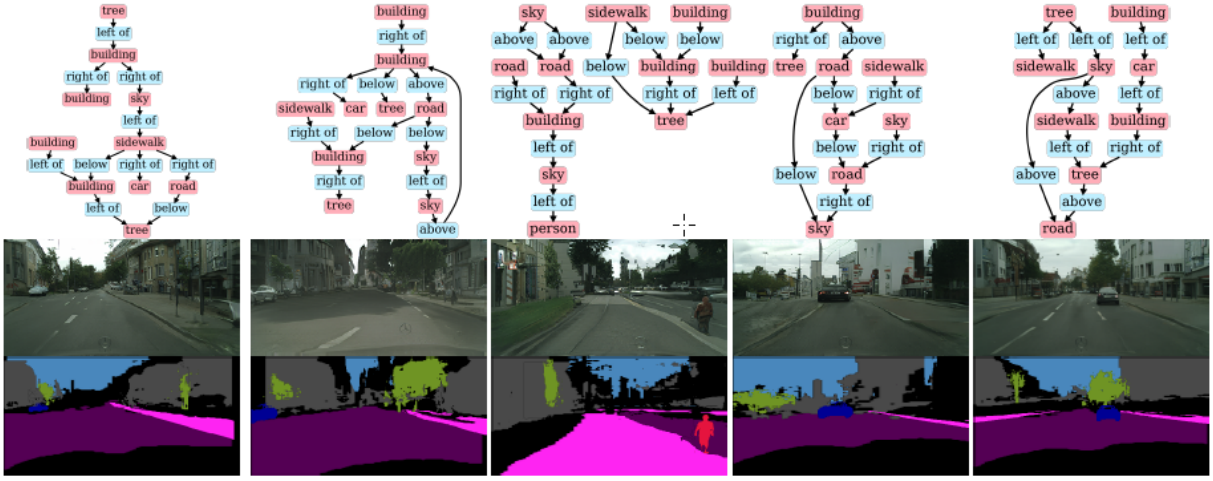
As one way to solve domain adaptation tasks, we developed and analyzed a method for directly synthesizing traffic scene imagery from a domain-invariant scene representation without rendering. Specifically, we rely on synthetic scene graphs as our internal representation and introduce an unsupervised neural network architecture for realistic traffic scene synthesis. The synthetic scene graphs can be enhanced with spatial information about the scene and we can do scene manipulation on the graph level.

The following figure provides an overiew of our approach:



We derive a synthetic scene graph from a procedurally generated scene that does not provide visual characteristics such as textures or materials. For the synthesized scene graph, we then produce an image of the corresponding traffic scene, which resembles the content of the underlying synthetic scene and the realistic appearance of the target data.

Some application examples are provided in the figure below.



The synthesized traffic scenes are based on a generator trained on Cityscape.

For more details we refer to this paper: Savkin, Artem, et al. "Unsupervised Traffic Scene Generation with Synthetic 3D Scene Graphs." *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

3.5.3.2 Bosch Experiments and Results

3.5.3.2.1 Camera Sensor Abstraction

The target of the experiments is to achieve sensor abstraction with a new sensor. In other words, how can we make use of data coming from two different domains to achieve a good performance on both domains.

There is a performance drop in the KPI of pedestrian detection between the two domains represented by MV and BIT-TS rendering pipelines. The simplest method of domain adaption would be to mix the data from the two domains for training a new model. More complex domain adaption using GAN (Generative Adversarial Networks) can also be used to learn an appearance based transformation from one domain to the other.

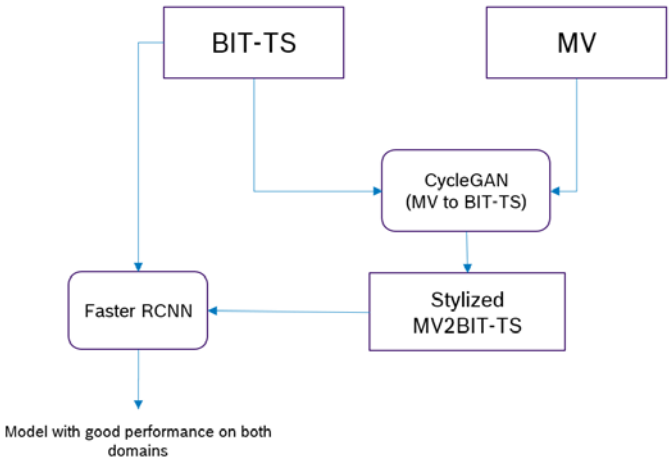


Figure: Experiment design to acheive sensor abstraction between tow different domains

The experiments are based on training on mixed data from the two domains as well as using CycleGAN learn to transform from MV to BIT-TS domain and then the Faster-RCNN model can be trained using transformed images as shown in the figure above.

The following figure shows a summary of the experiments. The first two rows (Ex_Baseline_00 and ExBaseline_01) show the baseline performance achieved by training only on one domain. The second two rows (Ex_Baseline03_01 and Ex_Baseline03_02) show the performance on the other domain. The third two rows show the effect of mixing the data from both domains and testing on each domain. The last two rows show the performance of mixed training with stylized images from MV to BIT-TS

Description	Exp_Name	Train Data	Test Data	mAP
Baseline	Ex_Baseline_00	MV_default_rgb	MV_default_rgb	0.72
	Ex_Baseline_01	BIT_default_rgb	BIT_default_rgb	0.73
Domain Gab	Ex_Baseline_03_02	BIT_default_rgb	MV_default_rgb	0.43
	Ex_Baseline_03_01	MV_default_rgb	BIT_default_rgb	0.49
Mixed Training **without** GAN stylization	Ex_rend_pipeline_00_02	BIT_default_rgb + MV_default_rgb	MV_default_rgb	0.65
	Ex_rend_pipeline_00_01	BIT_default_rgb + MV_default_rgb	BIT_default_rgb	0.68
Mixed Training **with** GAN stylization	Ex_rend_pipeline_03	BIT_default_rgb + MV2bit_default_rgb	MV_default_rgb	0.70
	Ex_rend_pipeline_01	BIT_default_rgb + MV2bit_default_rgb	BIT_default_rgb	0.74

Figure: Results of the domain adapation experiments

Simple Mixing of datasets from both domains enhances the performance but still there is a domain gap in performance. Mixing the datasets using GAN-based domain adaption can achieve a performance that is comparable to baseline and hence achieve sensor abstraction for pedestrian detection.

3.5.3.2.2 Local Domain Adpation for Object Detection

GRL (Gradient Resversal Layer) algorithm for domain dapation is based on full frame domain classification. Bosch has developed an algorithm based on GRL where domain classification loss is computed on restricted image areas e.g. on vehicle. The features can then be guided to particularly being domain invariant for target objects. The base detector is one-shot object detector such as YOLO. Experiments on Bosch internal data has shown improved performance for object detection tasks. The figure below shows the region of interests where the domain classification is applied.

Figure: Applying domain classification only on region of interests

3.5.3.3 Colorspace Sensor Abstraction

Experiments for studying sensor abstraction between RGB and Grayscale color spaces has been conducted. Target is to make the model color space independent.

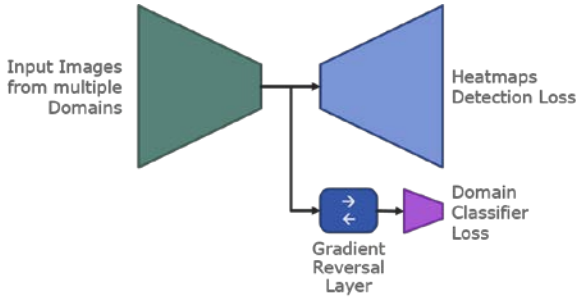
The figure below shows that training a model on one colorspace and testing it on different colorspace causes performance drop. (Ex_Baseline_02_01 and Ex_Baseline_02_04). Using color-based augmentation such as photometric distortion and grayscal enable the model to achieve color space invariance. This can be seen in Ex_color_01 and Ex_color_02. Results of Ex_color_03 and Ex_color_04 shows that even training with grayscale images and using photometric distortion also leads good results (3-4% less than baseline) when tested on RGB images.

Exp_Name	Train	Augmentation	Test	mAP
Ex_Baseline_02_01	MV_default_rgb	no color aug	MV_tr06_default_rgb	0.62
Ex_Baseline_02_04	MV_default_rgb	no color aug	MV_tr06_default_gray	0.34
Ex_color_01	MV_default_rgb	photometric_dist +gray	MV_tr06_default_gray	0.59
Ex_color_02	MV_default_rgb	photometric_dist+gray	MV_tr06_default_color	0.61
Ex_color_03	MV_default_gray	photometric_dist	MV_tr06_default_rgb	0.58
Ex_color_04	MV_default_gray	photometric_dist	MV_tr06_default_gray	0.59

Figure: Experiments to study colorspace sensor abstraction

3.5.3.4 DFKI experiments and results

In order to mitigate the drop in accuracy when switching from a source domain to a target domain, we have followed the concept of modifying the training such that the differences of domains in the learned feature space are compensated. In detail, we follow the domain adversarial approach of [Y. Ganin and V. Lempitsky. "Unsupervised domain adaptation by backpropagation." *ICML*, 2015], which aims to learn domain-invariant features during the supervised training on the source domain. We have adopted the adversarial scheme to fit our application (2D human pose estimation) with a StackedHourglassNetwork [A. Newell, K. Yang, J. Deng. "Stacked hourglass networks for human pose estimation." *ECCV,* 2016]. I.e. we have split the HourglassNetwork at the bottleneck into feature extractor and keypoint detector and branch off a subnetwork (domain classifier) which aims to differentiate between images of the source and target domain. While this distinction is only possible with sufficiently diverse features of both domains, the goal is to obtain domain-invariant features. Therefore, the gradient between the feature extractor and domain classifier are inverted by a Gradient Reversal Layer. As a result, the loss of the domain classifier is enforcing features for both domains, which can not be differentiated, i.e. they are domain-invariant. An overview of the concept is outlined in the following figure:



It is important to note that the data of the target domain does not need to provide labels for the application task (2D human pose). To compute the loss for the domain classifier, knowledge of the domain for each sample is sufficient (i.e. source/target, binary classification). As such, the approach is self-supervised w.r.t the target domain. In our experiments, we found that the training of such an adversarial approach is very sensitive to many hyperparameters, e.g. the losses for domain classification and human pose need to be carefully balanced.

Due to the longlasting lack of suitable data with variation of sensors and annotations of human poses, we have performed a proof of concept on two alternative domains, i.e. synthetic and realistic data.
Our results show, that this method performs limited in case the domain gap is too large. To little surprise, none of our experiments could outperform a model which was trained on the target domain in a supervised fashion.

4 AP2.4 - Bewertung der Qualität synthetischer Daten

4.1 E2.4.1a Final: nur projektintern für KI Absicherung verfügbar

4.2 E2.4.1b Final: nur projektintern für KI Absicherung verfügbar

4.3 E2.4.1c Final: nur projektintern für KI Absicherung verfügbar

4.4 E2.4.2 Final: nur projektintern für KI Absicherung verfügbar

4.5 E2.4.3 Final: nur projektintern für KI Absicherung verfügbar

4.6 E2.4.4 Final: nur projektintern für KI Absicherung verfügbar

4.7 E2.4.5 Final: Wirkkettenanalysen aus der Anwendung gezielt variierter Datensätze für grenzwertige Anwendungssituationen (zur Veröffentlichung)

4.7.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	*Methodik*
Type of content	*Reports on observed defects, analysis and counter measures or workarounds*
Classification level	*INT, LI, PU*

4.7.2 Description of the result

This report describes a number of experiments for the estimation of effects and their consequences upon parameter variations for the generation of corner cases.

The methods employed first identify parameters such as image brightness and contrast, pedestrian orientation, density and proximity to the camera, and then vary these parameters systematically to create corner cases or situations which appear seldom in practice or which the network has not yet being exposed to, e.g. trained for. A posterior evaluation on the performance sheds light on the effects caused by such parameter variation.

The identification of critical values in the parameter variation allow to estimate the sensitivity of the model with respect to the selected parameter, but also to identify which parameters are critical at all.

4.7.2.1 Approach

Using synthetic data from KIA and from real datasets such as Cityscapes, the AI model is trained and then exposed to corner cases not previously seen by the network, possibly applying first different techniques for domain adaptation. The generated corner cases are product of the systematic variation of selected parameters. Finally the performance of the network is evaluated in order to assess the impact of the variation of the selected parameter and compared with cases for which the network was trained.

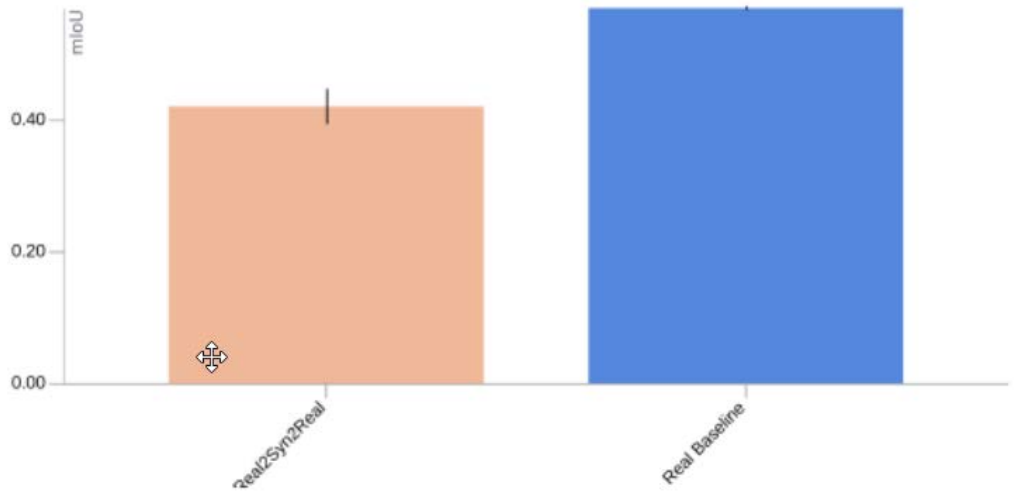
The identification of critical values upon the parameter variation above which a significant performance increase is achieved allow to identify the sensitivity of the network trained with the given dataset to the selected parameter variation.

4.7.3 Result

4.7.3.1 BMW

Analysis 1: One intention in using synthetic data is to train the AI function for situations that are very seldom in practice, like almost accidents. In order to examine data quality w.r.t. this objective we introduced two artifical domain gaps, (1) luminance (dim vs. bright frames) and (2) pedestrian proximity (far vs. close) and partitioned the data accordingly. With this basis we investigated a setup where the AI function was trained on real dim data / real data with far pedestrians, adapted to the synthetic domain with bright frames / close pedestrians and evaluated on real bright data /real data with close pedestrians. This mimics the situation where we use synthetic data to train for rare situations.

The overall setup we use for this experiment is based on our experiments with domain adaptation techniques in E2.4.3. In particular we again use DeeplabV3, and the datasets Cityscapes, Synscapes, KI-A Tranche 3 and KI-A Tranche 4 (BIT-TS part). Averaging over the domain adaptation methods and the two artificial domain gaps, we obtain this result:

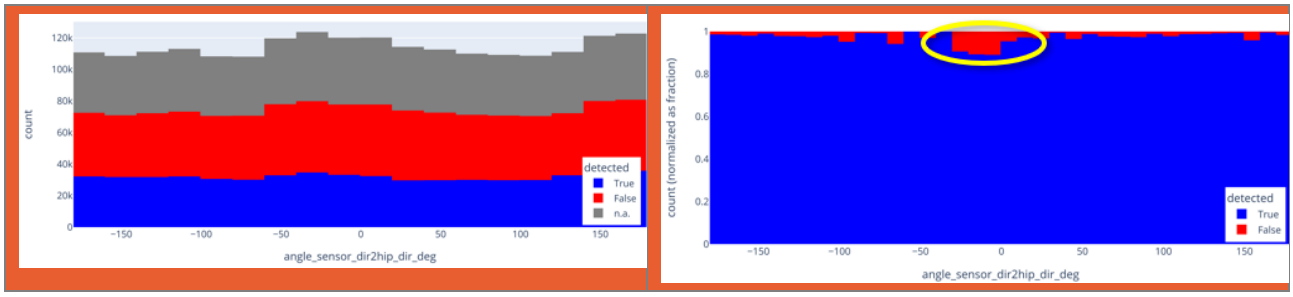


A considerable performance it achieved, especially when we consider that the AI function never saw training data of the real target partition, but the baseline of having real data for the second partition is not beat. Note further that in this case the unsupervised domain adaptation methods CCSA (best) and AdaptSegNet showed best performance.

Anaylsis 2: Motivated by the evidence workstreams in the project, an analysis was conducted that examines how the data is distributed when we group it along the influence factor of the orientation of the pedestrians towards the camera. Furthermore, the effect of the orientation on the performance was evaluated. As basis for the evaluation, we use the SSD from TP1 in release 3 version 2, as provided by Opel, and the data of Mackevision tranche 5. The evaluation is done for the pedestrian relevance categories as proposed by TP4, i.e. basically those pedestrians that are within the driving tube of the ego vehicle.

The result is that the tranche 5 data covers all orientations with a high number of samples and rather uniformly (see left figure below). Note however, that there is one sequence that differs

from a uniform distribution. Examining whether the false negativ rate of the considered SSD is essentially the same for all orientations, a Kolmogorov-Smirnov test provides the result that this is not the case. The false negative rate for pedestrians facing to 11h to 12h (away from the camera, face is not visible, -30° - 0° in the figure below) is higher and that deviation is statistically significant (see right figure below). While this result cannot be generalized to other SSD parameters or other algorithms, it nevertheless shows the relevance of such kinds of analysis for the safety argumentation.

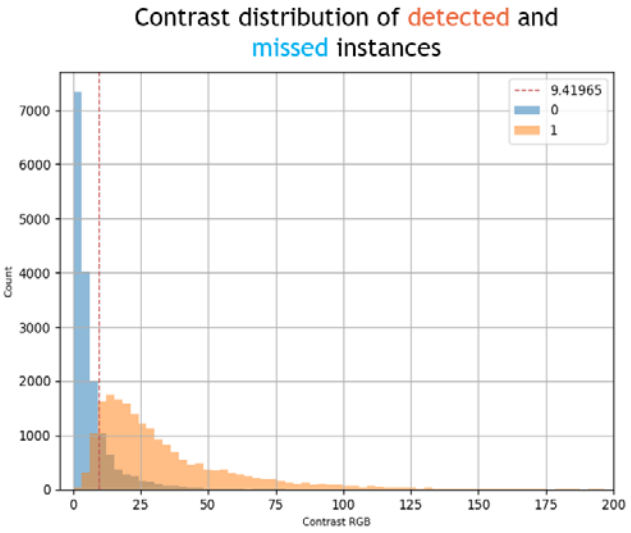


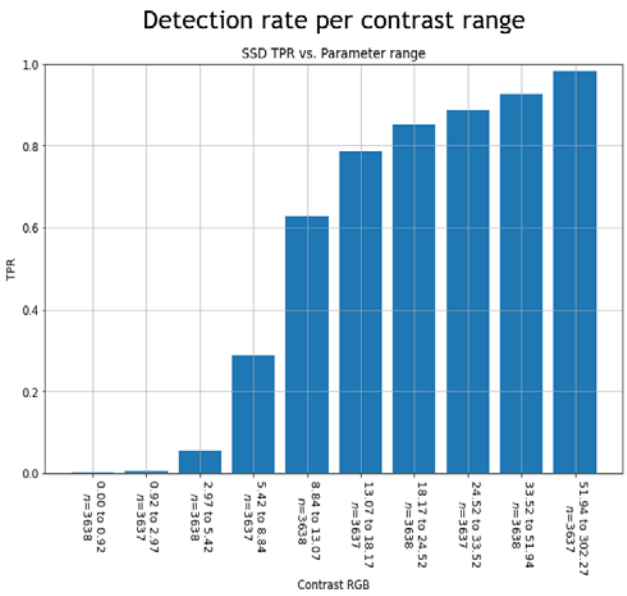
4.7.3.2 Bosch

The evaluation of the effect-chain requires the systematic variation of the input data. As an experiment to evaluate the interaction and correlation of parameters as described in E2.4.2 and E2.4.3 in more detail, a systematic experiment regarding pedestrian contrast was conducted.

A subset of instances which are initially detected by the POPC release of the SSD detection method in Bit Tranche 04 Data is selected. (*375 images, 4546 person instances).* The contrast of the full image is reduced, by decreasing the value range in 8 steps. This results in a systematic reduction of image contrast, with no further changes to the image. For each variant the metadata describing the contrast between instance pixels and background is measured and the POPC SSD is applied again. The initially detected instances (original image) are analyzed in all variants. This enables to analyze detection performance with respect to image appearance in a more controlled setting, as scene parameters (e.g., occlusion) do not vary between the contrast variants.

We evaluate the detection rate for 10th percentiles in regard to the contrast metadata. Additionally, the distribution of contrast values for detected and non detected pedestrians is shown.





The drop in performance in regard to the pedestrian contrast can be shown. To define a critical threshold regarding the pedestrian contrast value we group pedestrians instances into detected and non detected pedestrians and apply the Youden's J metric to divide the distributions. Based on this experiment, this results in a critical threshold for detection at 9.42 as indicated by the red dashed line in the Figure.

4.7.3.3 Luxoft

Problem

Usually, the evaluation of DNNs predictions and the search for corner cases is limited to the analysis of metrics obtained from the segmentation results of the instances itself, isolated from their environment. By doing so, important information revealing when or why a certain class is poorly predicted is overseen. Moreover, when using for example the mIoU of a single class in an image to evaluate the performance of the DNN, all pixels that belong to all instances of this class are merged in the computation of this metric, averaging potentially good with bad prediction results of this class.

Hypothesis

The segmentation errors of an instance of a class not only depend on factors like image contrast, but also of the environment that surrounds the instances.

Backgrounds that share colors and textures seen during training in pedestrians, the only class we are evaluating in this work, may have a negative impact in the prediction results.

Metrics like mIoU are only meaningful to evaluate the average performance of a DNN, but it is not enough to gain insight in the source of errors produced by the DNN

Method

The background of pedestrian images taken from the KIA dataset have been replaced by a set of new backgrounds shown in the bottom row of Fig. 3. These can be grouped in 5

categories depending on the object they have been extracted from: buildings, cars, clothes, grounds and sky. In order to avoid cross domain issues, all backgrounds have been taken from images of the same dataset used to train and test the DNN (KIA Tranche 3). The background kernels (bottom row of Fig. 3) are randomly scaled and repeated in a mosaic to cover the whole surface of the bounding box that has been used to extract the pedestrian from the test dataset images.

The new set of images (top row of Fig. 3) have been segmented by the DNN (DeepLabV3+ trained with KIA tranche3+4)

Preliminary Results

The prediction results are shown in the middle row of Fig3. It can be observed, that the worst predictions (images 5, 6 and 9) belong to the background generated with clothes of pedestrians (gray jacket, blue jean and red jacket respectively). Moreover, these backgrounds that obviously have the same colors and textures seen during training in the pedestrians, offer a clear contrast with the pedestrian for the subjective human perception. However, the DNN is completely wrong about the prediction. This suggests, that DNNs "pay more attention" to textures and colors than to shapes. On the other hand, the image 8 has been rather good predicted despite the blouse used as background, maybe due to a larger contrast.

Further Work

In order to distinguish between the influence of color and texture, a new set of synthetic backgrounds will be generated, where one color will be applied to different textures and one texture to different colors. This should give more insight about the source of "confusion" involved in the prediction process.

The corner cases identified in this work, like for example the images 5, 6 and 9, could provide new training material to enhance robustness and generalization of DNN models.

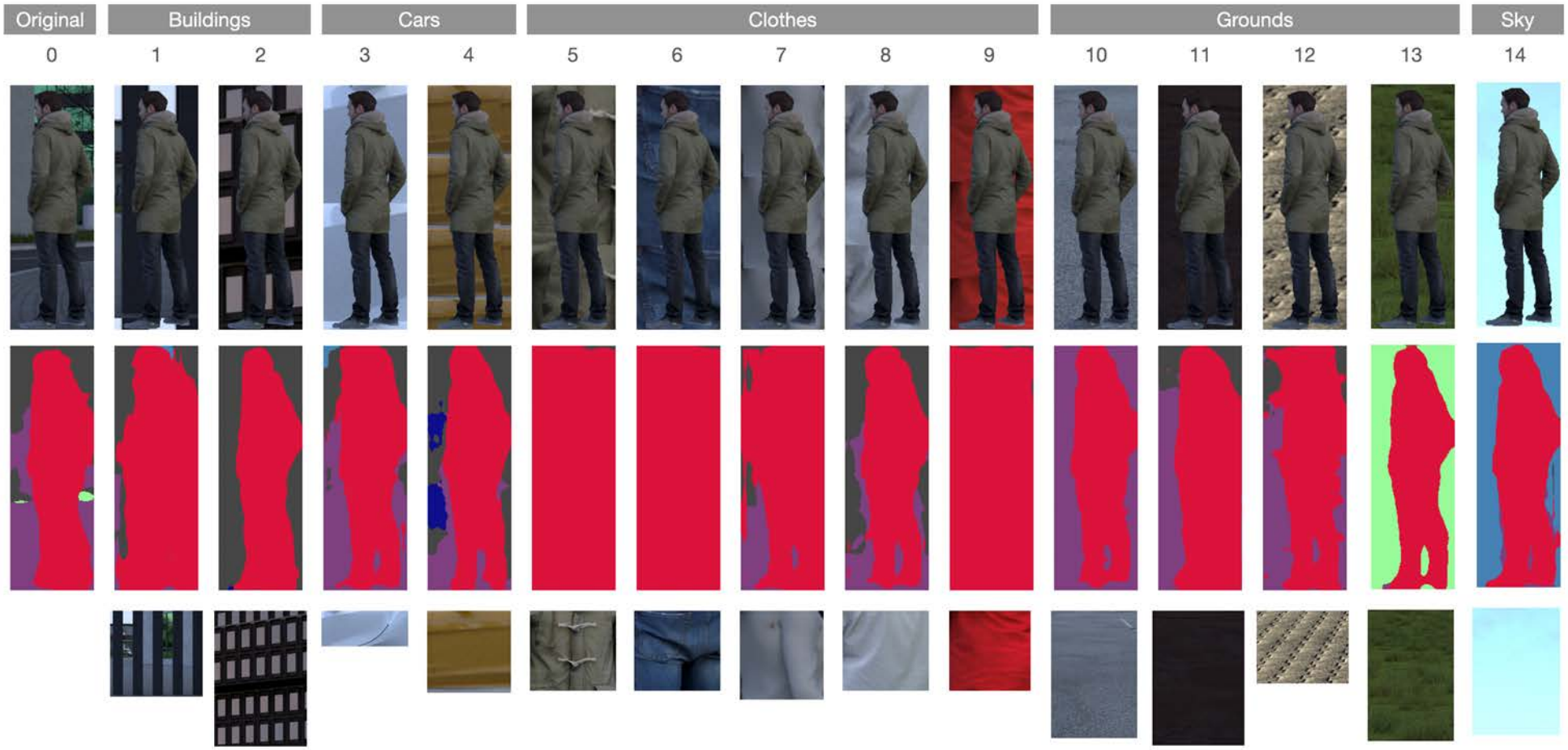


Fig. 3: Background replacement experiment and segmentation results

4.8 E2.4.6a Final: Guideline für die Erzeugung und Anwendung synthetischer Daten für Training und Test von KI-basierten Algorithmen für dedizierte Anwendungen (zur Veröffentlichung)

4.8.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	
Type of content	*Tool*
Classification level	*INT, LI, PU*

4.8.2 Description of the result

The following report describes guidelines and lessons hearned regarding the generation and usage of synthetic data as well as evaluation of results transfer to real-world applications.

The guidelines and lessons learned regarding data generation and usage were gathered along the project. Here we find a summary with direct references to results in other reports for deeper discussions.

Guidelines were grouped in the following categories:

- Guidelines on Data Requirements
- Guidelines on Data Production
- Guidelines on Data Analysis
- Guidelines on Transfer to Real-World Applications

4.8.2.1 Guidelines on Data Requirements

4.8.2.1.1 Data Use-Cases Requirements

KI Absicherung showed that in the beginning of a synthetic data generation project it is of utmost importance to get clarity on the use cases that the data must address. Use cases may involve the usage of the synthetic data for training only or for testing only or for both, alone or - more likely - in a combination with real data. For the combination with real data, it is important to understand what should be addressed by the synthetic data and what by the real data. Any specific scene can be approximated with real data, but the "chaos" of reality where things happen which no developer imagined, can hardly be captured. Clarity about the use cases is needed in order to align the data generation with it and prioritize data requirements accordingly. For instance, for training with synthetic data only, variance of the dataset is important. For testing and validation, coverage of the ODD is important as well as potentially oversampling of certain aspects, like putting more pedestrians on the road as in the reality, if this is required as input for the safety argumentation. While individualization of scenes according to users' needs can be done over the course of the project, the kind of scenes,

scenarios and variance that can be generated are a fundamental decision. From the project experience, we suggest to use synthetic data as a complement to real data.

A further important decision is whether data is supposed to be produced in an incremental manner, like in KI Absicherung, or within one go. The former allows the developers to learn what is important and optimize, while the latter has the advantage that a target data distribution, like some kind of input space coverage, can systematically be designed. In reality some degree of iteration is always necessary, but a clear idea of the target distribution is still helpful.

In regard to the domain gap between synthetic and real data, it makes sense to reduce the domain gap as small as possible with the available budget and tooling. A larger domain gap is acceptable when it leads to the desired result. E.g. images from domain randomization typically have a big domain gap, but can help to increase robustness of a detection function.

Guideline	Name	Description
GL-DR-1	Definition of use cases for the synthetic data	- Will the synthetic data be used for training, for testing or for both? - Will the synthetic be used in combination with real data? - If the synthetic data is used in combination with real data, what aspects will be covered by the synthetic data and what aspects by the real data? - Best practice: use synthetic data as a complement to real data
GL-DR-2	Prioritization of data generation goals	- Training with synthetic data only: data must possess enough variance - testing and validation with synthetic data: coverage of the ODD must be guaranteed, i.e. corner cases - validation of safety argumentation: oversampling of key aspects like e.g. adding more pedestrians to a scene than can be found in reality.
GL-DR-3	Definition of data production modality	- Incremental data production: allows to improve data quality but has less control on final target data distribution - One-shot data production: well-defined target data distribution for e.g. optimal input space coverage

Guideline	Name	Description
GL-DR-4	Domain gap between synthetic and real data should be as small as the available tooling and budget allows. A larger domain gaps is acceptable if it leads to the desired result.	The domain gap between synthetic and real data should be in relation to: 1) the available tooling and budget 2) the desired result, e.g. increased detection robustness through domain randomization, i.e. the automatic creation of a high data volume to improve learning of selected features in a given domain. For instance, in order to learn a large number of pedestrian poses, the number of pedestrians in a given scene may be unrealistic large but the quality of individual instances can remain moderate.

4.8.2.1.2 Content Requirements

In KI Absicherung it was useful to classify requirements into the following classes:

- requirements on production tools
- requirements on optical effects and rendering fidelity
- requirements on ground truth
- requirements on (sets of) frames
- requirements on scenes
- requirements on assets

Requirements that do not relate to direct geometric properties of a scene, but to the interaction of scene elements as captured by the sensor are difficult to handle. The requirement "scenes contain pedestrians in the shadow of trees" is an example. An interactive authoring tool is desirable for such requirements. Note that this does not mean that geometric properties are more or less important than interactions for DNN perception.

Many scene requirements become easier to handle if there is a scene specification language, like OpenScenario, because the languages support an unambiguous specification and possible also generation abilities. Even a format of limited expressiveness, like the JSON format introduced in the project, is a great help, because it allows us to express what is on actual interest to the engineer, e.g. certain occlusion situations. A pragmatic specification approach, e.g. based on a fixed scene, leads to quicker results than targeting on the most powerful specification language. The scene specification language should be based on an ontology for the problem space. In our case that is pedestrian detection in an urban environment.

Sometimes, randomization of scenes is preferable over exact scene design for a specific effect. This holds also true for requirements addressing the interaction of scene elements. With randomization, a larger number of frames is produced from a coarse scene specification and the

interesting frames are selected afterwards. Thus, randomization is an alternative to the detailed specification described above or can complement it.

Guideline	Name	Description
GL-CR-1	Definition requirements on production tools: - Requirements on optical effects and render quality - Requirements on ground truth	Define requirements on tooling for capturing scene content using two broad categories: optical effects / render quality and ground truth.
GL-CR-2	Tools requirements: Capture of object interactions in a scene	A scene authoring tool is recommended to capture object interactions, e.g. “scene contains pedestrians in the shadow of trees”. Geometric properties of the scene are automatically specified by such a tool.
GL-CR-3	Tools requirements: Usage of an ontology-based scene specification language	Scene specification languages like OpenScenario allow for disambiguation of the specification and automatic instance generation for specific distributions. The ontology should be adapted to the targeted problem space.
GL-CR-4	Frame set content creation requirements: Pragmatic scene specification for faster, more precise results	A pragmatic specification approach, e.g. based on a fixed scene, leads to quicker results than targeting on the most powerful specification language.
GL-CR-5	Frame set content creation requirements: Scene randomization versus detailed scene specification	Randomization produces a large number of frames from a coarse specification. Individual frames fulfilling the targeted interaction can be selected a posteriori.
GL-CR-6	Definition of requirements on frame sets	- Requirements on scenes - Requirements on assets

4.8.2.1.3 Requirements for specific Use-Cases / Challenges

Besides a detailed definition of the general use-case, specific challenges regarding the data content, data production and use-case have to be specified and communicated. This is crucial, especially for a subsequent safety argumentation.

The scenes have to be designed in a way that they cover particular factors having a high impact on our down-stream task.

Guideline	Name	Description
GL-UR-1	Scene design must cover performance limiting factors	The scenes have to be designed in a way that they cover particular factors having a high impact on our down-stream task. Considering pedestrian detection this can include factors like different kinds of occlusion or illumination.
GL-UR-2	Crucial parameters, effects and their variations for the specific use-case must be identified	Besides scene and scenario design also very specific sensor effects have to be considered. The same scene captured with a different camera can lead to very different resulting images. These specific requirements regarding a certain domain, e.g., camera sensors, can be based on expert knowledge. The identified parameters can lead to the definition of corner cases for the chosen use-case. Example: corner cases concerning light, as described in E2.4.1b. The production of synthetic data in the KI Absicherung project revealed that especially these specific requirements based on expert descriptions are challenging to implement for data production, which results in a high effort for coordination of requirements.
GL-UR-3	Identified parameters and their variations must be implementable by the tool chain	The production tooling has to be capable to fulfill all requirements. As many tools for rendering focus on the optical impression for humans rather than the physical correctness of effects, the capability of the production tooling to fulfill certain requirements (e.g., high dynamic range with at least 32-bit) should be verified in advance.
GL-UR-4	Usage of procedural image generation may alleviate implementation challenges associated with physically based effects	

4.8.2.2 Guidelines on Data Production

4.8.2.2.1 Data handling

One aspect of data quality is how good the data can be handled by the user. Considering that data is a long-term asset that is used by many people, it makes sense to give high priority to this aspect. Otherwise users might use other datasets or start new data collection campaigns. In particular it is important to keep specifications, like the labeling specification, constant over

data production. In KI Absicherung we invested considerable effort to level up generated data to updated versions of the specifications. However, if this would not have been done, users would have ignored the older data. In order to ensure that data conforms to its format specification it is highly recommended to have a data reader/data loader available right from the beginning of data production. This way we can ensure code-level compatibility of the data with downstream tools.

Guideline	Name	Description
GL-DP-DH-1	Keep consistency of data labeling across the entire dataset	It is important to keep specifications, like the labeling specification, constant over data production, particularly if the the data is produced incrementally as it was the case in KI-Absicherung. Doing so, will allow users to re-use and compare older data deliveries.
GL-DP-DH-2	A data reader/loader must be available from the beginning of data production and be consistent across all deliveries.	This will ensure that the data conforms to its format specification and will also provide code-level compatibility of data with downstream tools.

4.8.2.2.2 Dataset Structure

In the setting of the project, various kinds of ground truth and sensors had to be supported. Depending on the exact way the data is packaged, this typically causes that a large portion of the downloaded data is not of interest to the engineer implementing a specific use case. A central big data platform that allows to select which dataset features and/or frame attributes are necessary for a use case and that provides just this data is therefore recommended. If this is not possible, a solution allowing the engineer to select the classes of data that he/she needs and providing only those for download makes sense. For instance, in order to train camera-based semantic segmentation algorithms, the sensor PNGs and the ground truth PNGs are sufficient. There is no need to also provide 3d bounding boxes or depth maps.

Guideline	Name	Description
GL-DP-DS-1	A centralized big-data platform to manage and select data of interest for a particular use case allows for a streamlined development process	Developing a specific use case requires only portion of the data delivered. For instance for the development of camera-based semantic segmentation algorithms only sensor images and the corresponding semantic segmentation ground truth is required. Depending on how the data is packaged (compressed or uncompressed), a central data management system will provide this data to the developer and thus save time and communication bandwidth when transferring it.

Guideline	Name	Description
GL-DP-DS-2	A database with indexed metadata as alternative to a full-fledged data management system will also allow the developer to select the data that he/she needs and thus save time and communication bandwidth	

4.8.2.2.3 Render Tooling

Guideline	Name	Description
GL-DP-RT-1	Scene complexity and variety	Synthetic data can be characterized by: Scene complexity, asset (i.e. the 3D models) variation + fidelity (e.g. quality of surface properties) and the rendering fidelity - see next box. The first two can be characterized at a higher semantic level and include the appearance of of a high number of different objects, like pedestrians in a variation that is a fair approximation of what can be expected in the real world. For pedestrians that includes variations in size or height, sex, age, shape or figure and clothing. This leads to a high number of different assets required to cover practical appearance of human beings. The lack of variety is measurable by the quality metric developed in AP2.4 and has a significant influence, as datasets with few different assets are not keeping in par with real data sets.
GL-DP-RT-2	Render and asset/3D model fidelity	The render fidelity is characterized by the final appearance of the rendering output and a function of asset quality, in terms of shape or geometry quality, but also the surface properties and the actual rendering method. The influence of the rendering method can be measured by applying performance metrics to two (or more) datasets of the same content rendered with different render methods. It was demonstrated that a realistic sensor simulation (which includes physical-based rendering PBR) performs significantly better than basic CGI methods.

4.8.2.2.4 Assets

According to our experience a combination of self-created assets and usage of asset libraries makes sense. Asset libraries allow to increase the number of different assets at low cost, although these might not support all desired features, like the possibility to animate a person asset and provide pose estimation ground truth. Self-created assets are necessary in order to support all the desired features and they are also necessary in order to cover data gaps in asset libraries. Obviously, these libraries focus on assets of commercial interest and can therefore introduce bias. Furthermore, the quality of assets from asset libraries strongly varies. This may

cause problems in the tooling. In the project we defined asset quality guidelines for glTF assets such that they can be used in both data pipelines used in KI Absicherung.

Asset quality guidelines should be defined, refer to glTF Asset requirements agreed between BIT TS and Mackevision for the assets in the project.

Guideline	Name	Description
GL-DP-A-1	The combination of use-case specific (self created) assets and generic asset libraries provides a cost-effective / time saving solution	Asset libraries allow to increase the number of different assets at low cost, although these might not support all desired features, like the possibility to animate a person asset and provide pose estimation ground truth. Self-created assets are necessary in order to support all the desired features and they are also necessary in order to cover data gaps in asset libraries.
GL-DP-A-2	In order to guarantee a consistent and for the use-case optimal asset quality guidelines for the production of new assets should be defined and provided in a standardized format.	Commercial (i.e. non use-case specific) libraries focus on assets of commercial interest and can therefore introduce bias. Furthermore, the quality of assets from asset libraries strongly varies. This may cause problems in the data generation tooling. In KI-Absicherung asset quality guidelines were defined for glTF assets such that they can be used in the data pipelines of all data providers involved in the data production.
GL-DP-A-3	Large scale operation	PBR rendering scales very well, as it is usually implemented in batch mode and this shows an 'embarrassing parallelism' as these batch jobs can be distributed on a cluster of machines (e.g. with HPC frameworks like SLURM or Kubernetes). Rendering pipelines build on games-engines, like Unreal are not so affine to deployment in the data center. Because rendering times are usually faster, it is acceptable to some extend to run the data synthesis on few workstation type of machines.
GL-DP-A-4	Control of production pipeline	Control over the production pipeline is essential to garantee the computation of synthetic data that is required for validation and/or training. The set-up and production of scenarios with

Guideline	Name	Description
		specific content, e.g. safety critical situations and selected object classes or properties require a) options to specify these requirements and b) the turn-around times should fast. In the latter case an immediate response is very useful as it enables more 'interactive engineering cycles. It is very benefitial if as many as possible scene and production parameters can be explicitly changed, e.g. in configuation files. This ability to parameterize is a requirement for fully automated pipelines and 'synthesis-in-loop' techniques.

4.8.2.2.5 Toolchain Quality Analysis

Flaws from the toolchain can have a big impact on the training and testing of ML models. For instance, in early versions of our toolchain the 2d bounding boxes were not very tight fitting thereby limiting the performance that can be achieved in training. As a systematic way to analyze how asset and toolchain properties can influence the training and test result, applying PFMEA (process failure mode and effects analysis) proved useful. We focused our analysis on part that are similar in both toolchains and identified the asset quality as a major risk, which also became evident in data generation.

Guideline	Name	Description
GL-DP-TQA-1	Systematically identifying and evaluating the sensitivity of production errors along the data generation toolchain saves post-production data correction time and effort and may also provide better ML algorithm performance during training and testing	Flaws from the toolchain can have a big impact on the training and testing of ML models. For instance, in early versions of KI-Absicherung's toolchain the 2d bounding boxes were not tightly fitted thereby limiting the performance that can be achieved in training. As a systematic way to analyze how asset and toolchain properties can influence the training and test result, applying PFMEA (Process Failure Mode and Effects analysis) proved useful. We focused our analysis on part that are similar in the data production toolchains from the involved data providers and identified the asset quality as a major risk, which also became evident in data generation.
GL-DP-TQA-2	Automatic testing of data set	Several possible data set flaws identified include: unnatural distribution of positions

Guideline	Name	Description
		of objects/classes, intersection of objects, absence of objects classes (assets). These effects can be tested automatically and an analysis helps improving of the data set production.

4.8.2.3 Guidelines on Data Analysis

4.8.2.3.1 Physical Plausibility

Guideline	Name	Description
GL-DA-PP-1	Production toolchain should produce physically plausible results regarding all relevant effects of the use-case	Depending on the use-case the physical plausibility of all relevant effects have to be considered. For the use case of pedestrian detection in the automotive context we have to ensure that the used material models support all relevant effects of the task. This includes the modeling of the pedestrians themselves but also the background. For the application of camera sensor models the render-tooling should produce physical plausible irradiance values in a defined unit. The plausibility of relevant effects should verified and monitored in the produced data.
GL-DA-PP-2	Role of rendering parameters for as approximation problem	An approximation of physical effects, like light propagation and object behavior is in principle possible with recent state-of-the-art simulation and rendering techniques. However, this comes with a very high cost in required processing times. In practice all rendering and simulation engine are providing approximations of physical effects. When specific situations are tested it is not always required to render images with highest realistic quality. However, the generation of images of specific real sensors requires the ability to simualte the characteristic of these sensor and this requires the generation of best posible physical approximation. It is therefore advisable to have a simulation system that is able to produce this highest quality if needed and be reduced in quality to trade in more images at lesser quality if required.

4.8.2.3.2 Dataset Properties and Coverage

Guideline	Name	Description
GL-DA-DPC-1	Coverage analysis and comparison based on metadata improves understanding and characterization of dataset content.	To be able to compare datasets and judge their value for specific use-case requires detailed analysis of dataset properties and coverage. Multiple approaches were developed in E2.4.2. Based on the ODD and expert hypothesis a large amount of additional metadata is extracted for the dataset. Such metadata can be used to compare and check the coverage of different datasets in regard to specific parameters in an automated fashion. The analysis of such parameters proofed to be useful for coverage analysis and dataset comparison based on image and object "appearance" and can be a valuable addition to common high-level dataset metrics such as "number of images" or "number of classes".
GL-DA-DPC-2	Human understandable features can be used as feedback to data production to iteratively improve simulation and data-content for the defined use-case.	While "handcrafted" features and metadata for coverage and data analysis are not comprehensive in terms of a CNN feature-space, they are understandable by humans and can therefore directly be used as feedback to data production to improve the simulation and data-content for a specific use-case.
GL-DA-DPC-3	The evaluation of a DNN trained with the synthetic data on a set of different real-world datasets can serve as indirect measurement method.	One rather indirect method to measure dataset coverage and realism of the synthetic data is to evaluate a DNN trained with the synthetic data on a set of different real-world datasets. The evaluation metric mean intersection over union (mIoU) can here be viewed as a quality proxy metric. For one we can examine the performance increase or decrease on real-world datasets over different KI-A Tranches and draw conclusions on the improvement of our data if the performance on the real-world data improved as well. Another method to asses dataset properties is to observe the performance of real-world dataset trained DNNs on the synthetic data, i.e., a huge decrease of performance can indicate an increase in the domain gap of the two datasets.

Guideline	Name	Description
GL-DA-DPC-4	Detailed analysis based on metadata can provide valuable insights into the effect of specific data properties on a detection methods performance.	Typical performance metrics to benchmark detection methods (e.g., mAP) averaging over a whole testset alone are not sufficient to analyze the effect-chain from data production to method behaviour. Metadata from data production and post-processing can be used to correlate data properties to detection performance and to analyze the detection performance for specific semantic dimensions and ranges.
GL-DA-DPC-5	Performing cross-evaluation of a DNN trained with one dataset on another dataset provides and indirect a method to compare both datasets	One rather indirect method to measure dataset coverage and realism of the synthetic data is to evaluate a DNN trained with the synthetic data on a set of different real-world datasets. The evaluation metric mean intersection over union (mIoU) can here be viewed as a quality proxy metric. For one we can examine the performance increase or decrease on real-world datasets over different KI-A Tranches and draw conclusions on the improvement of our data if the performance on the real-world data improved as well. Another method to asses dataset properties is to observe the performance of real-world dataset trained DNNs on the synthetic data, i.e., a huge decrease of performance can indicate an increase in the domain gap of the two datasets. Table 1 shows the cross-evaluation results on different real-world datasets and KI-A Tranches 2, 3, 4 and 7. Performance on real-world datasets improves from KI-A Tranche 2 to Tranche 7 with a small performance decrease on KI-A Tranche 4. See Table 1 and E2.4.3 for further discussion.

Trained Model	Evaluated Dataset (mIoU)								
	Real-World						Synthetic		
	A2D2	**BDD100K**	**Cityscapes**	**IDD**	**KITTI**	**Mapillary Vistas**	**KIA-Tr2**	**KIA-Tr4**	**KIA-Tr7**
A2D2	**78.57141**	60.17681	70.74164	60.52	66.73244	68.18469	68.61105	54.93	63.47
BDD100K	54.89118	**67.77636**	66.0582	64.13	60.6451	67.42464	61.62715	53.75	55.10
Cityscapes	50.40337	58.15999	**81.55987**	61.56	62.69225	64.41166	64.45452	52.11	63.67
IDD	53.41	57.80	62.36	**79.56**	56.95	65.88	60.36	59.36	57.44
KITTI	40.07763	45.0974	38.04616	46.2	**70.41219**	48.79877	51.02358	38.85	43.29
Mapillary Vistas	66.52379	69.26247	76.91137	72.09	70.55438	**80.53372**	71.20947	58.28	70.12
KIA-Tr2	13.68218	18.02433	17.83769	12.73	37.69851	20.90127	**83.46519**	46.72	39.32
KIA-Tr3	35.96	35.42	40.37	31.51	47.03	41.8	80.68	66.95	57.13
KIA-Tr4	25.89	28.8	33.55	30.07	40.28	37.31	67.92	**77.6**	59.89
KIA-Tr7	38.4	36.35	49.83	35.97	48.89	44.07	62.52	58.22	**87.64**

Table 1: Cross-evaluation results on different real-world and synthetic datasets

4.8.2.4 Guidelines on Transfer to Real-World Application

Guideline	Name	Description
GL-TR-1	Performance improvement through systematic closing of gaps between synthetic and real data.	The indirect method GL-DA-DPC-3 gives only a qualitative answer whether on dataset is lacking some properties compared to a baseline dataset. A more guided approach is possible through PLF analysis and projection into, e.g. a PLF space reduced by PCA. With this method it is possible to detect 'missing' data, for example in our work we were able to detect missing 'dark' clothing in the investigated synthetic dataset. For an in-depth discussion pleas see E2.4.3
GL-TR-2	As long as a considerable amout of data from the target domain is available, for semantic segmentation models no approach in domain adaptation leads to significant performance improvements over using fine-tuning, but merely to improved convergence speed during training.	Performance on the target domain, the real world in our case, can be improved if domain adaptation is applied. Our result from applying various domain adaptation approaches for semantic segmentation is that none of them beets fine tuning in terms of the achieved performance. Only the speed of convergence is higher for some of the methods. The following domain adaptation methods were tested in a setup with 3000 frames from the target domain: • fine tuning on the target domain (supervised) • CCSA - classification and contrastive semantic alignment method (weakly supervised): S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, “Unified deep superviseddomain adaptation and generalization,” 2017. • AdaptSegNet (unsupervised): Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker, “Learning to adapt structured output space for semantic segmentation,” 2020.

Guideline	Name	Description
		• FTAdaptSegNet (supervised): FTAdaptSegNet is an extension of AdaptSegNet that we invented in order to allow the mechanism to use labels for target data to the degree to which they exist. • CrDoCo - cross domain consistency (unsupervised): Y.-C. Chen, Y.-Y. Lin, M.-H. Yang, and J.-B. Huang, “Crdoco: Pixel-level domain transfer with cross-domain consistency,” 2020.
GL-TR-3	Limiting the domain (ODD) of a DNN model leads to better results than expanding the ODD and train over a larger dataset	
GL-TR-3	Applying white-box domain adaptation techniques (e.g. error generator) leads to modera performance improvements, however they are easier to apply as DNN based techniques such as GAN.	Application of the Intel error generator on the synthetic data can be understood as target domain adaptation approach with the distinct advantage that its application is deterministic and the visual effect it has on the synthetic images can be fully understood, i.e., compared to black box methods as GANs. However, the domain adaptation is limited to the sensor effects provided by the error generator, i.e., blur, tone-mapping, chromatic aberration and saturation. Application of the error generator with parameter settings extracted from the real-world Cityscapes dataset improves the generalization capability of the DeeplabV3+ DNN measured by cross-evaluation on different datasets significantly as can be seen in Table 2. While performance on the synthetic datasets stayed almost the same the DeeplabV3+ DNN trained with the error generator applied on the images could improve cross-domain performance on all real-world datasets with the most significant gains of around 7% on the Cityscapes dataset.

Trained Model	Evaluated Dataset (mIoU)										
	Real-World						Synthetic				
	A2D2	**BDD100K**	**Cityscapes**	**IDD**	**KITTI**	**Mapillary Vistas**	**KIA-Tr2**	**KIA-Tr4**	**KIA-Tr7**	**GTAV**	**Synscapes**
KIA-Tr3	35.96	35.42	40.37	31.51	47.03	41.8	**80.68**	**66.95**	57.13	28.15	**49.56**
KIA-Tr3 + Error Generator	**40.89**	**40.91**	**47.2**	**37.38**	**49.2**	**45.06**	78.05	66.13	**57.4**	**31.89**	49.05

Table 2: Cross-evaluation results on different real-world and synthetic datasets with and without sensor simulation. See E2.4.3 for further discussion.

4.9 E2.4.6b Final: Guideline für die Bewertung der Übertragbarkeit der Erkenntnisse von synthetischen Daten auf eine reale Anwendung (zur Veröffentlichung)

4.9.1 Formal Classification

Criteria	Classification according to VHB
Type of result	*Document*
Group/Cluster	
Type of content	*Guideline*
Classification level	*INT, LI, PU*

4.9.2 Description of the result

Due to the nature of the subject of this report, the results of E2.4.6b will be merged with those of E2.4.6a (agreed with TP2 partners)

5 AP2.5 - Datengenerierung und Noisy Data

5.1 E2.5.1 Final: nur projektintern für KI Absicherung verfügbar

5.2 E2.5.2 Final: nur projektintern für KI Absicherung verfügbar