Modelling Pedestrians in Autonomous Vehicle Testing

Priisalu, Maria

2023

LUND UNIVERSITY

PO Box 117
221 00 Lund
+46 46-222 00 00

Modelling Pedestrians in Autonomous Vehicle Testing

Modelling Pedestrians in Autonomous Vehicle Testing

by Maria Priisalu

LUND
UNIVERSITY

Thesis for the degree of Doctor of Philosophy in Engineering
Thesis advisors: Prof. Cristian Sminchisescu, Assoc. Prof. Magnus Oskarsson
Faculty opponent: Assoc. Prof. Jörgen Ahlberg

To be presented, with the permission of the Faculty of Engineering of Lund University, for public criticism in the lecture hall (MA:3) at the Centre of Mathematical Sciences on Monday, the 6th of November 2023 at 13:00.

Organization
LUND UNIVERSITY

Centre of Mathematical Sciences
Box 118
SE–221 00 LUND
Sweden

Document name
DOCTORAL DISSERTATION

Date of disputation
2023-11-06

Sponsoring organization

Author(s)
Maria Priisalu

Title and subtitle
Modelling Pedestrians in Autonomous Vehicle Testing

Abstract
Realistic modelling of pedestrians in Autonomous Vehicles (AV)s and AV testing is crucial to avoid lethal collisions in deployment. The majority of AV trajectory forecasting literature do not utilize the motion cues present in 3D human pose because it is hard to gather large datasets of articulated 3D pedestrian motion. In this thesis we discuss the difficulties in data gathering and propose a pedestrian model that overcomes the issues by utilizing various datasets and learning paradigms to learn articulated semantically reasoning pedestrian motion. We show that such learnt pedestrian models can and should be utilized in AV testing, instead of heuristics as in previous work, to test AVs on realistic and hard scenarios. We propose a framework for generating varied AV test scenarios by posing AV test case generation as a visual problem. Finally we provide a method to improve existing articulated human pose forecasting by utilizing individual specific motion cues on the fly. This thesis discusses the difficulties in articulated pedestrian sensing, proposes a pedestrian model to overcome these difficulties showing a direct use of the pedestrian model in AV testing, and shows the possible further improvements to articulated pedestrian motion forecasting should articulated models be utilized in AV trajectory planning. We hope that this work aids in the further development of articulated and semantically reasoning pedestrian models in AV testing and trajectory planning.

Key words
pedestrian sensing, pedestrian forecasting, pedestrian motion synthesis, generative testing, Autonomous Vehicle testing, Reinforcement Learning

Classification system and/or index terms (if any)

Supplementary bibliographical information

Language
English

ISSN and key title
1404-0034. Doctoral Theses in Mathematical Sciences

ISBN
978-91-8039-827-5 (print)
978-91-8039-828-2 (pdf)

Recipient's notes

Number of pages
232

Price

Security classification

I, the undersigned, being the copyright owner of the abstract of the above-mentioned dissertation, hereby grant to all reference sources the permission to publish and disseminate the abstract of the above-mentioned dissertation.

Signature ____________________ Date 2023-10-11

Modelling Pedestrians in Autonomous Vehicle Testing

by Maria Priisalu

LUND
UNIVERSITY

Cover illustration front: An image illustrating a synthetic pedestrian walking around in a 3D pointcloud of the scene. The image is printed with the permission of Ciprian Paduraru.

Funding information: Swedish Foundation for Strategic Research (SSF) Se- mantic Mapping and Visual Navigation for Smart Robots grant no. RIT15-0038. This work was also supported by the European Research Council Consolidator grant SEED, CNCS-UEFISCDI PN-III-P4-ID-PCE-2016-0535 and PCCF-2016-0180, the EU Horizon 2020 Grant DE-ENIGMA.

Faculty of Engineering, Centre of Mathematical Sciences

Doctoral Theses in Mathematical Sciences 2023:4
ISSN: 1404-0034
ISBN: 978-91-8039-827-5 (print)
ISBN: 978-91-8039-828-2 (pdf)
LUFTMA-1083-2023

Printed in Sweden by Media-Tryck, Lund University, Lund 2023

NORDIC SWAN ECOLABEL
Printed matter
3041 0903

Media-Tryck is a Nordic Swan Ecolabel
certified provider of printed material.
Read more about our environmental
work at www.mediatryck.lu.se

MADE IN SWEDEN

To my husband Simon, and my grandmother Juta.

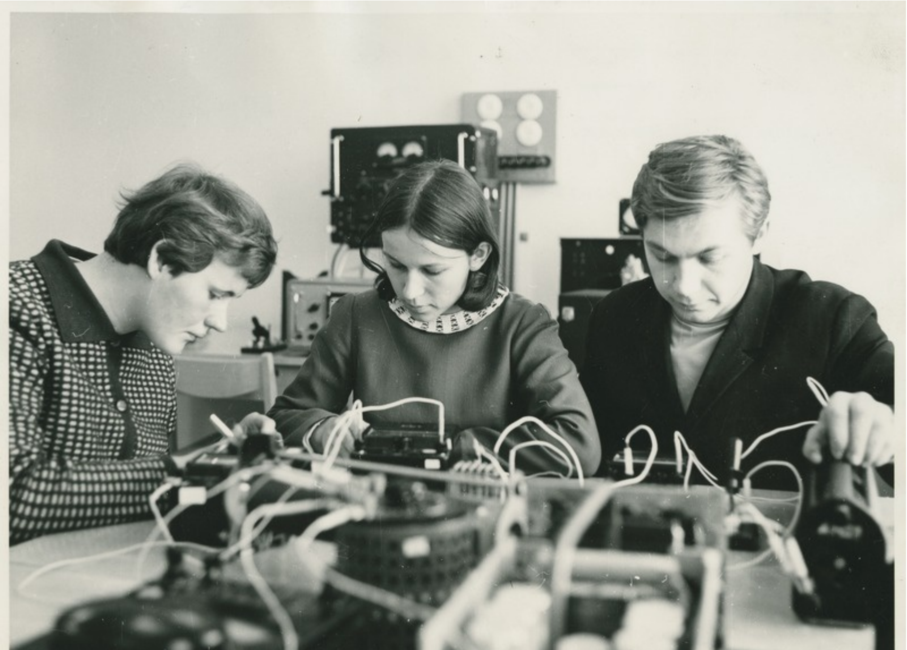


Figure 1: Juta and Aavo Mikomägi performing an electronics lab with a classmate (to the left) at the Tallinn Technical University.

Abstract

Realistic modelling of pedestrians in Autonomous Vehicles (AV)s and AV testing is crucial to avoid lethal collisions in deployment. The majority of AV trajectory forecasting literature do not utilize the motion cues present in 3D human pose because it is hard to gather large datasets of articulated 3D pedestrian motion. In this thesis we discuss the difficulties in data gathering and propose a pedestrian model that overcomes the issues by utilizing various datasets and learning paradigms to learn articulated semantically reasoning pedestrian motion. We show that such learnt pedestrian models can and should be utilized in AV testing, instead of heuristics as in previous work, to test AVs on realistic and hard scenarios. We propose a framework for generating varied AV test scenarios by posing AV test case generation as a visual problem. Finally we provide a method to improve existing articulated human pose forecasting by utilizing individual specific motion cues on the fly. This thesis discusses the difficulties in articulated pedestrian sensing, proposes a pedestrian model to overcome these difficulties showing a direct use of the pedestrian model in AV testing, and shows the possible further improvements to articulated pedestrian motion forecasting should articulated models be utilized in AV trajectory planning. We hope that this work aids in the further development of articulated and semantically reasoning pedestrian models in AV testing and trajectory planning.

Popular Science Summary

Autonomous vehicles (AV) are vehicles that are equipped with a logical unit that can steer the vehicle. The AV is also equipped with sensors. These sensors such as cameras and radars are used to sense objects and traffic participants around the vehicle. To avoid collisions the AV must be able to detect and avoid the traffic participants around itself. To avoid lethal accidents collisions with vulnerable road users in particular should be avoided. Before deployment, the AV must be tested to ensure that it can avoid collisions with pedestrians. This is often done in simulation to avoid putting humans at risk. To perform these simulations we need to model pedestrians realistically. Unfortunately capturing detailed pedestrian behaviour data is hard. We show that it is possible to model pedestrians even with missing data by making use of multiple sources of information. Finally, we show that realistic pedestrian models can be used to effectively test AVs in simulation.

Detecting humans in traffic is hard. Human detection methods that perform the best on standard computer vision benchmarks detect pedestrians poorly in videos gathered onboard a vehicle. This is because pedestrians appear often at a distance, off-centre and are poorly visible in the onboard videos. This is different from standard benchmarks where humans are centred in the images and well visible. Poor detection quality of pedestrians makes any further detailed modelling of pedestrians hard.

A difficulty in modelling traffic from data is that road users often show monotone behaviours. Pedestrians tend to walk on sidewalks, and cars tend to drive on streets. A key question is how to model traffic behaviours that we know exist but rarely observe in data. We present a realistic pedestrian model that avoids collisions even though such behaviour is never seen in the traffic dataset.

AV models tend to model pedestrians as boxes. This is sub-optimal, as the human body pose contains a lot of information about the human's future motion. Just by looking at an image of a pedestrian running we, as humans, can say that the pedestrian will continue running for example. The human pose is not being modelled in AVs because it is hard to capture pedestrian poses in traffic. Our pedestrian model can be used to extend datasets with human poses, and can even be used to create collision scenarios of existing data.

Once a pedestrian's pose is detected, it should be used to forecast the pedestrian's motion such that the AV can plan a safe trajectory. Existing methods for human pose prediction are evaluated on unrealistically short time horizons. Each individual has his/her own unique motion patterns, but this cannot be used by the prediction model when the model only observes less than 0.5s of motion. An individual's motion patterns should be exploited to get an accurate prediction of the individual's future motion. We show that classical statistics methods can be used to personalize human motion prediction on the fly.

Collisions are rare in traffic, and even more so in traffic datasets. Therefore collisions need to be simulated to test the AV's behaviour in scenarios that could lead to a collision. We note that existing simulation methods model pedestrians unrealistically. Our methods make it possible to simulate collisions between AVs and realistic pedestrians in varied traffic scenes, with varied traffic density and with varied traffic participant behaviour.

In conclusion, this thesis shows the difficulties in pedestrian detection and modelling. We propose methods to overcome the shortcomings of existing pedestrian motion models. Finally, we show that it is possible to generate collision scenarios to test AVs with realistic pedestrian models.

List of Publications

I **Semantic Synthesis of Pedestrian Locomotion**

M. Priisalu, C. Paduraru, A.Pirinen, C. Sminchisescu
Asian Conference on Computer Vision (ACCV) 2020

MP was responsible for experiments, analysis and writing the manuscript. AP aided in analysis and writing. CP was responsible for the Waymo dataset, the locomotion agent and visualizations. The paper is based on an idea proposed by CS but developed further by MP and AP. MP proposed and developed the idea of combining RL and supervised learning.

II **Generating Scenarios with Diverse Pedestrian Behaviors for Autonomous Vehicle Testing**
M. Priisalu, A.Pirinen, C. Paduraru, C. Sminchisescu
Conference on Robot Learning (CoRL) 2021

MP was responsible for experiments, analysis and writing the manuscript. AP aided in analysis and writing. CP was responsible for final visualization of selected trajectories. The paper is based on an idea proposed by CS but developed further by MP and AP, formalized by MP as a previously unstudied optimization problem.

III **Varied Realistic Autonomous Vehicle Collision Scenario Generation**

M. Priisalu, C. Paduraru, C. Sminchisescu
Scandinavian Conference on Image Analysis (SCIA) 2023

MP carried out all of the experiments, analysis and writing. CP developed a partial API to CARLA for future developments. The paper's initial idea arose from discussions of paper II, developed further and formalized as a MARL by MP.

IV **Personalized Human Pose Forecasting**

M. Priisalu, T. Kronvall, C. Sminchisescu
Manuscript in preparation

MP is responsible for experiments, analysis and writing of the manuscript. TD has aided in the experiments and analysis. The paper is based on an idea proposed by MP, but developed further by MP and TK.

V **Semantic and Articulated Pedestrian Sensing Onboard a Moving Vehicle**

M. Priisalu
Technical report. arXiv preprint arXiv:2309.06313

All work, writing and analysis done by MP.

Acknowledgements

I would like to thank my advisor Cristian Sminchisescu for introducing me to the field of pedestrian modelling and Reinforcement Learning, and my co-advisor Magnus Oskarsson for his guidance and support during the last two years of my Ph.D. studies. I would also like to thank Carl Olsson, Anders Robertsson, Gabrielle Flood, Kalle Åström, Niels Christian Overgaard, Mikael Nilsson and Maria Sandsten for their belief in me and help during the latter years of my PhD studies.

I am extremely happy to have had the joy to closely collaborate with Aleksis Pirinen, who taught me to calmly continue analysis no matter how dire the experimental results may be. Thank you Aleksis also for introducing me to seeing humans as RL agents, this has made my research and everyday life much more enjoyable. I am also grateful for having had the pleasure to discuss research and matters of life with the rest of the tight-knit research family consisting of David Nilsson, Erik Gärtner, Martin Trimmel, Ted Kronvall and Henning Petzka. The joint discussions have been well balanced by the calmness of David, the enthusiasm of Erik, the optimism of Martin, the philosophical ways of Ted and the pedagogical ways of Henning. I would in particular like to thank my co-author Ted for his insightful discussions on research during our meetings, and entertaining discussions on all other matters in between meetings. Finally, I am thankful for the work that my co-author Ciprian Paduraru put into paper 1 in particular.

I have had the luck to share my office with Gabrielle Flood in the early years of my studies. I will never forget the late nights, that at times turned into early mornings, spent together working on our respective research. I am happy to have always been able to share this tough journey with Gabrielle. Thank you Gabrielle for all the support and for introducing me to biking as a past time. I would also like to thank Ida Arvidsson for always reminding me of the wonderful things in life beyond the four walls of the department. I am always cheered up by just seeing Ida down the corridor. It has been a pleasure to have been a doctoral student together with Ida and Gabrielle.

I am thankful that through my studies I met my dear friends Alexia Papalazarou and Tien Truong. I am thankful for your endless support and belief in me. Thank you for being my cheerleaders and for the joyful time spent together.

I am happy to have had so many wonderful encounters through my PhD studies. Thank you to the numerical analysis group, especially Fatemeh Mohammadi, Azahar Monge, Lea Versbach, and Wafaa Assad for the always enjoyable fika discussions. I would like to also thank the PhD's on the second floor in particular David Gillsjö, Martin Ahrnbom, Marcus Valtonen Örnhag, Mårten Nilsson, Mats Bylund, Anna Gummeson, Olof Rubin, Ivar Persson, and Felix Augustsson for making the Ph.D. corridor always feel like a little community. I would like to thank Anna for her cheerful discussions about biology, for sharing my sense of style, and for teaching me to downhill ski at SCIA. Finally, I would like to thank Olof for being a great office mate, and for happily discussing pointless things with me.

I would like to thank the dance communities Black Diamond and Estilo Faixa Sweden for bringing me joy and reminding me of the world outside of academia. I would also like to thank my friends for understanding when I was a poor friend due to the heavy burden of my Ph.D. studies.

Finally, I would like to thank the people whom I am the most grateful to, my family. I would probably have given up or worked myself to death without all of you. I would in particular like to thank my husband, Simon for helping me through the roughest times and for bearing with me even on days when I have been a living zombie. I would also like to thank Juta for always taking the time to listen to all of my worries and for always providing me with words of wisdom no matter what.

Funding

This work was supported by the Swedish Foundation for Strategic Research project - Semantic Mapping and Visual Navigation for Smart Robots- (grant no. ROT15-0038). This work was also supported by the European Research Council Consolidator grant SEED, CNCS-UEFISCDI PN-III-P4-ID-PCE-2016-0535 and PCCF-2016-0180, the EU Horizon 2020 Grant DE-ENIGMA.

Contents

Chapter 1

Introduction

An autonomous vehicle (AV) is a vehicle equipped with sensors and a reasoning unit that decides the AV's motion given the sensors' observation of the world. The reasoning unit should use the sensor's data to move the vehicle in traffic in a safe and legal manner.

AVs promise to save lives by reducing the number of traffic accidents occurring due to human error. To do so AVs must at first be able to reason at least as well as human drivers. But how should the AV's driving performance be evaluated?

Pedestrians and bicyclists are particularly vulnerable in traffic, and collisions with them even at relatively low speeds could be lethal. Therefore it is crucial that an AV avoids collisions with pedestrians. The AV's collision avoidance ability with vulnerable road users (VRU) cannot be tested in real physical experiments due to the risk it poses to VRUs. Therefore there is a need for simulated tests that evaluate the AV's ability to avoid collisions with VRUs.

Simulating test cases poses a number of problems. Firstly simulating realistic and varied sensor data is hard. Secondly, the simulated traffic behavior should be realistic. Thirdly to be time-efficient the simulations should only contain the test cases that are hard for the AV. We present methods that use realistic human models to test the behavior of AVs in near-collision scenarios in papers II and III.

Modeling pedestrians is hard due to the stochastic motion of humans, as seen in Fig.1.1. The data appears stochastic because the intents (i.e. why the pedestrian chose this particular trajectory and speed) of pedestrians even in ground truth data is in general unknown. And even if the intents of the pedestrians are known then each individual has unique dynamics due to his/her anatomy and behavior traits. Paper I proposes a pedestrian model that takes into consideration the surrounding traffic and human dynamics.

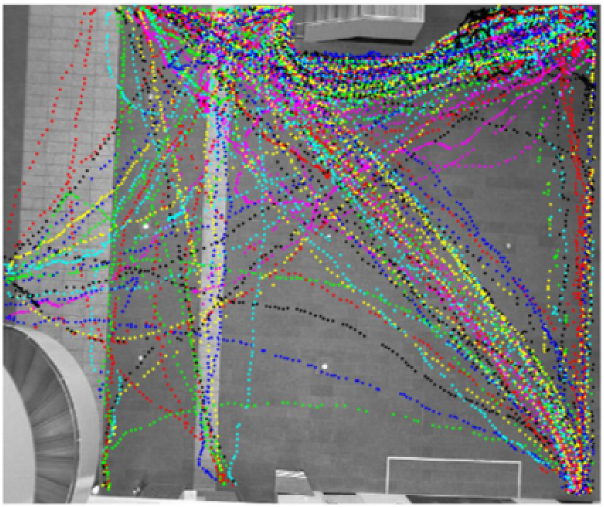


Figure 1.1: Pedestrian trajectories as seen from above in a study hall. The trajectories run crisscross through the room with some general motion trends. Pedestrian trajectories in the Edinburgh Informatics Forum Pedestrian Database, image from [1]. Reprinted with author's permission.

A large number of AV trajectory planning models [2–35] treat pedestrians as boxes. This however omits the motion cues available in the human pose, as seen in Fig.1.2. Humans are often treated as boxes because it is hard to extract human poses in traffic data and because bounding boxes are common when modeling vehicles [34, 36–44]. To remedy this sometimes human head direction, action or 2D pose (in the image plane) are used to provide partial motion cues in pedestrian behavior prediction [45–52], and just a few models use 3D pose [53, 54] to predict if a pedestrian will cross the street or not. In a perfect world, we would wish to obtain a dataset of pedestrian poses gathered in the wild with all of the surroundings that may affect a pedestrian's motion through traffic. Currently, there is no such dataset as exact human-pose estimation requires motion capture technology that has largely been confined to indoor settings and requires the use of markers. We discuss the difficulties in recovering articulated human motion in paper V.

If a perfectly labeled dataset of human poses in traffic was made available then this would enable AVs to foresee pedestrian motion more exactly with pedestrian pose forecasting models. The majority of human pose forecasting models foresee the average motion of an average individual. But poses also contain each individual person's unique motion patterns. These unique patterns even enable person identification from gait [55–58]. As observations of an individual accumulate the motion models can be re-calibrated to fit the specific individual's rather than an average human's motion. This is done in paper IV.

The scientific contribution of this thesis is presented in the papers I-V. Gathering articulated 3D pedestrian motion in traffic, to preserve the underlying semantic relations in

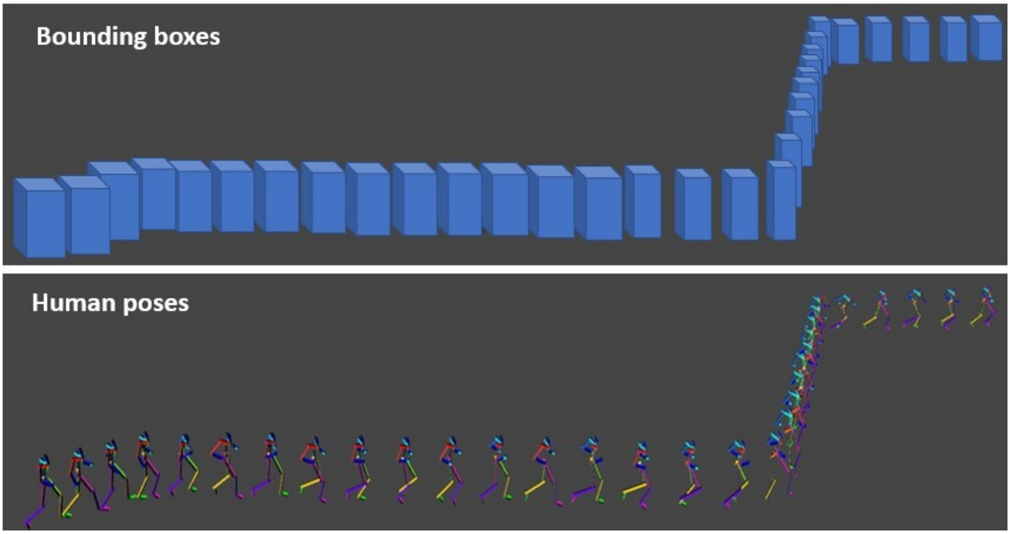


Figure 1.2: It is much harder to foresee the pedestrian's future speed and direction when observing the bounding boxes (above), than when observing the poses (below).

the motion, is hard as discussed in paper v. In paper i a semantically reasoning, collision-avoiding, and articulated pedestrian model is presented, by combining different learning methods and datasets. Papers ii and iii show that realistic pedestrian models such as the one from paper i can and should be used to generate realistic and hard test cases for AVs, in particular when the number of pedestrians increases. In paper iv a method to personalize articulated human motion forecasting on the fly with timeseries analysis is presented. Altogether the thesis discusses why articulated pedestrian sensing is hard in traffic in paper v, proposes a method to remedy the lack of articulated in-traffic pedestrian data in paper i, shows that learned pedestrian models can and should be used in AV testing in papers ii and iii, and proposes improvements to articulated pedestrian forecasting models motivated by typical interactions in traffic in paper iv. We argue that articulated and semantically reasoning pedestrian modeling can improve pedestrian motion forecasting accuracy in AV planning and testing. We hope that this thesis will aid in the further development of articulated and semantically reasoning pedestrian modeling in AV path planning and testing, as improved pedestrian motion forecasting may save lives in AV deployment.

We provide an overview of the necessary background knowledge for the papers i-v with comments on recent developments in the respective background fields in chapters §3-§6. The methods developed in this thesis are Machine Learning (ML) and statistical models. Therefore a basic introduction to ML and timeseries analysis with a focus on the methods covered in the thesis are provided in §3, providing a basis to papers i-v. More specifically the ML methods utilized in papers i,iv and v that sense and model pedestrians are introduced in §4. We discuss the different subtasks of pedestrian sensing that are commonly solved in the computer vision community. To clarify what is meant by an AV in papers II and III a short introduction to the basic building blocks of an AV is provided in §5 followed by a discussion on the difficulties of sensor simulations of AVs and difficulties in AV testing relevant to papers II and III. Finally, a short introduction to 3D reconstruction focusing on the used methods is given in §6, in detail discussed in paper V. 3D reconstruction is essential in pedestrian and AV modeling and testing because the AV needs to know the distance to objects when planning its motion. Distance is however lost when taking 2D images of the 3D world, so this needs to be recovered using 3D reconstruction methods.

Chapter 2

Research Questions

This thesis largely concentrates on pedestrian sensing and modelling; in particular in the testing of AVs. Pedestrians, vehicles and bicycles behave differently in traffic and this should be reflected in their models. A large amount of research effort is put into modelling vehicles, and often a model developed to forecast vehicles is simply extended to model pedestrians. However, the motion of pedestrians does not resemble cars, this should be reflected in the model. Such methods also ignore the additional information available in the pedestrian's semantic surroundings and pose relevant for pedestrian motion prediction. Therefore the first research question can be formulated as follows.

[**RQ1**] What affects a pedestrian's motion, and how can we model this in pedestrian forecasting models utilized in AVs and AV testing?

In paper v we discuss how pedestrians can be sensed and based on this in paper i we propose a pedestrian model that models human dynamics in interactions with the scene. In papers ii and iii we note that there is a gap between our knowledge of human motion and the models used in AV testing. In papers ii and iii we show that state-of-the-art (STOTA) pedestrian models should be used in AV testing. The thesis argues that pedestrian pose should be modelled instead of bounding boxes in AV's path planning, as poses provide rich motion cues needed to improve motion planning. Secondly, human motion in traffic is affected by the environment semantics, and this should be included in the ideal model of a pedestrian. Paper iv proposes a general method to personalize human motion forecasts to a specific individual's dynamics.

The majority of decisions taken in traffic are based on semantics and geometry. For example, cars tend to drive in the centre of the road and aim to avoid other vehicles. One of the key research questions is the following.

[**RQ2**] How can we model the effect of semantics and geometry on pedestrians?

[**RQ3**] How should a scene be modelled to allow for the simulation of semantically reasoning pedestrians and AVs?

It should be noted that geometry and semantics are inherently interlinked. By detecting the 3D shape of a car it is easier to detect that the object is a car, while vice versa observing a car it is easier to reconstruct it as we may have prior knowledge of a typical car's shape. In traffic, an object's motion is dictated by its semantic class and the semantic objects surrounding it. Therefore semantics must be included when modelling the decision-making process of the different traffic participants. In papers I-III and V the scene is a 3D model labelled by semantics and RGB. This explicit modelling of semantics allows for the evaluation of safety measures during simulation. More compact representations exist for AV decision-making but these are less applicable for simulations. This is an important trade-off because AVs generate large amounts of data that should be utilized to improve existing models.

Traffic data is special because we as humans have a lot of understanding and prior knowledge of the rules of traffic and what may influence other traffic participants. But the majority of motion in traffic is mundane and uniform. The majority of traffic interactions are safe, and thus they provide little knowledge to artificial agents on what an unsafe situation may look like. Therefore there is a need to utilize our prior knowledge in modelling to avoid relearning known rules and to model critical but less likely situations rather than the average behavior of traffic. The following research questions are central to this thesis.

[**RQ4**] How can we utilize prior knowledge in models?

[**RQ5**] How can we learn from datasets that are not representative of the true variability?

[**RQ6**] How can we optimize models for specific behavior rather than the average behavior observed in the data?

In the papers I-III, Reinforcement Learning (RL) is utilized to extrapolate beyond the existing data and to learn from critical situations not present in the data while utilizing prior knowledge. In paper IV individual-specific available information is utilized to personalize average motion forecasts. The average motion forecast in paper IV can be seen as prior knowledge while the personalization is learnt.

[**RQ7**] How can we balance learning with prior knowledge?

The balance between prior knowledge and learning boils down to the classical bias-variance trade-off because prior knowledge inherently introduces bias while reducing variance and thus speeding up learning. This is tackled differently in the different papers. In paper I prior knowledge and learning are balanced by combining RL and supervised learning. In papers II and III a prior is used to speed up early learning. In paper IV the prior is adapted with classical statistical methods.

We treat scenario-based AV testing to allow testing of the full pipeline of AV. The following is a relevant question to ask.

[**RQ8**] What variations in traffic data may affect an AV's safety? How can we test an AV to ensure safety under varying conditions?

An AV when utilized should be able to avoid collisions in any traffic layout with any traffic density, with any traffic behavior and under any visual qualities. Therefore it should be tested with varied road-layouts, traffic density, traffic behavior and visual qualities. The papers ii and iii propose methods to do this, omitting only visual qualities as this falls beyond the breadth of this work. In particular traffic participant behavior (in particular of pedestrians) is under-studied in previous work on scenario-based test case generation.

In traffic, the same driver or pedestrian may exhibit different behaviors from day to day depending on the traffic density, the goal location and his/her mood (depending on how hurried, tired or upset the person is). Therefore to ensure safe travels an AV must react correctly independently of how its traffic co-participants are behaving. One of the central questions of this thesis can be formulated as follows.

[**RQ9**] How can we model different pedestrian behaviors in traffic?

Papers ii and iii present AV testing methods that allow for any goal-driven pedestrian behavior models to be utilized. Paper i allows different behaviors to be learned in safe traffic situations and in near-collision scenarios. Paper iv models individual human dynamics.

It is hard to detect articulated humans in real traffic data, therefore humans are often modelled by bounding boxes. This gives rise to the following research question.

[**RQ10**] How can we learn to model pedestrian motion when only partial or noisy labels are available?

In paper v it is seen that label errors aggregate when sensing humans in a pipeline and in 3D reconstructions. Therefore paper i utilizes bounding boxes to learn pedestrian behavior in traffic scenes and a separately trained human dynamics model to capture realistic human dynamics. The model in paper i can be used to augment existing traffic datasets with articulated pedestrians. In papers ii and iii in a generated collision, in the spirit of RL, it is unclear what decisions along the trajectory of a pedestrian led to a collision. Nonetheless, with just the knowledge that a collision occurred, we can learn how to place pedestrians such that collisions occur with an AV. In paper iv, not all data is available when learning is initialized, so the model is updated as new data becomes available. In general, when a large enough dataset of correct labels is available supervised learning allows for precise and efficient training of a model. If this is not the case we must resort to other methods such as RL with a weaker and indirect (we don't know which timestep in the sequence provided the signal) learning signal.

Finally, the thesis deals with sample-efficient AV testing. It is impossible to test the AV on all possible scenarios that may occur as there are so many possible outcomes. Therefore we can concentrate tests on scenarios that the AV finds hard. One of the research questions dealt with in the thesis is the following.

[**RQ11**] How can we efficiently test a driving system in realistic scenarios?

In paper v it is seen that errors accumulate when sensing pedestrians in a pipeline, making the final results too noisy to be useful. This illustrates the importance of testing the full pipeline of the AV (i.e. testing the outcome of steering at the result of a given sensor input) as done in papers ii and iii to detect accumulating errors. In traffic, realism includes but is not limited to the distribution of traffic participants in the scene, the behavior and motion of all traffic participants in reaction to one another, and the realism of the sensors. When testing AVs we are not interested in the most difficult scenarios if they are impossible in real life, but the difficult scenarios that are in fact likely to occur. Therefore instead of modelling co-participants in traffic as adversarial as in previous works, in papers ii and iii we learn which realistic placements of traffic members can result in collisions with an AV. AV testing therefore needs to answer the following question.

[**RQ12**] How can we balance difficulty and realism in AV test cases?

In papers ii and iii we suggest balancing the two by incorporating factors that promote realism in the prior and reward of the test case generator. In paper ii we note that test case generation is a constrained optimization problem and by introducing further constraints on the AV we can ensure that difficult cases are found. It is important to introduce realistic constraints, such as constraining the AV's observation of pedestrians by ensuring the pedestrians are occluded in a realistic manner. Paper iii presents a general framework that is easily extendable with realistic constraints on the AV. Data augmentation, as proposed in papers i, ii and iii, allows the creation of test cases from real data, providing high sensor realism. Because realistic traffic data is often similar, in that most traffic participants follow traffic rules, it is particularly difficult to generate realistic, varied and critical traffic participant behaviors.

Chapter 3

Machine Learning and Statistics

Machine Learning (ML) is a field of mathematical models that adapt to data. The process of a model adapting to data is referred to as learning. ML has gained popularity because the world contains a large number of phenomena that can be modeled but we have not yet found an analytic model that captures the behavior of the system so we must approximate the behavior from the available data instead. A large number of the ML methods are function approximators. In difference to classical statistical methods ML provides highly flexible models (i.e. often with more parameters) but with fewer statistical guarantees. The higher number of parameters means that more data is needed to fit ML models than classical statistical models in general. With digitalization, dataset sizes have grown, and dataset growth within visual problems has further been brought about by crowd-sourcing and crawling the net to generate datasets such as [59, 60]. In the current age of data models with billions of parameters [61–64] can be trained, allowing complex behaviors such as speech or vision to be modeled.

ML can be roughly divided into three paradigms *supervised learning*, *unsupervised learning*, and *Reinforcement Learning* (RL). The central problem in *supervised learning* is that of regression analysis to find a function approximator that can model the output or dependent variables given input or independent variables. In *supervised learning* we assume that the function values i.e. *labels* are known for a number of data points in the input variables. The data may contain errors and noise so it is treated as a set of random variables. The relationship between the input and output variables is often non-linear and unknown. *Unsupervised learning* methods learn the structure of the data without access to labels or the need for human annotators to provide explicit feedback on the model's performance. Finally, RL instead of learning from a dataset, interacts with a system of interest and learns to operate the system from interactions. An example of this is a model that learns to drive a car by controlling the wheel, the gas, and the brake pedals. In more classical terms RL can be seen as a discrete-time system identification problem in automatic control.

The problem central to ML is the following; given some parametrization Θ of the approximating function f_Θ we wish to minimize a measure of fitness, the *loss* J, in expectation

$$\min_{\Theta} \mathbb{E}[J(f_\Theta, \boldsymbol{p})], \tag{3.1}$$

where $\boldsymbol{p}$ is a data point (in RL $\boldsymbol{p}$ is one roll-out or trajectory). ML is different from classical statistics utilizes models with more parameters than classical statistical models, and in general assumes that the underlying structure in the data is unknown. Therefore ML does not have the same statistical guarantees on model fitness as classical statistical methods but allows the models to be more expressive in practice. However, when data is scarce classical models can outperform ML models as fewer parameters need to be estimated.

3.1 Deep Learning

This section will provide an introduction to supervised and RL. Independently of the chosen paradigm *Artificial Neural Networks* (ANNs) [65] are popular function approximators in ML. This is because ANNs are flexible, can model large datasets well, and have proven in practice to be optimizable in a reasonable amount of time. Inspired by the brain, an ANN consists of small units called *perceptrons*. Each perceptron is connected to a number of neurons structured into layers. The goal is to learn the weights of the connections. A perceptron weighs the M-dimensional input vector $\mathbf{u} \in \mathcal{R}^M$ with the weight vector $\mathbf{w} \in \mathcal{R}^M$ before applying a non-linearity f_a known as the *activation function*. The perceptron output is

$$y_p = f_a(\mathbf{u}^T\mathbf{w} + w_0), \tag{3.2}$$

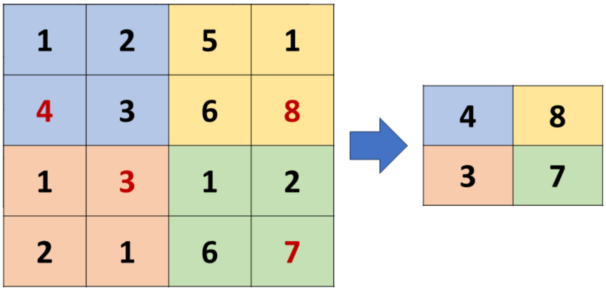
where $w_0 \in \mathcal{R}$ is a learnable intercept known as the bias. For a network consisting of N_l layers the input $\mathbf{u}$ of perceptrons in layer l, $2 \leq l \leq N_l$ typically consists of the concatenated outputs y_p of the perceptrons in layer $l - 1$. If all of the weights $\mathbf{w}$ in a layer are non-zero then the layer is called a *fully connected layer*. The outputs of the layers are referred to as features. Features describe task-specific characteristics of the data. Features found in higher layers of the networks are often more abstract than features from lower layers that are closer to the model input. Deep neural networks are ANNs with a large number (at least more than 10) of layers and therefore contain abstract features. It is common that the last layers of the ANN, containing the most abstract features, are fully connected layers allowing for the full interaction of knowledge among features. The last layer of the network often contains a special non-linearity to obtain outputs that are probabilities. In particular, to ensure that the sum of outputs from the last layer is one, the *soft-max* function can be applied. The

i-th element of the softmax function value is given by

$$\mathrm{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{N_z} \exp(z_j)}, \tag{3.3}$$

where $\mathbf{z} \in \mathcal{R}^{N_z}$ is a vector with i-th element z_i, and N_z is the size of the vector.

3.1.1 Convolutional Neural Networks



1	2	5	1
4	3	6	8
1	3	1	2
2	1	6	7

4	8
3	7

Figure 3.1: Max pooling of a 16 by 16 image by a 4 by 4 max-pooling window. The image is divided up into partitions the size of the filter (the different partitions are in blue, yellow, red, and green) and the largest value within each partition (in red text) is returned. The result is the 4 by 4 image to the right.

In images, objects need to be detected independently of their placement in the image. Therefore translationally invariant convolutional filters are commonly used to constrain the number of weights in each layer. A convolutional filter K, also known as the kernel, of size $w_k \times h_k \times D$ is applied to an image (possibly the output of a previous layer) I of size $W \times H \times D$. The convolution is given by

$$\mathbf{Y}(m, n) = \sum_{i=1}^{w_k} \sum_{j=1}^{h_k} \sum_{k=1}^{D} I(m - i, n - j, k) K(i, j, k). \tag{3.4}$$

The application of the filter on the whole image in (3.4) is called a *convolution*. A *Convolutional Neural Network*(CNN) is an ANN that contains convolutional layers. In a convolutional layer, the perceptrons share $w_k \times h_k \times D$ weights to model all of the connections with the $W \times H \times D$ dimensional input, where in general $w_k \times h_k < W \times H$. For very large datasets it has been shown that *Transformers* (recently introduced ANN architecture) outperform convolutions, as they learn this convolutional weight sharing, but are not confined to it [66]. Dilation can be used to increase the visual field of a filter without increasing the number of weights. *Dilational convolution* with stride $s_r \in \mathcal{Z}^+$ is given as,

$$\mathbf{Y}(m, n) = \sum_{i=1}^{w_k} \sum_{j=1}^{h_k} \sum_{k=1}^{D} I(m - is_r, n - js_r, k) K(i, j, k). \tag{3.5}$$

Typical activation functions utilized in the network are the sigmoid function and the *Rectified Linear Unit* defined as $ReLU(x) = max(x, 0)$ [67]. ReLU is a simple piece-wise linear function but with enough layers (possibly infinitely many) a network with only ReLU activation function can approximate any continuous function.

The *sigmoid function* $\sigma(x) = (1 + e^{-x})^{-1}$ is a smooth growing curve around the origin and approximates the Heaviside step function when approaching positive or negative infinity.

Pooling is used to extract the results of the most important features from an image. *Max-pooling* [68] subsamples an image by selecting maximal values from partitions of an image as seen in Fig.3.1. The final result, to the right, is reduced in size by a factor equal to the size of the filter.

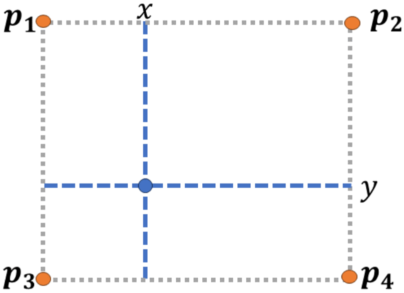


Figure 3.2: Given four points on a grid $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$ with known function values, we wish to estimate the function value at the point (x, y)).

When a dense label is required it may be necessary to upsample images. This can for example be done by bilinear interpolation. A *bilinear interpolation* of a point (x, y) between four points $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$ on a pixel grid as shown in Fig.3.2 is found by first interpolating along the x axis

$$F_{i,1}(x) = \frac{x - p_{1,x}}{p_{2,x} - p_{1,x}} F(\mathbf{p}_1) - \frac{x - p_{2,x}}{p_{2,x} - p_{1,x}} F(\mathbf{p}_2), \tag{3.6}$$

$$F_{i,2}(x) = \frac{x - p_{3,x}}{p_{4,x} - p_{3,x}} F(\mathbf{p}_3) - \frac{x - p_{4,x}}{p_{4,x} - p_{3,x}} F(\mathbf{p}_4), \tag{3.7}$$

where $(F(\mathbf{p}_1), F(\mathbf{p}_2), F(\mathbf{p}_3), F(\mathbf{p}_4))$ are the function values of the function F that is being interpolated at points $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4)$ and the x and y coordinates of the points are $(p_{1,x}, p_{2,x}, p_{3,x}, p_{4,x})$ and $(p_{1,y}, p_{2,y}, p_{3,y}, p_{4,y})$ respectively. Then finally the interpolation is given by also interpolating along the y axis

$$F_i(x, y) = \frac{y - p_{1,y}}{p_{3,y} - p_{1,y}} F_{i,1}(x) - \frac{y - p_{3,y}}{p_{3,y} - p_{1,y}} F_{i,2}(x). \tag{3.8}$$

3.1.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN)s [65] are used to model temporal relations because they allow for weight sharing across time steps. RNN units contain self-connections, meaning that the RNN layer's output $\mathbf{h}_t \in \mathcal{R}^{M_h}$ becomes the unit's input at the next timestep,

$$h_t = f_a(\mathbf{u}^T\mathbf{w} + \mathbf{h}_{t-1}^T\hat{\mathbf{w}} + w_0), t = 1 \dots T_h \tag{3.9}$$

where $\hat{\mathbf{w}} \in \mathcal{R}^{M_h}$ are the weights of the self-connection, and T_h is the length of the RNN's memory. During training, the gradients of $\hat{\mathbf{w}}$ might explode or diminish. To avoid this the *Long Short Term Memory* (LSTM) can be used instead. One LTSM unit maintains its memory in the cell state $c_t \in \mathcal{R}$, and the cell states of all of the M_c LSTM units in a layer are gathered in a vector $\mathbf{c}_t \in \mathcal{R}^{M_c}$. The LSTM unit controls information flow with three gates; the input gate g_t^u- controls information flow from $\mathbf{u}$, forget gate g_t^c - controls information flow from $\mathbf{c}_{t-1}$ and output gate g_t^h- controls how much of the information from the cell state $\mathbf{c}_t$ is output. The gates $g_t^u, g_t^c, g_t^h \in (0, 1)$ are all given by

$$g_t^u = \sigma(\mathbf{u}^T\mathbf{w}_u + \mathbf{c}_{t-1}^T\hat{\mathbf{w}}_u + w_{u,0}) \tag{3.10}$$

$$g_t^c = \sigma(\mathbf{u}^T\mathbf{w}_c + \mathbf{c}_{t-1}^T\hat{\mathbf{w}}_c + w_{c,0}) \tag{3.11}$$

$$g_t^h = \sigma(\mathbf{u}^T\mathbf{w}_h + \mathbf{c}_{t-1}^T\hat{\mathbf{w}}_h + w_{h,0}) \tag{3.12}$$

where $\mathbf{w}_u, \mathbf{w}_c, \mathbf{w}_h \in \mathcal{R}^M$ and $\hat{\mathbf{w}}_u, \hat{\mathbf{w}}_c, \hat{\mathbf{w}}_h \in \mathcal{R}^{M_c}$ are the weights $w_{u,0}, w_{c,0}, w_{h,0} \in R$ are biases of the input, forget and output gate respectively and σ is the sigmoid function. An update variable $\hat{c}_t$ is used to calculate the update step by the current timestep's input,

$$\hat{c}_t = \tanh(\mathbf{u}^T\mathbf{w}_{\hat{c}} + \mathbf{c}_{t-1}^T\hat{\mathbf{w}}_{\hat{c}} + w_{\hat{c},0}), \tag{3.13}$$

where $\mathbf{w}_c \in \mathcal{R}^M$ and $\hat{\mathbf{w}}_c \in \mathcal{R}^{M_c}$ are weights and $w_{\hat{c},0} \in \mathcal{R}$ is a bias, and *tanh* is the hyperbolic tangent. The update variable $\hat{c}_t$ is used with gaiting to update the cell state,

$$c_t = g_t^c c_{t-1} + g_t^u \hat{c}_t. \tag{3.14}$$

Finally, LSTM's value is given by gating g_t^h of the cell state,

$$h_t = g_t^h \tanh(c_t). \tag{3.15}$$

Recently Transformers have gained popularity over RNNs because they can learn to pay attention to occurrences over a larger time-span, and therefore effectively have a longer memory.

3.1.3 Training

ANN weights are found through numeric optimization. The loss function of an ANN has in general an unknown shape, most importantly the loss is not guaranteed to be a convex function, so it is common to use Stochastic Gradient Descent (SGD) [65] to optimize it. The dataset is often too large to fit in memory so we cannot calculate the exact gradient of the expectation in (3.1). Therefore iterative batch-based optimization methods, such as SGD, are used. The gradient of the loss is estimated by a finite randomly drawn batch of the data that is used to update the parameters Θ_t of iteration $t \in \mathcal{Z}^+$ by

$$\Theta_{t+1} = \Theta_t - \eta \frac{1}{N_b} \sum_{i=1}^{N_b} \frac{\partial}{\partial \Theta} J(f_\Theta, \boldsymbol{p}_i), \tag{3.16}$$

where $\eta \in \mathcal{R}$ is the learning rate, deciding the step size of the optimization, and $\boldsymbol{p}_i$ is i-th point in the t-th batch of size N_b. Iterating through the whole dataset once in batches is known as one epoch, it is common to train ANNs for multiple epochs.

Since ANNs often have a large number of parameters, learning may progress at different speeds for different parameters. Therefore it may be necessary to adapt the learning rate η for the different parameters. This can be achieved by the popular Adam (name derived from adaptive moment estimation) [69] that utilizes first and second moment estimates of the gradient to adapt the learning rate of different parameters. By treating the batch estimate of the partial derivative $g_{t,k} = \frac{1}{N_b} \sum_{i=1}^{N_b} \frac{\partial}{\partial \Theta_k} J(f_\Theta, \boldsymbol{p}_i)$ of the k-th parameter where $1 \leq k \leq N_\Theta$ at optimization step t as a stochastic variable, the first two moments of this random variable can be estimated by the moving averages $m_{t,k}$ (mean) and $v_{t,k}$(uncentered variance). Moving averages are used because $g_{t,k}$ is assumed to be non-stationary. The moving averages $m_{t,k} \leftarrow \beta_1 m_{t-1,k} + (1 - \beta_1) g_{t,k}$ and $v_{t,k} \leftarrow \beta_2 v_{t-1,k} + (1 - \beta_2) g_{t,k}^2$ have update rates β_1 and β_2 respectively. The estimated $m_{t,k}$ and $v_{t,k}$ are biased estimates. The unbiased estimates $\hat{m}_{t,k} = m_{t-1}/(1 - \beta_1^t)$ and $\hat{v}_{t,k} = v_{t-1}/(1 - \beta_2^t)$ are used to scale the learning rate by the estimated mean $\hat{m}_{t,k}$ and the inverse square root of the uncentered variance $(\hat{v}_t)^{-1/2}$. The learning rate scaling ratio $\hat{m}_{t,k}/\sqrt{\hat{v}_{t,k}}$ can be treated in some sense as a signal-to-noise ratio (SNR), the larger the SNR the larger the learning rate. The full algorithm is given in Algorithm 1, where ϵ is the machine epsilon added to avoid numeric instabilities.

The network biases w_0 are zero-initialized and the weights $\mathbf{w}_l$ of layer l are initialized randomly [70] according to the uniform Glorot initialization (also known as Xavier initialization) given as

$$\mathbf{w}_l \sim U\left(-\sqrt{\frac{6}{N_u + N_h}}, \sqrt{\frac{6}{N_u + N_h}} \right), \tag{3.17}$$

Algorithm 1 The Adam Optimization Algorithm

```
$m_{0,k} = 0$, $v_{0,k} = 0, t = 0$
while $\Theta_{t,k}$ not converged do
    $t = t + 1$
    Calculate $g_{t,k} = \frac{1}{N_b} \sum_{i=1}^{N_b} \frac{\partial}{\partial \Theta_k} J(f_\Theta, \boldsymbol{p}_i)$
    $m_{t,k} \leftarrow \beta_1 m_{t-1,k} + (1 - \beta_1) g_{t,k}$
    $v_{t,k} \leftarrow \beta_2 v_{t-1,k} + (1 - \beta_2) g_{t,k}^2$
    $\hat{m}_{t,k} \leftarrow \frac{m_{t-1}}{(1-\beta_1^t)}$
    $\hat{v}_{t,k} \leftarrow \frac{v_{t-1}}{(1-\beta_2^t)}$
    $\Theta_{t,k} = \Theta_{t-1,k} - \eta \frac{\hat{m}_{t,k}}{\sqrt{\hat{v}_{t,k}+\epsilon}}$
end while
```

where U is the uniform distribution, N_u is the size of the input vector $\mathbf{u}$ to the ANN's l-th layer, and N_h is the number of perceptrons in the l-th layer.

The gradient $\nabla_\Theta J(f_\Theta, \boldsymbol{p}_i)$ is calculated layer by layer with the *back propagation algorithm* [65], by first calculating the gradient of the last layer and then moving backward in the network the gradients of all of the weights can be calculated efficiently reusing the gradients of deeper layers due to the chain rule.

It should be noted that in difference to classical statistical models, ANNs are often over-parameterized. Studies have shown that over-parametrization and large datasets [71–73] are key components to attain high accuracy ANNs. Still *early stopping* [68] is common practice to avoid overfitting to the training dataset and thus not being able to generalize to new unseen data. To this end, the dataset used is commonly divided into the *training*, *validation*, and *test set*. The model is trained on the training set and tested on the validation set throughout the training process. Once the validation error does not improve significantly or even starts to increase the training is terminated and the model with the best performance on the validation set is evaluated on the previously withheld test set to give an estimate of how the model would perform on new unseen data. Any *hyperparameters* (parameters that need to be optimized separately from Θ such as learning rate, or model's architecture) are tuned based on the model's performance on the validation set.

3.2 Supervised Learning

In the supervised learning setting the goal is to learn a function that can model a labeled dataset. In the parametric case *Maximum Likelihood Estimation* can be used to find model parameters that maximize the model's likelihood. The *likelihood* $f_l(\Theta; \mathcal{D})$ is the model parameter's Θ probability density function conditioned on the given the datapoints $(\boldsymbol{p}, \mathbf{y}) \in \mathcal{D}$ with labels $\mathbf{y}$. The *Maximum Likelihood Estimate* (MLE) is found by

$$\Theta_{MLE} = \arg\max_{\Theta} f_l(\Theta; \mathcal{D}), \tag{3.18}$$

where the parameters Θ that describe the dataset $\mathcal{D}$ with the highest probability according to the model are found. Assuming that the data labels $\mathbf{y} \in \mathcal{R}^{N_y}$ follow the deterministic function f_Θ but are observed with additive Gaussian noise

$$\mathbf{y} \sim \mathcal{N}\left(f_\Theta(\boldsymbol{p}), \boldsymbol{\Sigma}\right), \tag{3.19}$$

where $\boldsymbol{\Sigma}$ is the covariance matrix then the likelihood f_l is given following notation of [74] as,

$$f_l(\Theta; \mathcal{D}) = \prod_{i=1}^{N_\mathcal{D}} \mathcal{N}\left(\mathbf{y}_i | f_\Theta(\boldsymbol{p}_i), \boldsymbol{\Sigma}\right). \tag{3.20}$$

Then the *negative log-likelihood* L is given as

$$L(\Theta; \mathcal{D}) = \sum_{i=1}^{N_\mathcal{D}} \left(\mathbf{y}_i - f_\Theta(\boldsymbol{p}_i)\right)^T \boldsymbol{\Sigma}^{-1} \left(\mathbf{y}_i - f_\Theta(\boldsymbol{p}_i)\right) + \mathbf{c}, \tag{3.21}$$

where $\mathbf{c} = \frac{N_\mathcal{D}}{2}\left(N_y \ln(2\pi) + \ln(|\boldsymbol{\Sigma}|)\right)$ is a constant. If the additive errors are identical and independently distributed (i.i.d.) that is $\boldsymbol{\Sigma} = \sigma_l^2 \mathbf{I}$, where $\sigma_l \in \mathcal{R}^+$ and $\mathbf{I}$ is an identity matrix, then the MLE estimate is given by the least squares loss,

$$\Theta_{MLE} = \arg\min L(\Theta; \mathcal{D}) = \arg\min \sum_{i=1}^{N_\mathcal{D}} \|\mathbf{y}_i - f_\Theta(\boldsymbol{p}_i)\|^2. \tag{3.22}$$

This will be utilized in Paper I to combine the supervised and RL objectives.

3.2.1 Object Detection

Object detection is the task of placing a *bounding box* around an object of interest in an image. The bounding box should also be given a class label among a set of semantically meaningful class labels (such as "car", "traffic light" etc). This is a supervised learning task requiring human-annotated bounding boxes and labels.

The *Region Convolutional Neural Network* (RCNN) [75] performs object detection in two stages; region proposal where regions of interest are extracted from images and object classification by evaluating convolutional features per region of interest that are then classified. *Faster RCNN* (FRCNN) [76] is a popular object detection network because it attains high

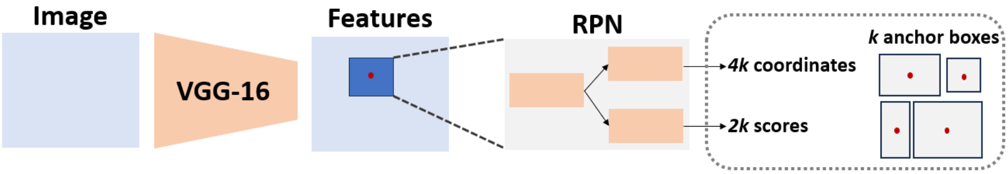


Figure 3.3: The FRCNN architecture. From left to right. Given an image (in blue) VGG-16 convolutional layers are applied (in orange) resulting in convolutional features (in light blue). The RPN is applied as a sliding window (in dark blue, with the anchor point in red) on the convolutional features. The RPN (in light grey) consists of a shared fully connected layer followed by a regression head producing bounding box coordinates and a classification head producing object-class scores. The RPN is evaluated for k anchor boxes (in white) with different scales and aspect ratios centered around the anchor point (in red).

classification accuracy with good throughput. Faster RCNN stems from RCNN. FRCNN improves upon RCNN by utilizing a *Region Proposal Network* (RPN) that shares features with the classifier. The convolutional features are borrowed from the popular *VGG-16* [77] a popular CNN for image classification (providing a class label per image). The RPN is a small network that slides across the convolutional features and consists of two heads one regressing over the region of interest borders and the other classifying the object see Fig.3.3. When sliding across the convolutional features the center of RPN's field of view is fitted with *k anchor boxes* with different scale and aspect ratios. The regressor head learns the exact placement of the *k* bounding boxes and the classifier evaluates the class belonging of the *k* anchor boxes.

The full loss of the model consists of a per-class log-likelihood loss for the object classification and a robust loss for the bounding box placement and size that is active only if the object classification is correct. The classification is considered correct when the ratio of *intersection over union* (iou.) area of the ground truth and predicted bounding box is at least 0.7, and the class label is correct.

The full model is trained iteratively by first training the class regression and object detector separately. Then the detector's convolutional weights are shared and frozen, while the regressor is trained. Followed by a step where only the weights specific to the detector are trained.

3.2.2 Semantic Segmentation

Semantic segmentation is the task of labeling each pixel in an image with a class label. STOTA semantic segmentation is performed by CNNs [78–82]. Semantic segmentation requires heavy labeling from human annotators and is therefore particularly difficult in video where each frame should be labeled. Weak supervision has become a popular method to overcome the need for large amounts of annotations in video segmentation [79, 81, 83, 84]. The *Gated Recurrent Flow Propagation*(GRFP) net [79] is an optical flow-based video seg-

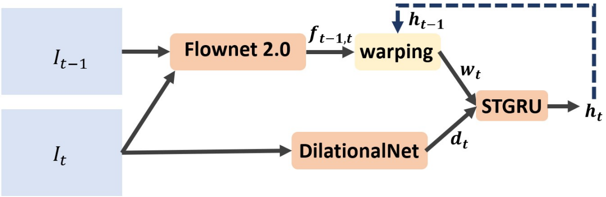


Figure 3.4: The GRFP architecture: Optical flow $\mathbf{f}_{t-1,t}$ is calculated from frames I_t and I_{t-1} and used to warp the previous frame's semantic segmentation $\mathbf{h}_{t-1}$. The STGRU calculates the current frame's segmentation $\mathbf{h}_t$ from the warped segmentation $\mathbf{w}_t$ and the current frame's per frame segmentation.

mentation network. Optical flow is an estimate of the motion of objects from one frame to the next in a video sequence. The GRFP model uses the convolutional image segmentation network *DilationalNet-10* [80] to segment the frame I_t at timestep $t \in [2, \ldots T_G]$, where T_G is the length of the video sequence, producing a segmentation $\mathbf{d}_t$. *DilationalNet-10* is a CNN that uses dilated convolutions to avoid loss of resolution in layers and to make dense label prediction easier. The GRFP net uses also a CNN, the *FlowNet 2.0* [85], to estimate the optical flow $\mathbf{f}_{t-1,t}$ between frames $t-1$ and t. The flow is used to warp the segmentation $\mathbf{h}_{t-1}$ of the previous frame $t-1$. The resulting warped segmentation $\mathbf{w}_t$ is an estimate of the expected segmentation at timestep t. It promotes temporal smoothness along frames, and $\mathbf{d}_t$ provides new information in case of occlusions and large motions. The two segmentations $\mathbf{d}_t$ and $\mathbf{w}_t$ are input to a *Spatio-Temporal Transformer Gated Recurrent Unit* (STGRU) that outputs the GRFP's estimated semantic segmentation $\mathbf{h}_t$, see Fig.3.4. The STGRU's gating is designed to utilize $\mathbf{d}_t$ when occlusions occur, and otherwise $\mathbf{w}_t$ for temporal smoothness. To obtain the probability that the pixel at x, y belongs to a class the $\mathbf{h}_t$ is placed through a softmax function, producing $\mathbf{z}_{x,y}$. During training, STGRU allows warping to the timestep with ground truth labels. The network is trained to minimize the negative log-likelihood

$$L(\Theta; \mathcal{D}) = -\sum_{I_t \in \mathcal{D}} \sum_{x,y} \log(p(\mathbf{z}_{x,y} = \mathbf{c}_{x,y} | \Theta, I_t)), \tag{3.23}$$

where $\mathbf{c}_{x,y}$ are the ground truth labels, and the probability is obtained as the softmax estimated probability $\mathbf{z}_{x,y} = \mathbf{c}_{x,y}$.

Instance segmentation is a related task, where the objective is to identify the pixels that belong to individual objects in images that contain multiple instances of the objects. In traffic, this would most importantly allow us to identify individual pedestrians and vehicles on a pixel level. *Path Aggregation Network for Instance Segmentation* (PANNET) [86] is based on the FRCNN architecture. It adds an additional mask network, as shown in Fig.3.5, on top of the convolutional features, that estimates the mask of the individual object within a bounding box. PANNET uses Pyramid features [87], a filtered concatenation of low and upsampled high-level features, with an additional bottom-up feature fusion network

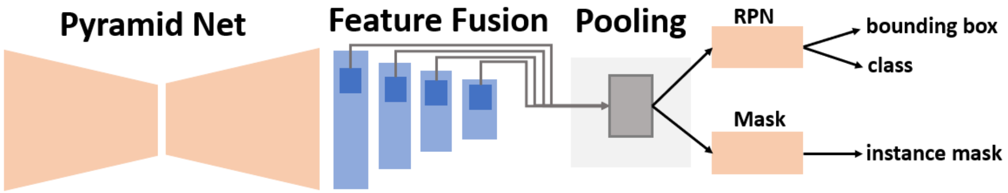


Figure 3.5: Architecture of PANNET. *From left to right* Pyramid features are extracted by Pyramid Net (in orange) followed by additional bottom-up feature fusion (in light blue) and multi-resolution pooling (in grey of dark blue features from multiple scales). A sliding network with two branches is applied to the pooled features. The RPN branch extracts a bounding box and a class and the masking branch extracts pixel-level instance masking.

followed by multi-resolution pooling to obtain feature details from low-level features and semantic context from high-level features. The full model architecture is shown in Fig.3.5.

Recently the connection between instance segmentation in video (segmenting all instances of an object in each frame) and the similar task of multi-object tracking (where the individuals have to be re-identified or tracked from one frame to the next) has gained attention [83, 88, 89]. Segmentation networks trained on large and varied semi-automatically collected datasets (1 labeled billion object masks) are able to generalize zero shot to other related tasks [90]. Segmentation is becoming less supervised in both human labeling and in label classes [88, 91–96]. The latter is obtained by language model encoded semantic labels which enable multi-class labeling of objects and segmentation with previously unseen class labels during inference. The trend of reducing the need for human annotations can also be seen in object detectors [94, 97] and 3D segmentation [98, 99]. Recently the term *Computer Vision in the Wild* has been coined to the study the transferability of the popular natural language supervised visual models across computer vision tasks [100].

Pointcloud segmentation is a task where a class label should be given to all points in a 3D pointcloud. One popular architecture for pointcloud segmentation is the *Pointnet++* [101]. Pointnet++ consists of layers of *set abstractions* (SA). A SA operates on groups of points centered around a centroid point. A SA consists of three processes; subsampling of the centroids, grouping points around the centroid, and pointnet feature evaluation. The centroids are selected by iterative farthest point sampling. Then for each centroid point, all points within a radius distance are grouped to belong to the centroid. Finally, the centroid and a feature vector calculated from its group are concatenated and output. The *Pointnet* [102] *feature vector* of the variable length n_p group members $(\mathbf{p}_1, \ldots, \mathbf{p}_n)$ is found by

$$f_p(\mathbf{p}_1, \ldots, \mathbf{p}_{n_p}) = f_\gamma \left(\max_{i=1,\ldots,n_p} f_h(\mathbf{p}_i) \right), \tag{3.24}$$

where f_γ and f_h are learnable multi-layer perceptrons and f_p is invariant to permutations and can approximate any continuous set function. Features across layers or varying size group balls are concatenated into features to make the network robust to changes in pointcloud density. To propagate dense labels to all points, feature propagation is performed by interpolating final labels between points according to distance. The model is trained with the cross entropy loss. Let the estimated probability of a point belonging to class c_i be $p(c_i|X,\Theta)$, with N_i points belonging to the class out of a total of N points. Then assuming the class probabilities are independent the likelihood is given by,

$$L(\Theta) = -\log\left(\prod_{i=1}^{M_C} p(c_i|X,\Theta)^{N_i}\right) = -\sum_{i=1}^{M_C} N_i \log(p(c_i|X,\Theta)), \tag{3.25}$$

where M_C is the number of classes. The *cross entropy loss* (CE) is obtained by normalizing by the number of datapoints N,

$$CE(\Theta) = -\sum_{i=1}^{M_C} \frac{N_i}{N} \log(p(c_i|X,\Theta)) = -\sum_{i=1}^{M_C} p(c_i) \log(p(c_i|X,\Theta)). \tag{3.26}$$

Note that this is the same loss as (3.23) simply with a multiplicative constant difference.

3.3 Reinforcement Learning

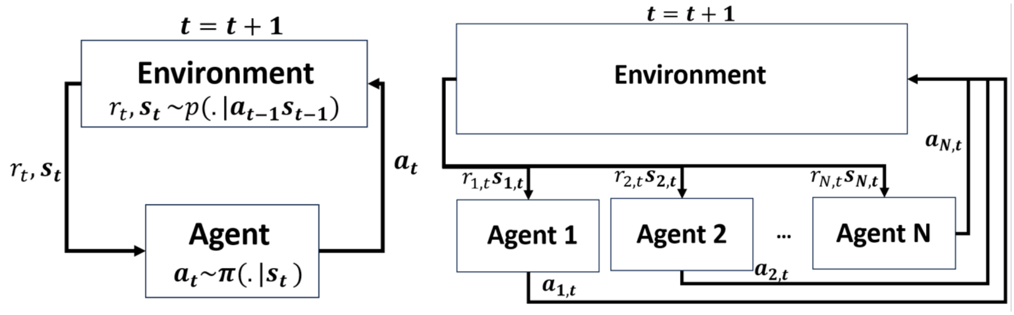


Figure 3.6: *To the left:* A Markov Decision Process: In a state $\mathbf{s}_t$ an agent takes actions $\mathbf{a}_t$ according to its policy π, and obtains from the environment its subsequent state $\mathbf{s}_t$ and reward r_t. The agent's policy π is chosen such that cumulative future reward is maximized. This is an RL problem when the environment dynamics p are unknown. *To the right:* A Multi Agent RL game. Each of the N agents takes actions based on its individual states. The agents observe each other (possibly partially) through their states. Agents optimize their individual rewards which may have different objectives.

An agent interacts with a world it does not know the dynamics of, RL learns how the agent should act to perform a given task in this unknown world. More formally RL solves an unknown *Markov Decision Process* (MDP) [103]. An MDP as visualized in Fig.3.6 *Left* consists of a set of states $\mathbf{s} \in \mathcal{R}^{N_s}$ and actions $\mathbf{a} \in \mathcal{R}^{N_a}$. The agent can transition along states by taking actions. The world is assumed to be stochastic so state transitions are defined by an unknown probability function $\mathbf{s}_{t+1} \sim p(.|\mathbf{s}_t, \mathbf{a}_t)$. Each state-action transition is evaluated by a reward $r_{t+1} = f_r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) \in \mathcal{R}$, where f_r is a real valued function.

We wish to find a policy $\mathbf{a}_t \sim \pi(.|\mathbf{s}_t)$ that maps from states to actions and describes the behavior of the agent that maximizes the expected cumulative reward,

$$J_{RL} = \max_{\pi} \mathbb{E}\left[\sum_{t=1}^{T_s} \gamma^t r_t | \mathbf{s}_0 \sim \mu, \mathbf{a}_{t-1} \sim \pi(.|\mathbf{s}_{t-1}), \mathbf{s}_t \sim p(.|\mathbf{s}_{t-1}, \mathbf{a}_{t-1})\right], \tag{3.27}$$

where μ is the distribution of initial states, and γ is a discounting factor to encourage early high rewards. An agent's discounted expected cumulative future reward when following policy π from state $\mathbf{s}_t$ is known as the value function V_π, and can be expressed through the Bellman equations,

$$V_\pi(\mathbf{s}_t) = \mathbb{E}\left[r_{t+1} + \gamma V_\pi(\mathbf{s}_{t+1}) | \mathbf{a}_t \sim \pi(.|\mathbf{s}_t), \mathbf{s}_{t+1} \sim p(.|\mathbf{s}_t, \mathbf{a}_t)\right]. \tag{3.28}$$

Similarly, the Q value is defined as the discounted expected cumulative future reward when following policy π from state $\mathbf{s}_t$ after performing the action $\mathbf{a}_t$,

$$Q_\pi(\mathbf{s}_t, \mathbf{a}_t) = \mathbb{E}\left[r_{t+1} + \gamma V_\pi(\mathbf{s}_{t+1}) | \mathbf{s}_{t+1} \sim p(.|\mathbf{s}_t, \mathbf{a}_t)\right]. \tag{3.29}$$

When (3.28) and are evaluated for the optimal policy π^* we refer to the equations as the *Bellman optimality equations.*

Model-based RL methods estimate the unknown world dynamics p from the agent's interactions with the environment and then solve the Bellman optimality equations using methods for solving MDPs [103]. Model-based RL method's performance depends on the estimation of the world dynamics p, and often it is hard to obtain a good estimate of p because world interactions are seldom uniform in states and actions. Model-based RL methods need special care when utilized with non-stationary world dynamics. *Model-free* methods instead directly estimate the optimal policy π^*, these methods are known as *policy gradient methods*, or the optimal value functions V_{π^*} or Q_{π^*}, known as *action value methods.* Most notable examples of action value methods in visual problems is *deep Q-learning* [104–106] and of policy-based methods *Proximal Policy Optimization* (PPO) [107–109] and of methods that learn both the policy and value functions, known Action-Critic methods, *Asynchronous Advantage Actor Critic* (A3C) [110] is among the most well known methods.

Policy Gradient Methods

Policy gradient (PG) methods [103] parameterize the policy π_Θ and directly optimize the loss (3.27). The expectation in (3.27) is given as,

$$\int_Q \mu(\mathbf{s}_0) \sum_{t=0}^{T_s-1} \gamma^t r_{t+1} \prod_{k=0}^{t} \pi_\Theta(\mathbf{a}_k|\mathbf{s}_k) p(\mathbf{s}_{k+1}|\mathbf{s}_k, \mathbf{a}_k) d\tau, \tag{3.30}$$

where $\mathbf{s} = (\mathbf{s}_1 \dots \mathbf{s}_{T_s})^T$, $\mathbf{a} = (\mathbf{a}_0 \dots \mathbf{a}_{T_s})^T$, Q is the region of integration over $\mathbf{s}_0, \mathbf{s}, \mathbf{a}$ and $d\tau = d\mathbf{s}_0 d\mathbf{s} d\mathbf{a}$. The Policy Gradient Theorem [103] simplifies the derivative of (3.27) to

$$\int_Q \mu(\mathbf{s}_0) \sum_{t=0}^{T_s-1} \gamma^t r_{t+1} \frac{\partial \pi_\Theta(\mathbf{a}_t|\mathbf{s}_t)}{\partial \Theta} p(\mathbf{s}_{t+1}|\mathbf{a}_t, \mathbf{s}_t) \prod_{k=0}^{t-1} \pi_\Theta(\mathbf{a}_k|\mathbf{s}_k) p(\mathbf{s}_{k+1}|\mathbf{a}_k, \mathbf{s}_k) d\tau. \quad (3.31)$$

The expectation in (3.31) cannot be estimated with *Monte Carlo* (MC) simulation because the action distribution is missing. Utilizing the standard manipulation, below, MC estimation can be performed,

$$\frac{\partial \pi_\Theta(\mathbf{a}|\mathbf{s})}{\partial \Theta} = \pi_\Theta(\mathbf{a}|\mathbf{s}) \left(\frac{1}{\pi_\Theta(\mathbf{a}|\mathbf{s})} \frac{\partial \pi_\Theta(\mathbf{a}|\mathbf{s})}{\partial \Theta} \right) = \pi_\Theta(\mathbf{a}|\mathbf{s}) \frac{\partial \log(\pi_\Theta(\mathbf{a}|\mathbf{s}))}{\partial \Theta}. \quad (3.32)$$

Now a *Markov Chain Monte Carlo* (MCMC) estimate of the expectation over $\mathbf{s}_0, \mathbf{s}, \mathbf{a}$ can be formed, and the gradient of the loss, as below, is used in SGD to optimize the parameters,

$$\frac{\partial}{\partial \Theta} J_{RL} \propto \mathbb{E}\left[\sum_{t=0}^{T_s-1} \gamma^t r_{t+1} \frac{\partial}{\partial \Theta} \log(\pi_\Theta(\mathbf{a}_t|\mathbf{s}_t)) \right] \approx \sum_{i=1}^{M_s} \sum_{t=0}^{T_s-1} \gamma^t r^i_{t+1} \frac{\partial \log(\pi_\Theta(\mathbf{a}^i_t|\mathbf{s}^i_t))}{\partial \Theta} \quad (3.33)$$

where M_s samples of the full trajectory are sampled, with the i-th trajectory being

$$(\mathbf{s}^i_0, \mathbf{a}^i_0, \mathbf{s}^i_1, \mathbf{a}^i_1, \dots, \mathbf{a}^i_{T_s}, \mathbf{s}^i_{T_s+1}). \quad (3.34)$$

The REINFORCE [111] algorithm samples N_s trajectories with the current estimate of the parametric policy, then updates the policy parameters with the gradient estimate (3.33). The policy updates with sampling are repeated until convergence.

PPO optimizes a surrogate objective that is a lower bound of (3.27). The lower bound is optimized instead to avoid taking too large gradient steps into areas of unexplored state-action space. The A3C utilizes the advantage instead of the reward in (3.27). The advantage $A_{\pi^*}(\mathbf{s}_t, \mathbf{a}_t) = Q_{\pi^*}(\mathbf{s}_t, \mathbf{a}_t) - V_{\pi^*}(\mathbf{s}_t)$ is estimated by a neural network. In Q-learning the Q-function is estimated from data with the Bellman equations and an optimal deterministic policy is extracted [103] by $\pi^*(\mathbf{s}_t) = \arg\max_{\mathbf{a}} Q_{\pi^*}(\mathbf{s}_t, \mathbf{a})$.

RL in Traffic

A number of problems in traffic are particularly well suited to be modeled by RL because in traffic we often encounter intelligent agents that interact with their environment. A traffic participant can be seen as an RL agent that aims to reach a goal under some preferences described by its reward. In general traffic participants are not ominous and their observation of the world is not necessarily the true state of the world. Therefore in general RL models of traffic participants solve a *Partially Observable Markov Decision Process* (POMDP). In a POMDP an agent must learn the world dynamics in the presence of missing data (for example due to occlusions or incorrectly estimated velocities and distances).

The interaction of traffic participants can be described as a *Multi Agent Reinforcement learning (MARL) Game* [112, 113], see Fig.3.6 *right*, where all traffic participants interact with one another but ultimately each traffic participant is trying to reach their goals according to their individual priorities that can be expressed as a reward. Each traffic participant may have their own individual dynamics. Because the traffic agents interact with one another their optimal behavior depends on the behavior of the other traffic participants. The problem solution is non-stationary, therefore we often seek a *Nash equilibrium* - a state where no agent can improve their utility by selecting a different action. Traffic as a general sum game in a POMDP is particularly difficult to find the solutions to [114]. Because of the non-stationarity MARL specific methods exist, but a number of them are empirically outperformed by PPO [109].

RL versus Supervised Learning

Both RL and supervised learning have their strengths and weaknesses. In RL MC sampling is used to estimate expectations. This together with the credit assignment problem in RL leads to less sample-efficient learning than in supervised learning. By the credit assignment problem, we mean that the reward does not clearly clarify which states must be visited to reach a final reward-giving state (for example realizing that one must open a door to enter a new room). But MC sampling also provides robustness as a large variety of state-action pairs are observed during training when compared to supervised learning.

Combining RL and supervised learning to obtain robust but sample efficient learning is presented in Paper 1, and in other applications [115, 116]. *Inverse Reinforcement Learning* (IRL) [117, 118] refers to an alternative method of learning from expert demonstrations with RL. In IRL the task is to recover a reward function that can explain the demonstrations, after which the regular RL problem is solved. This is in general a hard problem to solve and may require solving the RL problem multiple times. *Imitation Learning* (IL) is a set of often supervised methods in the RL setting that learn a policy that imitates the demonstrations. *behavior Cloning* (BC) [119] is an example of an IL algorithm that uses supervised learning to model the state-action distribution of the policy from demonstrations. *Generative Adversarial Imitation Learning* (GAIL) [120] is also an imitation learning method that performs imitation learning with a *Generative Adversarial Network* (GAN) [121]. The learned policy is a generator network that is evaluated by a discriminator network that discriminates between demonstrations and samples from the learned policy.

3.4 Timeseries Analysis

A *stochastic process* is defined by [122] as a family of random variables $Y_1, Y_2 \ldots$. The index of the random variables will be referred to as time. Timeseries analysis is concerned with modeling stationary stochastic processes. Here we will treat only real valued one dimensional stationary processes with zero mean. A *wide sense stationary process* is defined by [122] as a process with a constant and finite mean, a finite variance, and with an auto-covariance between Y_t and Y_k that can be expressed as function $c(t-k)$ of the time difference $t-k$. An *autoregressive* (AR) model of order P of a process assumes that the next random variable is a sum of the past P random variables and a white noise. The AR model is given as

$$Y_t = -\sum_{i=1}^{P} a_i Y_{t-i} + \epsilon_t, \tag{3.35}$$

where a_i are model parameters and $\epsilon_t \sim \mathcal{N}(0, \sigma_w)$ is normal distributed noise with variance σ_w. It is common that we only have one realization of the timeseries, so if the timeseries is stationary then we can use $(y_1, y_2 \ldots)$ to estimate a_i and σ_w. Note that ϵ_t are assumed to be independent. To find the parameters $\boldsymbol{\Theta}_a = (a_1, \ldots, a_P)$ we take the expectation of (3.35) after multiplication with Y_{t-k}, we get,

$$\mathbb{E}\left[\epsilon_t, Y_{t-k}\right] = \mathbb{E}\left[Y_t, Y_{t-k}\right] + \sum_{i=1}^{P} a_i \mathbb{E}\left[Y_{t-i} Y_{t-k}\right], \tag{3.36}$$

Since ϵ_t and Y_{t-k} are uncorrelated, and Y_t are zero-mean, (3.36) simplifies to the *Yule-Walker equations*,

$$\sigma_w^2 \delta(k) = c(k) + \sum_{i=1}^{P} a_i c(k-i), \tag{3.37}$$

where $\delta(k)$ is the *Kronecker delta* function ($\delta(k) = 1$ when $k = 0$, and 0 otherwise for $k \in \mathcal{Z}$). After evaluating (3.37) for various k a system of linear equations can be obtained from which model parameters $\boldsymbol{\Theta}_a$ can be found given the estimated *auto-correlation* $c(k)$ values.

Prediction Error identification Method(PEM) [123] is performed to obtain online estimates of the parameters $\boldsymbol{\Theta}_a$. The one-step prediction error is minimized, here using the L2 norm and a forgetting factor λ to allow for time-varying parameters,

$$\boldsymbol{\Theta}_n^* = \arg\min_{\boldsymbol{\Theta}_n} \sum_{t=1}^{n} \lambda^{n-t} \|y_t - \hat{y}_t\|_2^2, \tag{3.38}$$

where $\hat{y}_t = \phi_t^T \boldsymbol{\Theta}_t$, and $\phi_t = [-y_{t-1}, \ldots - y_{t-P}]^T$, and $\boldsymbol{\Theta}_t$ are the current estimates of the parameters $\boldsymbol{\Theta}_a$. The least squares estimate

$$\boldsymbol{\Theta}_n = \left[\sum_{t=1}^{n} \lambda^{n-t} \phi_t \phi_t^T \right]^{-1} \sum_{t=1}^{n} \lambda^{n-t} \phi_t y_t, \tag{3.39}$$

is obtained by setting the gradient of (3.38) with respect to $\boldsymbol{\Theta}$ to zero. Let $\mathbf{P}_n^{-1} = \sum_{t=1}^{n} \lambda^{n-t} \phi_t \phi_t^T$ and $\mathbf{f}_n = \sum_{t=1}^{n} \lambda^{n-t} \phi_t y_t$, then

$$\boldsymbol{\Theta}_n = \mathbf{P}_n \mathbf{f}_n, \tag{3.40}$$

$$\mathbf{P}_n^{-1} = \lambda \mathbf{P}_{n-1}^{-1} + \phi_n \phi_n^T, \tag{3.41}$$

$$\mathbf{f}_n = \lambda \mathbf{f}_{n-1} + \phi_n y_n. \tag{3.42}$$

Following [123] we can rewrite (3.40) as

$$\boldsymbol{\Theta}_n = \mathbf{P}_n \left(\lambda \mathbf{f}_{n-1} + \phi_n y_n \right) = \mathbf{P}_n \left(\lambda \mathbf{P}_{n-1}^{-1} \boldsymbol{\Theta}_{n-1} + \phi_n y_n \right), \tag{3.43}$$

$$= \mathbf{P}_n \left(\left[\mathbf{P}_n^{-1} - \phi_n \phi_n^T \right] \boldsymbol{\Theta}_{n-1} + \phi_n y_n \right), \tag{3.44}$$

$$= \boldsymbol{\Theta}_{n-1} + \mathbf{P}_n \phi_n \left(y_n - \phi_n^T \boldsymbol{\Theta}_{n-1} \right), \tag{3.45}$$

resulting in an online parameter update, where $\mathbf{P}_n$'s update (3.41) can efficiently be inverted with the *matrix inversion lemma* giving

$$\mathbf{P}_n = \frac{1}{\lambda} \mathbf{P}_{n-1} \left(\mathbf{I} - \frac{\phi_n \phi_n^T \mathbf{P}_{n-1}}{\phi_n^T \mathbf{P}_{n-1} \phi_n + \lambda} \right), \tag{3.46}$$

where $\mathbf{I}$ is an identity matrix. The update simplifies further when considering the product,

$$\mathbf{P}_n \phi_n = \frac{\mathbf{P}_{n-1} \phi_n}{\phi_n^T \mathbf{P}_{n-1} \phi_n + \lambda}, \tag{3.47}$$

in the update of the parameters (3.45).

Chapter 4

Human Motion Modelling

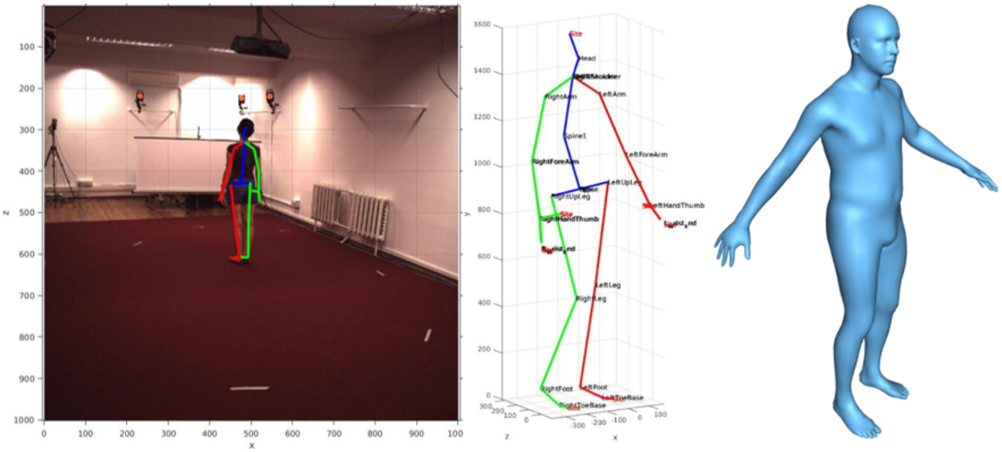


Figure 4.1: *To the left:* 2D projection of the human skeleton onto the image plane results in a 2D skeleton that depends on the camera view. *Middle:* Human pose formed by joints and skeleton connecting the joints in 3D. *To the right:* A dense human body model, an example of GHUM in the zero-pose.

Human motion modeling is divided between different subdomains that model different variables of the human motion of which we will cover articulated human motion modeling and human trajectory modeling. *Human motion forecasting* is concerned with capturing the motion of the limbs and joints of a moving human, while *trajectory forecasting* aims to model the factors that affect a walking human's path or trajectory in traffic. In paper I these two fields are combined. Articulated human motion forecasting requires access to measurements of human joint or body part positions over time see Fig.4.1. It is not easy to obtain 3D reconstructions of human bodies because human body shape is deformable in difference to a large number of static objects, and therefore we often cannot utilize the classical 3D reconstruction methods to 3D reconstruct humans from videos. Instead in each frame, the shape of the human must be estimated. To this end, there are a number of specialized models.

It is not obvious how to model human bodies in 3D. There are a number of alternatives for example learnable *dense body surface models* such as SMPL [124, 125] and GHUM [126] that fit a 3D mesh to a skeleton, this is known as *skinning*. The dense body models, as shown in Fig.4.1 *right*, adapt the mesh to both body pose and the individual's body shape. A dense model allows for direct image segmentation, physics-based [127] and/or detailed human motion modelling [128], but requires more effort to estimate than pose. A dense model may be fit directly to LiDAR pointclouds [129]. Modeling humans in LiDAR alone is hard because LiDAR provides a relatively sparse pointcloud of the human body, as seen in Fig.4.2, even if the human is close to the sensor (the density of points decreases as the human is further away from the sensor).

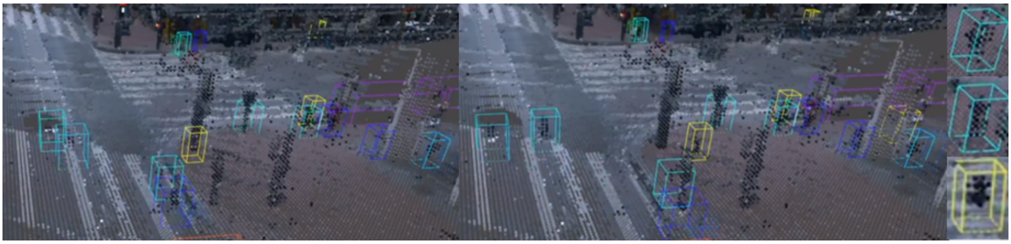


Figure 4.2: Pedestrians in Waymo LiDAR dataset.*To the left:* an early frame showing pedestrians at a crossing. Pedestrians are surrounded by blue, yellow, and turquoise bounding boxes. The point density of pedestrians varies from frame to frame as can be seen by comparing the two left most pedestrians in the left and middle image. Three sample zoom-ins of pedestrians are given to the right, showing that sometimes pedestrian pose is visible in LiDAR scans but not always. The ground points are aggregated across frames as detailed in Paper I supplementary material.

An alternative is to model humans by their joint positions as in Fig.4.1 *middle*. The human skeleton is a natural and compact expression of the *human pose*. The number of tracked joints and the specific joints considered vary from model to model. A difficulty in detecting human pose is detecting the appropriate bone length of the skeleton in varied clothing, poses, and in the presence of occlusions. Therefore it requires in-lab gathered ground truth labels to train robust pose estimation systems in 3D. Often *2D pose* is utilized instead to alleviate the need for 3D pose data. By 2D pose we mean the pixel locations of joint positions in images, shown in Fig.4.1 *left*. Even though data gathering is easier for 2D pose estimation, motion forecasting is harder in 2D because 2D poses are dependent on the camera location and rotation.

In traffic data, it is uncommon to model 3D human pose because ground truth labels are not available. Therefore trajectory forecasting often models humans with 3D or 2D bounding boxes or top view 2D locations of humans in a scene. Articulated human motion models are not yet popular in motion planning in traffic. Trajectory forecasting models concentrate on modeling the effect of other traffic participants and the scene semantics on the trajectory of a pedestrian. A human's motion in traffic is affected by a large number of variables [130, 131]; body dynamics, external factors such as other traffic participants, and the human's mood or intent. There exist also models that attempt to learn human intent from body pose dynamics [52, 53, 132] or images [45, 51].

4.0.1 Human 2D Pose Estimation

Human 2D Pose estimation is a task where given an image of a human the goal is to find the pixel locations $(\mathbf{x}_1 \ldots \mathbf{x}_{N_J})$ of N_J anatomical joints, as seen in Fig.4.1 *right.* We would like to model human motion in 3D poses, but this requires access to large amounts of varied in-lab gathered Motion Capture (MoCap) data. There are still relatively few such datasets because data gathering is time consuming. Therefore 2D human pose estimation is a popular research topic because 2D human joint locations can be quickly estimated by human annotators, allowing for in general larger datasets with large variations in individuals, poses, light and background to be generated.

Graphical models were popular 2D pose estimators in 2010 because they explicitly allow the modeling of the skeletal structure. A *graphical model* is a statistical model that models the interactions (conditional dependence) of stochastic variables in a graph. However, inference in graphical models is complex due to the recurrent structure of the information flow. *Pose Machines* [133] utilize an *inference machine* that approximates the information flow of message passing in a graphical model. The problem is posed as a multi-class classification problem where a classifier should classify each pixel as containing joint i. A *belief map* for joint i gives each pixel the confidence that joint i is located at the pixel. The belief maps for all joints are gathered in $\mathbf{b}_t \in \mathcal{R}^{w \times h \times N_J}$. An inference machine is a sequence of T_M classifiers $\mathbf{g}_t$, $t = [1 \ldots T_M]$, where each stage t in the sequence uses the previous classifer's $\mathbf{g}_{t-1}$ belief map $\mathbf{b}_{t-1}$ of the joint locations and hierarchical image features $\mathbf{x}_z$ to refine the belief maps. The pose machine can be summarized as,

$$\mathbf{b}_1 = \mathbf{g}_1(\mathbf{x}_z) \tag{4.1}$$

$$\mathbf{b}_t = \mathbf{g}_t(\mathbf{x}_z, \mathbf{b}_{t-1}), 2 \leq t \leq T_M. \tag{4.2}$$

At each stage the previous layer's belief map provides context. Each classifier $\mathbf{g}_t$, where $t = [1 \ldots T_M]$ has a separate loss and they are trained independently.

Convolutional Pose Machines [134], a popular 2D human pose estimation network, improved upon Pose Machines by utilizing fully convolutional features instead of the *Histograms of Gradients* (HOG) features and CNNs as classifiers $\mathbf{g}_t$. Convolutional features are naturally hierarchical, allowing natural progression in the receptive field of visual context. Convolutional Pose Machines are also differentiable end-to-end and avoid vanishing gradients by having a loss at the end of each stage. The model's full loss is a sum of the per-stage losses.

A shortcoming of Convolutional Pose Machines is that they can only estimate the pose of one human. To allow for multi-human pose estimation *OpenPose* [135, 136] estimates *Part Affinity Field*s (PAF) which are vector maps that model limbs that connect the estimated joint positions. The PAF's $\mathbf{l}_t \in \mathcal{R}^{w \times h \times 2 \times C_L}$ where C_L is the number or limbs, are calculated again by a T_L staged convolutional inference machine

$$\mathbf{l}_1 = \boldsymbol{\psi}_1(\mathbf{x}_j) \tag{4.3}$$

$$\mathbf{l}_t = \boldsymbol{\psi}_t(\mathbf{x}_j, \mathbf{l}_{t-1}), 2 \leq t \leq T_L, \tag{4.4}$$

where $\boldsymbol{\psi}_t$ are CNNs and the pretrained convolutional layers of VGG-19 [77] are used to calculate image features $\mathbf{x}_j$. The PAF final $\mathbf{l}_{T_L}$ and the image features $\mathbf{x}_j$ are used to calculate per joint belief maps in a second staged convolutional inference machine,

$$\mathbf{b}_1 = \boldsymbol{\rho}_1(\mathbf{x}_j, \mathbf{l}_{T_L}) \tag{4.5}$$

$$\mathbf{b}_t = \boldsymbol{\rho}_t(\mathbf{x}_j, \mathbf{l}_{T_L}, \mathbf{b}_{t-1}), 2 \leq t \leq T_J \tag{4.6}$$

where $\boldsymbol{\rho}_t$ are CNNs and T_J is the number of stages in the joint position estimating module. Because there are multiple instances of each joint it is unclear between which joint instances limbs should occur. The PAF is integrated along the lines that join possible connecting joints, and this value is used to weigh the possible connections. The limb connections are found by weighted bipartite graph matching, which is solved one limb at a time with the *Hungarian algorithm*. Finally, the limbs can be connected to form a skeleton. Both model parts are trained to minimize the L_2 distance to ground truth PAFs and joint position belief maps.

4.0.2 Human 3D Pose Estimation

Reconstructing human pose is hard because humans are non-rigid objects and wear clothes with varying appearances, and humans have varying body shapes and poses. Because humans are not rigid objects we cannot utilize the general 3D reconstruction methods if the human is observed over multiple timesteps. However, it is clear that humans can infer the 3D pose of another human from a 2D image. Therefore it should be possible to train a CNN to do the same. Unfortunately, there is a lack of large datasets with both 2D and 3D annotated human joint positions, therefore information needs to be shared across datasets.

To this end the *Deep Multitask Architecture for Fully Automatic 2D and 3D Human Sensing* (DMHS) [137] performs jointly human body part segmentation, 2D pose estimation, and 3D pose estimation. Like in OpenPose each task is carried out by a convolutional inference machine, see Fig.4.3, but the different inference machines share information at each step. The joint position belief maps are given by,

$$\mathbf{b}_1 = \boldsymbol{\rho}'_1(\mathbf{x}_b) \tag{4.7}$$

$$\mathbf{b}_t = \boldsymbol{\rho}'_t(\mathbf{x}'_b, \mathbf{b}_{t-1}), 2 \leq t \leq T_B, \tag{4.8}$$

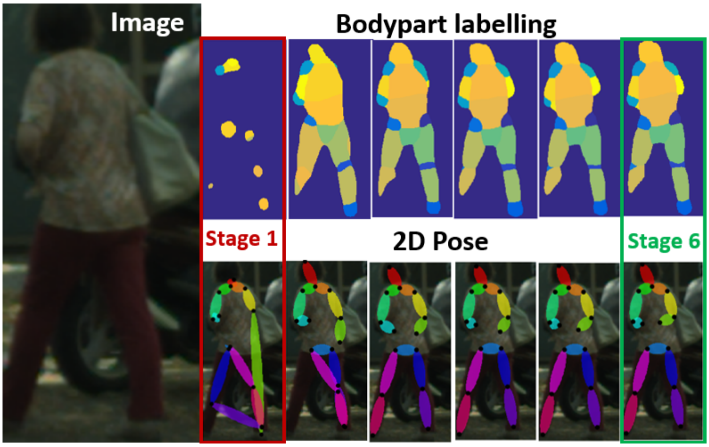


Figure 4.3: An example showing the progression of 2D pose estimation and body part labeling in DMHS stages. The estimate of joint positions and body parts is initially incorrect at stage 1 (in red) and is gradually improved by each stage until the final estimate of stage 6 is output. A similar process is performed for the 3D pose estimate, but it is not depicted here.

where $\mathbf{x}_b$ and $\mathbf{x}'_b$ are convolutional image features found by a 7 and 4-layer convolutional nets respectively, and $\boldsymbol{\rho}'_t$ are CNNs. Like previously L2 loss with ground truth belief maps is used to train the module. In body part segmentation each pixel is classified to one body part label. The body part segmentation's belief maps $\mathbf{z}_t \in \mathcal{R}^{w \times h \times N_B}$ are found by

$$\mathbf{z}_1 = \boldsymbol{\psi}'_1(\mathbf{x}_b, \mathbf{b}_1) \tag{4.9}$$

$$\mathbf{z}_t = \psi'_t(\mathbf{x}'_b, \mathbf{b}_t, \mathbf{z}_{t-1}), 2 \leq t \leq T_B, \tag{4.10}$$

where $\boldsymbol{\psi}'_t$ are CNNs. The CE loss is minimized when training the segmentation module. Finally a feature vector for estimating the 3D poses $\mathbf{r}_t \in \mathcal{R}^{w \times h \times N_R}$ is given by,

$$\mathbf{r}_1 = \boldsymbol{\xi}_1(\mathbf{x}_b, \mathbf{b}_1, \mathbf{z}_1) \tag{4.11}$$

$$\mathbf{r}_t = \boldsymbol{\xi}_t(\mathbf{x}'_b, \mathbf{b}_t, \mathbf{z}_t, \mathbf{r}_{t-1}), 2 \leq t \leq T_B, \tag{4.12}$$

where ϕ'_t are CNNs. Finally 3D positions of the joints $\hat{\mathbf{X}} \in \mathcal{R}^{N_J,3}$ are found by applying a fully connected network $\hat{\mathbf{X}} = \mathbf{f}_R(\mathbf{r}_{T_B})$. The total *mean per joint position error* (MPJPE) for the dataset is minimized to train the 3D reconstruction module,

$$\text{MPJPE}(\hat{\mathbf{X}}, \mathbf{X}^*) = \frac{1}{N_J} \sum_{i=1}^{N_J} \|\hat{\mathbf{X}}_i - \mathbf{X}^*{}_i\|_2, \tag{4.13}$$

where $\hat{\mathbf{X}}_i \in \mathcal{R}^3$ and $\mathbf{X}^*{}_i \in \mathcal{R}^3$ are the i-th joint coordinates in the estimated $\hat{\mathbf{X}}$ and ground truth 3D joint vectors $\mathbf{X}^*$ respectively. The full model's loss function is a sum of the three different module's loss functions. Each module can be trained separately or jointly. This allows the model parts to be trained on different datasets where different annotations are available (2D joint positions, body-part segmentation, and 3D joint positions) without the need for one dataset with all three label types. Therefore the model can be trained on more data, providing robustness during inference. Higher accuracy could be obtained if all three labels are available on the same image data.

4.1 Pedestrian Trajectory Forecasting

The problem of *pedestrian trajectory forecasting* is popularly considered to be the following. Given T top view frames of the scene, with the past T timestep positions in the top view image of all pedestrians in the scene, forecast the pedestrians' positions for the next S steps. For sample trajectories of humans see Fig.1.1. Human trajectories are affected by a number of factors; the goal of the human, the mood of the human, the individual's dynamics (may depend on age and other physical qualities), the behavior and placement of traffic participants, and the geometry or layout of the traffic scenario. Using a top view image of a scene [138–141] does not allow the existing trajectory forecasting models to observe the pedestrian's pose or head direction, making goal location estimation and the forecasting of exact human dynamics harder.

Probably the most influential model in the field is the *Social Forces* model [142] that attempts to model the interactions between pedestrians. Each pedestrian has a goal and a desired velocity towards the goal. Pedestrians in general tend to avoid other pedestrians and this is modelled by a repulsive force that is a function of the distance between pedestrians and the direction of motion. A similar repulsive force is exhibited towards obstacles. Pedestrians can also exhibit time-dependent attractive forces when traveling in a group, or when staying close to an object. All forces are exhibited towards a visible area of the moving direction. The motion of a pedestrian is found by applying these social forces to the desired velocity with a stochastic additive component.

The social forces model has motivated the *SocialLSTM* [143] that learns social interactions in an LSTM, by sharing the hidden states of neighboring pedestrians' LSTMs. SocialLSTM paved the way for a large number of ANNs [144–147] that model spatio-temporally the pedestrian interactions with one another. The Social Forces model also inspired the modeling of pedestrian to pedestrian interactions with GANs [148, 149], coarse to fine networks [150, 151], Graph Neural Networks [152, 153], Transformers [154] and a learnable physics model [155]. There exist also methods that model pedestrians as RL agents [23, 156, 157], perform multi-modal [151, 158, 159] or augmented predictions [160]. In an attempt to learn pedestrian intentions key point and waypoint anchored trajectory forecasting methods [161, 162] have been developed after paper 1 was published.

A number of critical works have been published showing that constant velocity methods can outperform a large number of trajectory forecasting methods [163], and a study [164] showing that the majority of social modeling methods do not understand collisions. In difference to this, in paper I our model is compared to the constant velocity baseline, and our model is trained to avoid collisions with RL. Paper I uses RL when learning beyond existing data while learning efficiently from observed trajectories through supervised learning. The majority of the available work however do not model the interactions between humans and vehicles, and only the later work use scene semantics in the modelling of pedestrian trajectories. Further none of the models ensure that the predicted trajectories are following realistic human dynamics, or model human pose. To our knowledge paper I is the first to model the influence vehicles and scene semantics have on pedestrians, and provide an articulated trajectory.

Models that simultaneously detect and predict the trajectories of pedestrians and vehicles [14, 16, 19] have become popular due to their easy use in AVs. A number of models that specifically model human crossing behavior at intersections based on gaze or 2D pose (i.e. pose in the image plane) [165–169] exist. These methods are often based on a single still image and cannot be used to model pedestrians in other locations in traffic or when the pedestrians are further away from the vehicle. Because both the human and the car influence one another we optimally wish to model their influence on one another. To this end, there exist a number of augmented reality studies [166, 170–172] where humans are asked to react to cars in simulation. The issue with these datasets is that they are relatively small because the experiments are time-consuming.

4.2 Human Pose Forecasting

Human pose forecasting is the problem of predicting the 3D pose for T_{out} timesteps given the 3D poses of the human for the past T_{in} timesteps. Human pose forecasting is concerned with modeling joint movement and often the pose is centered around the center of the hip joints, known as the root joint, to remove the motion of the full body. It has been noted in paper I and in [173, 174] that it is advantageous to forecast the trajectory and pose jointly. The main problem in human pose forecasting is concerned with how to model both the temporal and spatial relations present. The continuous movement of a joint creates temporal patterns, and the dependence of joints upon one another creates spatial patterns. Due to the physical properties of the human body, certain motions may be invalid, and some more probable.

The majority of early ANN based human pose forecasting methods [175, 176] use RNNs [177–179] or *Variational Autoencoders* (VAE)s [180, 181] that should learn the motion of the joints. Recently Graph Convolutional Networks [182–186] have gained popularity as a method to model relations between joints. The *Space-Time-Separable Graph Convolutional Network* *for Pose Forecasting* (STSGCN) [183] is a relatively low dimensional model that improved significantly over previous work. The model uses *Graph Convolutional Networks* (GCN) to encode human pose. GCN's approximate convolutions in graphs [187]. A graph here is a set of vertices described by feature vectors $\mathbf{v}_g$ that are connected by weighted edges. An *adjacency matrix* $\mathbf{G} \in \mathcal{R}^{N_g,N_g}$ described the graph connections of a matrix with N_g vetrices. The adjacency matrix component $G_{i,j}$ contains the weight of the edge connecting vertex i with vertex j. The diagonal elements $G_{i,i}$ of a graph adjacency matrix denote the number of vertices the vertex i is connected with. A *normalized graph Laplacian* is $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{G}\mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{D}$ contains the diagonal elements of $\mathbf{G}$. The eigenvalues Δ_g of $\mathbf{L}$ measure the frequency of the connections corresponding to the eigenvectors $\mathbf{U}_g$ of $\mathbf{G}$. A graph convolution $\star$ of a graph $\mathbf{v}_g$ with a filter $\mathbf{f}_g$ is applied as

$$\mathbf{f}_g \star \mathbf{v}_g = \mathbf{U}_g \mathbf{f}_g(\Delta_g)\mathbf{U}_g^T \mathbf{v}_g, \tag{4.14}$$

where $\mathbf{f}_g$ is the learnable filter that is applied to the diagonal elements of Δ_g. In STSGCNs, the convolution is approximated and separated into a matrix multiplication with two learnable vectors $\mathbf{W}_\Theta$ and $\mathbf{A}_\Theta$. Where $\mathbf{W}_\Theta \in \mathcal{R}^{N_c \times N_C}$ weighs the vertex features, where N_c is the number of channels in the input vertex features and N_C of the output features, and $\mathbf{A}_\Theta \in \mathcal{R}^{N_g N_t \times N_g N_t}$ is the learned adjacency matrix where N_t is the observable time horizon. A GCN layer consisting of a learnable graph convolution with a non-linearity f_σ is given as,

$$\mathbf{f}_g \star \mathbf{v}_g = f_\sigma(\mathbf{A}_\Theta \mathbf{V}_g \mathbf{W}_\Theta), \tag{4.15}$$

where $\mathbf{V}_g \in \mathcal{R}^{N_g N_t \times N_c}$ is the input to the GCN. To reduce learnt variables the STSGCNs explicitly separate the temporal and spatial components of the learnt adjacency matrix $\mathbf{A}_\Theta$. The model applies the spatial adjacency matrix spatially over one channel and the temporal adjacency matrix across multiple channels per spatial location. STSGCN uses the *Parametric Rectified Linear Unit* (PreLu) activation function that is a relaxation of the ReLU function allowing parametric a non-zero gradients for $x < 0$ $\text{PreLu}(x) = ax$, otherwise $\text{PreLu}(x) = x$. A four-layered GCN encodes the K past poses, which are decoded by a four-layered *Temporal Convolutional Network* (TCN). A TCN [188] consists of a 1×1 convolution followed by a causal convolution, performing convolutions across the time domain rather than spatially, followed by a nonlinearity. TCNs model temporal patterns and have a longer memory than RNNs [188]. The STSGCN is trained to minimize the average per joint position or angle errors.

From the animation community, human motion modelling has been done mostly with *motion retargeting*. That is small samples of MoCap data are replayed from a database with adjustments to new limb lengths. Interpolation is used to smooth motion between different motion samples from the database. The *Phase-Functioned Neural Networks* (PFNN) [189] is a popular model that utilizes ANNs instead of a database. PFNN models walking as a cyclic behavior according to the placement of the feet. A cycle begins when the right foot touches the ground and has phase π when the left foot touches the ground. All other poses are assigned a phase through interpolation. The phase functional neural network consists of a phase function $\mathbf{f}_\alpha$ that assigns ANN weights $\mathbf{w}_{NN} = \mathbf{f}_\alpha(\alpha, \boldsymbol{\alpha}, \mathbf{W}_\alpha)$ to the motion forecasting network based on the phase α of the motion. The phase function $\mathbf{f}_\alpha$ is a *Catmull-Rom spline*, a differentiable cyclic interpolation between the control points $\mathbf{W}_\alpha = (\mathbf{w}_{\alpha_1}, \mathbf{w}_{\alpha_2}, \mathbf{w}_{\alpha_3}, \mathbf{w}_{\alpha_4})$. The phase function $\mathbf{f}_\alpha = \sum_{i=1}^{4} f_w(\alpha, \boldsymbol{\alpha})\mathbf{w}_{\alpha_i}$ is a weighted sum of the control points, where the weights are given by a non-linear function $f_w(\alpha, \boldsymbol{\alpha})$ of the phase α and the control point phases $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$. The NN that forecasts joint positions $\mathbf{f}_{\mathbf{NN}}$ is a three-layered fully connected network with 512-dimensional hidden layers with *Exponential Rectified Linear Unit*(ELU) activations $\text{ELU}(x) = \max(x, 0) + \exp(min(x, 0)) - 1$. The network $\mathbf{f}_{NN}$ obtains its weights from the interpolation $\mathbf{f}_\alpha$, so the full model is trained by optimizing the objective,

$$J_{\text{PFNN}} = \min_{\mathbf{W}_\alpha} ||\mathbf{Y}_{\text{PFNN}} - \mathbf{f}_{NN}(\mathbf{f}_\alpha(\alpha, \boldsymbol{\alpha}, \mathbf{W}_\alpha), \mathbf{X}_{\text{PFNN}})||, \tag{4.16}$$

where $\mathbf{X}_{\text{PFNN}}$ is a vector consisting of the inputs, and $\mathbf{Y}_{\text{PFNN}}$ of the outputs of $\mathbf{f}_{NN}$. The vector $\mathbf{X}_{\text{PFNN}}$ contains the previous timestep's hidden layers of the network $\mathbf{f}_{NN}$, past joint velocities and positions, gait style (walking, running, jumping etc), and future planned trajectory for the human. The vector $\mathbf{Y}_{\text{PFNN}}$ contains the next timestep joint positions, velocities and rotations, the future translation of the root joint, the next timestep's phase and future predicted trajectory. The control phases $\boldsymbol{\alpha}$ are chosen at fixed points. More complicated human motions have been replicated for animation with learning-based models trained in a *physics engine* to mimic the motion of humans [108, 190–192].

There is still a gap between existing human trajectory models and human pose forecasting models. This is likely due to the lack of MoCap data of humans interacting freely in a general environment. The general lack of highly accurate but varied MoCap data has led to increased research efforts into generative methods that attempt to learn a latent space that is representative of the possible human motions [193–198]. It is however clear that a combined model capable of human trajectory forecasting and human pose forecasting could improve results on both tasks.

Chapter 5

Autonomous Vehicles

Autonomous vehicles reason about what steering signals to give based on sensor observations. By steering signals, we mean steering angle, throttle, gas and break signals. Vehicles can have a number of sensors such as cameras, Light Detection And Ranging (LiDAR), thermal cameras, radar, ultrasonic, event cameras, etc. A LiDAR consists of a consist of light-source and a light detector. Distance to objects is measured by the time of flight of the reflected light. LiDAR sensors produce a pointcloud that is denser the closer an object is to the sensor. We will concentrate on cameras and LiDAR as sensor inputs to allow the use of the vast majority of visual recognition and modelling from the computer vision community. A combination of sensors is expected to perform best, especially in varying visual conditions.

To make driving decisions based on sensor input there are a number of subtasks to be solved. The following two subtasks are usually assumed to be solved by a unit separate from what is considered to be the AV model:

- *Route Planning* - What roads to take to travel from a start location to a destination. There exist various commercial products such as Google Maps, etc. that solve this.
- *Localization* - Exact localization of the vehicle. This is needed to provide the vehicle close by sub-goals known as *waypoints* and a local layout of the road to ease perception.

The following subtasks are usually considered to be a part of the AV:

- *Perception* - Detecting traffic signs, other traffic participants and their motion, the layout of the neighbourhood of the AV i.e. the *scene*, and any other relevant objects.

- *Behavior Planning* - Planning how the other traffic participants may behave and how the AV should move to avoid collisions with them.
- *Motion Planning* - Planning how to move to reach the waypoint in a feasible (from the standpoint of the car dynamics) and comfortable manner.
- *Vehicle Control* - How to perform the planned motion in actuator control. Requires control and understanding of the vehicle physics in the current road conditions.

In modular AV systems, these subtasks are solved by one or more separate systems, and in end-to-end models, one single model is responsible for solving all of the above AV subtasks. End-to-end models are typically Deep RL or imitation learning models. End-to-end methods [199–204] have obtained impressive results on AV driving benchmarks [205] and allow the information to flow among the different sub-components (including uncertainty), but provide little insight to what the decisions were based on. On the other hand in hierarchical models such as [206] decisions are easier to interpret but separating the different problems into modules may inhibit the natural information flow, leading to miscommunication among the modules, or even negligence of important objects or situations. There exist a number of methods [207] that utilize one single model from *perception* to *motion planning* but use methods from automatic control such as *Proportional-Integral Derivative* (PID) control [208] for *vehicle control*. Independently of the design when evaluating the AV it is crucial to ensure that any errors made in the full pipeline should be detected during testing. Therefore papers II and III propose methods to test the full pipeline of an AV.

5.1 Perception and Behavior Planning

This work in particular concentrates on the perception and behavior-planning tasks in AVs. To this end, we will shortly give an overview of the models used to capture the objects in the AV's proximity. One problem that is central in AV modelling from images and LiDAR is the amount of data that is generated. To plan the AV's trajectory the distances to objects in the scene must be recovered, however, this takes a large amount of space and without compression, a scene observation of 2s can take around 10-30 GB to store (this is the case for the scenes in paper V). In an hour of driving this would mean around 20 TB of data. Therefore it is crucial to filter out only salient information so that the observed data can be saved or communicated to a server to improve future models. Therefore compact representations of the traffic scene have been explored in the literature [209–213]. Nonetheless 3D pointclouds and dense 3D reconstructions as well as LiDAR data can be used for realistic simulations.

Birds-Eye View (BEV) images (i.e. top view image of a scene) of the local neighbourhood have been considered [210], because they allow for clear distance and motion measurements and can be directly utilized in a number of neural computer vision models. Unfortunately, visual cues of pedestrians and other cars are lost. In BEV the AV cannot, for example, observe the pedestrian's pose or head direction which are both important cues for human motion prediction in traffic [165–169, 214]. The BEV allows for motion planning models [215, 216] to capture the semantic relations of the scene and the traffic agents, but is susceptible to miscalculations in depth.

Another common approach is to use rasterized High Definition maps [209, 217–219], where the scene is further simplified to road lanes and 3D bounding boxes for co-traffic participants. Though this method is extremely compact and allows for the direct matching of road lanes as captured in images to the road lanes as found in a map, it is possible that this method is too simple omitting the scene semantics and therefore omitting important data for the motion planning task.

Finally there exist models that simultaneously perform object detection, tracking and planning [14, 16, 19] in top view images. Trajectory forecasting and motion planning in the presence of occlusions has recently gained attention [220–224]. Similarly models that predict multi-modal motion and safe planning are of interest to be able to avoid threat filled actions [225–230]. Note that in the majority of the planning models pedestrians and vehicles are modelled homogeneously [34, 36–44], assuming that a similar model will be able foresee their behavior. We argue that there is no particular reason to believe that this is the case when humans and vehicles have different behaviors in traffic. This is also strongly motivated by the fact that models developed to foresee vehicles' motion exclude the visual and motion cues present in human pose and gaze.

5.2 Simulating Autonomous Vehicles

Simulation is often used to train and test AVs because it is cheaper and safer than training or testing in the real world. A large number of AV methods have been developed in simulation environments such as CARLA [205, 214]. These simulation environments are built on physics engines and utilize complex models of the vehicle's dynamics allowing the testing of steering. However, it is hard to generate realistic sensor output. The realism of the generated sensor output is easily evaluated by humans in images, but less so in LiDAR scans. Realistic LiDAR simulation is studied in [231], this requires correct modelling of datapoint and noise distributions. Even if sensor outputs are modelled in great detail an AV model that performs well on simulated data may fail on real data due to the discrepancy between the simulated training data and real test data. This is known as the simulated-to-real gap (Sim2Real). The AV research community has placed a lot of effort into closing the Sim2Real gap [232–235], but this requires additional treatment of existing methods.

The Sim2Real gap is particularly of interest for visual data when modelling pedestrians because in the real world humans can vary in size, height, age, body composition and dynamics. This is seldom represented in the simulated data. Secondly, the majority of pedestrian detection models that utilize articulated models, thus allowing for precise human motion modelling, rely on visual data. Finally, in traffic in particular there is also a Sim2Real behavioral discrepancy. It is common to encounter only the best and the worst behaviors of vehicles and pedestrians in simulations, but in reality there are a large number of variations in how traffic participants can behave depending on their goal, mood, traffic density, and the presence of abnormal traffic behavior and temporal constraints.

This has made the augmentation of real data an interesting research field to allow for more varied data with a smaller Sim2Real gap. To this end, there has been an explosion of visual models that allow one to generate new views in a traffic scene [236–239]. The rise of Neural Radiance Fields (NeRF) [240] has given hope to new view synthesis. The increase in new view synthesis methods gives hope that AVs could be tested in augmentations of real data. The drawbacks to this are still that a large amount of data needs to be recorded, and it is currently unclear how to adapt the models to show the possible visual and behavioral variability in the scene. Similarly, LiDAR data generation and augmentation have been studied in [231]. Ultimately the goal is to be able to augment data by being able to generate new views (so the AV can move freely in the space), to generate new behaviors for traffic participants possibly changing the number of traffic participants and by visually altering the scene.

5.3 Testing Autonomous Vehicles

Accidents and critical scenarios occur seldom in traffic. Therefore critical scenarios need to be simulated to test and train AVs. A critical scenario can be defined in a number of ways, from the occurrence of an accident to an uncomfortable driving style. The behavior of an AV should be tested under varied conditions to ensure traffic safety, namely under

- adversarial attacks - that is a small perturbation is added to sensor measurements changing the perceptive model's output and affecting the planned route of the AV,
- traffic density variations - variations in the number of vehicles and pedestrians present,
- behavioral variations - variations in the traffic participants behavior,
- presence of abnormally behaving traffic participants - other traffic participants may not follow traffic rules or behave out of the ordinary,
- car dynamics variations - variations in road conditions may change the car dynamics,

- visual variations- variations in the appearance of different objects, but also in the weather conditions,
- scene layout variations - different traffic scenarios.

To this end, testing can be performed in a scenario-based manner [241–245] or modularly [246–249]. Scenario-based testing allows for the testing of end-to-end AV models and for the testing of interactions of the modules of a modular AV model. Scenario-based testing allows for qualitative analysis of the realism of generated test scenarios. This is advantageous because a large number of high-risk scenarios are unrealistic, and AVs should be tested on critical and plausible scenarios. An easy example of a very high-risk but unrealistic scenario is one where all traffic participants (including pedestrians and bikers) actively seek collisions with the AV. The probability of a collision is high in this case, but the chances of encountering such a scenario in real traffic are slim, therefore this is not a particularly interesting test scenario. Instead realistic high-risk scenarios where for example a collision may occur due to poor visibility in regular traffic are of higher interest.

Test case generation can be separated into white-box treatments and black-box treatments of the AV. In white box models the decision-making model of the AV and the AV's dynamics are assumed to be known. White box modelling must be performed (and possibly developed) uniquely for each AV model architecture, but is in general more efficient than black box modelling where the AV's reasoning is considered to be unobservable- i.e. a black box. Black box modelling in AV testing however provides a general method to test and compare any AV models.

When utilizing dynamic programming or reachable sets to back-track collisions the world dynamics must be known and reversible in simulation. This is in general hard for end-to-end AV models that make decisions based on signal input, as backtracking requires the ability to reverse sensor simulations, and omits the possibility of utilizing real-world traffic data with unknown dynamics for external traffic participants. Further, back-tracking-based methods require that the set of collisions in space and time are known, which in general is not the case due to the sparsity of collisions. Finally, additional care must be taken to ensure that the initial state of scenarios that are found by back-tracking are realistic in traffic and that the scenarios are semantically coherent. Therefore the following discussed methods utilize forward simulations only to generate collisions.

There exist a number of methods to generate test cases for AVs. The problem can be solved through data interpolation [250], extrapolation [251] or by perturbation [252–254]. However, these methods often lead to scenarios that are similar to the original data, not providing insight into the AV's performance on unseen scenarios. Optimally the test generator should generate realistic and varied test cases that represent the possible variability in all realistic collision scenarios. Because real datasets are relatively small it is unlikely that the existing public datasets are able to cover all of the possible variability in real traffic scenario geometries, traffic behaviors, traffic density and visual variations. This limits also the variability of test cases that can be generated from the existing public datasets, therefore extrapolative methods must be used to generate more varied test cases. Some realism is expected to be lost with extrapolation.

Collision generation can also be seen as a black box or white box optimization problem [255–257], where scene-describing parameters are optimized to induce collisions. To utilize the MDP structure of the problem RL can be used instead to choose adversarial scene parameters [258, 259] or be used to model adversarial traffic participants [260]. Collision generation can be treated as an adversarial attack on the car [261, 262], but the generated scenarios are confined by the variability of the existing dataset. When only planning components are tested driveable-area-based test case generation methods can be used [263–265].

In this work, we draw attention to the pedestrian models utilized in test case generation. Often human behavior is modelled extremely simplistically as a constant velocity motion or as adversarial agents (i.e. collision seeking) with little respect for human dynamics. This makes collision generation easy, but the generated collisions are not likely because the human motion is unrealistic. Further, the scene's realism falls with an increasing number of unrealistic pedestrians. A realistic pedestrian's motion is semantically grounded in a traffic scene (i.e. the pedestrian follows traffic rules to some degree depending on the scene geometry and traffic density), physically plausible for the human body to exhibit and varying depending on the pedestrian's intention and the traffic around the pedestrian.

Finally, it is worth noting that there is a large number of possible additional open quality issues with AVs such as how to share data privately when communicating with other devices, how to perform calculations energy efficiently, what to communicate among AVs, how to scale up the data gathering and who is liable in the case of an accident.

Chapter 6

3D Reconstruction

An image is a two dimensional projection of the three dimensional world. Therefore by taking a picture we lose information along one dimension; the distance from the camera center to the object. By performing 3D reconstruction we aim to recover this distance such that we may find the 3D positions of all points seen in a camera. Here some of the basics of 3D reconstruction are shortly presented.

6.1 The Pinhole Camera Model

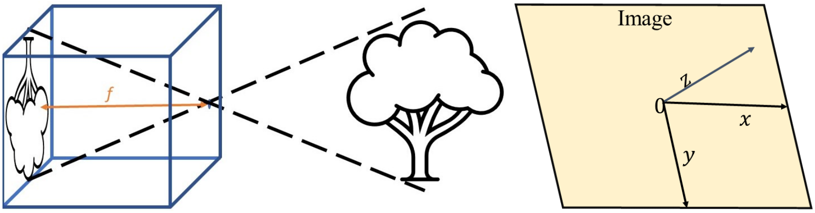


Figure 6.1: *To the left:* As light rays pass a small aperture they create an image of the world on the back of the camera. The distance from the aperture to the image is known as the focal length denoted by f. *To the right:* The camera's coordinate system in an image, where Z points outwards.

The pinhole camera models a camera as an enclosed box with a single hole. As light rays pass the hole a reversed image of the world is drawn on the back of the camera wall as seen in Fig.6.1. It is assumed that the hole in the camera is small enough such that only one ray of light from each point passes through it, allowing us to see a sharp image. The location of the aperture is known as the *focal point*, and the distance from it to the back of the camera is the *focal length*, denoted f. In practice, the image may be captured by a digital ray of sensors, placed at the back wall of the camera, that digitalizes the image. As the image is captured by an array of light sensors on the back of the camera the image may be deformed by the shape and placement of the sensors. This is commonly described by the camera's internal parameters gathered in the intrinsic matrix $\mathbf{K} \in \mathcal{R}^{3\times 3}$, that describes how the camera projects the 3D points onto the image plane. Images are commonly digitalized into pixels. Often each pixel is represented by three colors red green and blue giving rise to the RGB representation that consists of three images - one per color. The different colored images are referred to as the different channels of the image and are commonly indexed by a third channel depth index.

The camera is translated by $\mathbf{c}_x \in \mathcal{R}^3$ and rotated by $\mathbf{R} \in \mathcal{R}^3$ with respect to some global coordinate system, where $\mathbf{c}_x$ and $\mathbf{R}$ are in general unknown. The camera has its own coordinate system that defines the positions of the different pixels at $x \in \mathcal{R}, y \in \mathcal{R}$ coordinates (pointing to the right and downwards respectively in the image plane) and the coordinate pointing outwards from the camera is the depth or $z \in \mathcal{R}$ axis, as seen in Fig.6.1. To ease notation we will treat the image that is formed in front of the camera and is therefore not upside down. If the matrix $\mathbf{K}$ is known we can form a calibrated image that is formed at one unit length in front of the camera, i.e. $z = 1$, is centered around the camera's focal point and has square pixels. The image $\mathbf{x}$ of a 3D point $\mathbf{X}$ in the global coordinate system is formed by

$$z\mathbf{x} = \mathbf{K}[\mathbf{R}|\mathbf{c}_x]\mathbf{X}, \tag{6.1}$$

where $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{c}_x]$ is the camera matrix, and note that $\mathbf{X} \in R^4$ with number one as the last coordinate to allow translation to be written as a matrix multiplication.

6.2 Overview of Structure from Motion

In a typical *Structure from Motion*(SfM) problem an object or a scene is observed from a number of cameras. The aim is to find the 3D structure of the objects depicted in the images, and the positions and rotations of the cameras $\mathbf{P}$. The classical 3D reconstruction processes are often divided into a sparse and a dense reconstruction. A sparse 3D reconstruction is found from selected points in images to reduce computational costs and increase robustness to noise. As a second step, a dense reconstruction may be performed after the camera matrices $\mathbf{P}$ have been found.

Sparse reconstruction is usually performed on points that are expected to look similar across views. Typically these points are corner points or similar that are found by their high gradients in different directions and colors. The selected points are described by feature vectors such as the well-known *Scale Invariant Feature Transform* (SIFT) [266], which have recently been replaced by learned features [267]. The feature vectors are then compared by a similarity measure and matched across images. Because a large number of points can have similar appearances it is common to encounter a large number of outliers and noise.

To reduce the effect of outliers *Random Sample Consensus* (RANSAC) [268] is commonly used. This is a process in which a small (possibly minimal) number of matches are selected at random, and camera matrices are found according to the matches. This is done a number of times. The best fit among the random samples is found by maximizing the number of inliers (points with a reprojection residual below a certain threshold). The reprojection residual $||\mathbf{x}'_r - \mathbf{x}_r||$, see Fig.6.2, is the distance from the projection $\mathbf{x}'_r$ of a reconstructed point $\mathbf{X}'$ to the original 2D point $\mathbf{x}_r$ in the image space. The sparse reconstruction's solution is often finetuned by a global optimization over the 3D points and camera matrices, the optimization is known as *bundle adjustment* (see §6.4). Finally, *dense 3D reconstruction* is performed by finding the depth of each pixel in each image. This can be done by for example disparity estimation, see §6.3.

6.3 Basics of Binocular Triangulation

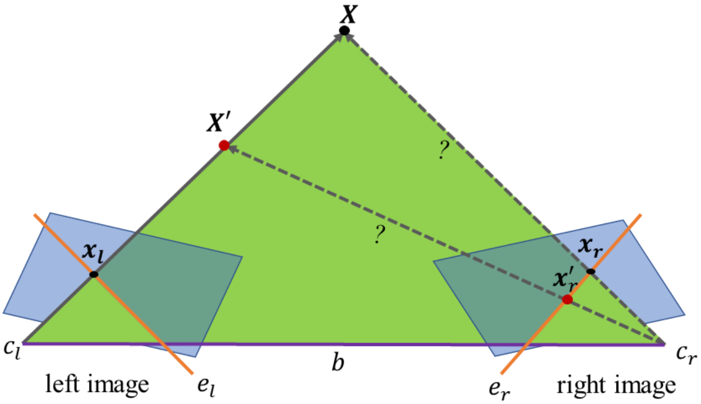


Figure 6.2: When observing a 3D point $\mathbf{X}$ with a projection $\mathbf{x}_l$ in the left image then we know that the corresponding point in the right image must lie of the epipolar line e_r in the right image marked as the orange line. The object could be found along any of the dashed grey lines, and $\mathbf{x}_r$ can be found by matching $\mathbf{x}_l$ with all pixels along e_r. The camera centers $\mathbf{c}_l$ and $\mathbf{c}_r$ are separated by the baseline b in purple. The epipolar plane is marked in green. If the point $\mathbf{x}_l$ is incorrectly matched with $\mathbf{x}'_r$ then this results in the incorrectly reconstructed point $\mathbf{X}'$ instead of $\mathbf{X}$.

A binocular rig consists of two identical cameras that are fixed such that they view the world horizontally in parallel (just like our eyes). Image rectification can be used to ensure that the images are exactly parallel. Any visible object will be shifted in pixel values from one image to the other along epipolar lines as seen in Fig.6.2. Epipolar lines lie in the same plane as the cameras, and in our work most often the epipolar lines are horizontal. Therefore if an object has x-coordinate $x_1 \in \mathcal{R}$ in the left image and x-coordinate $x_2 \in \mathcal{R}$ in the right then the displacement of the object between the images $d = x_1 - x_2$ is known as the *disparity*.

In Fig.6.3 we can see the triangle that is spanned by the cameras and the viewed object, for notational simplicity here we form the image in front of the camera center at a depth f. The

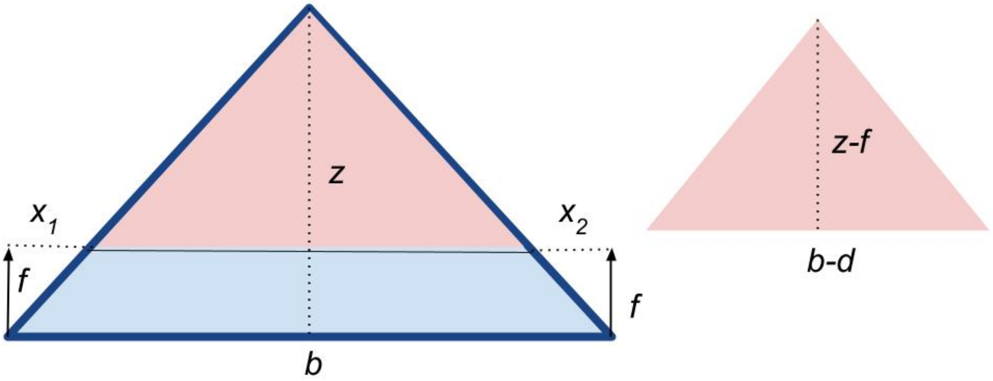


Figure 6.3: The same object appears on the right of the camera center in the left camera and to the left of the camera center in the right camera. This spans a triangle drawn in dark blue that has a base $b \in \mathcal{R}$ and height $z \in \mathcal{R}$. The dark blue triangle is similar with the pink triangle within it with base $b - d$ and height $z - f$.

unknown depth from the object to the cameras is denoted z and it is the height of the dark blue triangle. The distance between the cameras $b \in \mathcal{R}$, also known as the *baseline*, is the base of the triangle in dark blue. Because the image of the object is assumed to lie to the right of the focal point of the left camera x_1 is positive, and x_2 lies to the left of the right camera's focal point so x_1 is negative. Therefore the pink triangle's base can be expressed as $b - d$, and its height is $z - f$. Because the pink and the blue triangle surrounding it have the same ratio of height to base,

$$\frac{z}{b} = \frac{z - f}{b - d} \qquad\qquad \implies z = \frac{bf}{d}.$$

Finally, we note that the baseline and focal length are physical values that are known and can be measured from the experimental setup. Therefore the depth of an object z is inversely proportional to the disparity d. Dense disparity is however quite often noisy. It is found by comparing a small region of size $h \times w$ of an image to other regions of the image of the same size, by methods such as the classical *Sum of Squared Differences* (SSD) [269]. If the image contains multiple regions that look similar then the disparity may be confused about which are the regions that match. Note that there exist modern monocular depth estimation networks [270] that learn to estimate the depth of all objects in a single image. These methods were still in the development stage at the time of developing the paper v, so paper v utilizes disparity. It should be noted that the monocular depth estimator networks need to be trained on similar data as they are utilized on, and may be outperformed by classical methods in previously unseen scenes.

6.4 Bundle Adjustment

Bundle adjustment optimizes the camera matrices and 3D points such that re-projection error ($\|\mathbf{x}'_r - \mathbf{x}_r\|$ in Fig.6.2 for all points and images) of the recovered 3D object is minimal. This is done using non-linear least squares. For simplicity, we assume that the camera's internal matrix $\mathbf{K}$ is known. Let M_i points $\mathbf{X}_i \in \mathcal{R}^4$ in the world be projected in M_j cameras with camera matrices $\mathbf{P}_1 \ldots \mathbf{P}_{M_j}$. The point $\mathbf{x}_{i,j} \in \mathcal{R}^3$ is a projection of the point $\mathbf{X}_i$ in the j-th camera. The projection is described as $\mathbf{x}_{i,j} = \frac{1}{z^i}\mathbf{P}_j\mathbf{X}_i$, where z^i is the depth of the i-th point in the global coordinate system. Then bundle adjustment is the joint optimization of the camera's external parameters in $\mathbf{P}_j$ and the 3D points $\mathbf{X}_i$ to minimize the reconstruction error

$$\min_{\mathbf{X}_1 \ldots \mathbf{X}_{M_i}, \mathbf{P}_1 \ldots \mathbf{P}_{M_j}} \sum_i \sum_j \left\| \mathbf{x}_{i,j} - \frac{1}{z^i}\mathbf{P}_j\mathbf{X}_i \right\|^2 . \tag{6.2}$$

The problem is bipartite; if $\mathbf{P}_j$ are known then the optimization of $\mathbf{X}_i$ becomes independent, and vice versa. This can be utilized in the optimization of 6.2. When bipartiteness is utilized then the points and cameras can be optimized iteratively by alternating between keeping $\mathbf{X}_i$ fixed while optimizing $\mathbf{P}_j$ and keeping $\mathbf{P}_j$ fixed while optimizing over $\mathbf{X}_i$. The problem can be solved by for example linearized Gauss-Newton or Levenberg–Marquardt methods [269]. A well-known shortcoming of bundle adjustment methods is that they are often dependent on the initialization of the camera matrices and can fail to converge if initialized too far from the global optimum. Uncertainties from 2D point matching can be included in the bundle adjustment objective. It should be noted that since the development of the results of paper v modern deep learning-based methods for feature matching [271], and SfM [272, 273] and even dense *simultaneous localization and mapping* (SLAM) systems [274–276] have been developed.

In paper v a moving camera is treated as multiple differently positioned cameras after removing dynamic objects from the images. One of the key difficulties in reconstructions from a moving camera is motion blur, that is blur in an image as a result of motion or too low shutter speed. It is difficult to recognize the same points from multiple blurred images, therefore bundle adjustment can be calculated only for a few of the 3D points and cameras.

6.5 Procrustes Analysis

The set of 3D points $\{\tilde{\mathbf{X}}_1 \ldots \tilde{\mathbf{X}}_N\}$ that are estimated from images by SfM or SLAM are known as a 3D pointcloud. If we observe the same 3D object or scene again at a different point in time we may wish to recognize the same 3D structure. Therefore we need a method to align two 3D pointclouds. To this end, Procrustes Analysis [277] may be used to find the scaling, rotational and translational difference between two 3D pointclouds with known matching points. In paper v Procrustes Analysis is used to compare human poses.

Given two sets of N points gathered into matrices $\mathbf{A}, \mathbf{B} \in \mathcal{R}^{(N,3)}$, then full Procrustes analysis finds the optimal rotation, translation and scaling such that the ordered pointclouds match,

$$J_P = \min_{\beta,\gamma,\mathbf{\Gamma}} \|\mathbf{B} - \beta\mathbf{A}\mathbf{\Gamma} - 1_N\gamma^T\| \tag{6.3}$$

by optimizing the scaling $\beta > 0$ and the rotation matrix $\mathbf{\Gamma} \in SO(3)$, and translation $\gamma \in \mathcal{R}^3$, where 1_N is a vector of ones with length N. The points in $\mathbf{A}, \mathbf{B}$ are assumed to be centered around their centroids. The solution [277] to (6.3) is

$$\gamma^* = 0 \tag{6.4}$$

$$\mathbf{\Gamma}^* = \mathbf{U}\mathbf{V}^T \qquad \text{where } \frac{\mathbf{B}^T\mathbf{A}}{\|\mathbf{B}\|\|\mathbf{A}\|} = \mathbf{V}\hat{\mathbf{D}}\mathbf{U}^T, \mathbf{U}, \mathbf{V} \in SO(3). \tag{6.5}$$

$$\beta^* = \frac{\text{trace}(\mathbf{B}^T\mathbf{A}\mathbf{\Gamma}^*)}{\text{trace}(\mathbf{A}^T\mathbf{A})}, \tag{6.6}$$

where $\hat{\mathbf{D}}$ is a diagonal matrix with positive real values on the diagonal. If $\hat{\mathbf{D}}$ contains negative elements on the diagonal then this means that a reflection should be performed. Procrustes analysis is used in Paper v to correct impossible 3D reconstructed poses by finding the closest match in a dataset of plausible poses to a reconstructed pose.

Orthogonal Procrustes Analysis [278] can be used, where $\mathbf{\Gamma}$ is allowed to be in any orthogonal matrix, i.e. $\det(\mathbf{\Gamma}) = \pm 1$, thus allowing for reflections. The problem is simplified from (6.3) by dropping the scaling β and translation γ^*. The solution [278] is somewhat less elegant,

$$\mathbf{B}^T\mathbf{A}\mathbf{A}^T\mathbf{B} = \mathbf{V}_o\mathbf{D}_o\mathbf{V}_o^T \tag{6.7}$$

$$\mathbf{A}^T\mathbf{B}\mathbf{B}^T\mathbf{A} = \mathbf{U}_o\mathbf{D}_o\mathbf{U}_o^T \tag{6.8}$$

$$\mathbf{\Gamma}_o^* = \mathbf{U}_o\mathbf{V}_o^T \tag{6.9}$$

and is given by the diagonalizations (6.7) and (6.8), where $\mathbf{D}_o$ is a real diagonal matrix with non-zero diagonal entries and $\mathbf{V}_o$, $\mathbf{U}_o$ are orthonormal matrices.

Chapter 7

Concluding Marks

As humans, we have a large understanding of traffic rules as well as an inherent understanding of human motion. Training computers with the goal of reaching the same levels of understanding as humans is a challenging task for computers. In most countries, humans are allowed to build their internal understanding of the world and its dynamics for 15-18 years before being allowed behind the wheel of a vehicle. However, it is completely unacceptable for a computer with poorer computational power (in particular in vision) than the human brain to learn such behavior for 18 years (in fact longer due to the lower computational power of computers) by solving other related tasks needed to understand the world and its dynamics. Learning systems should in fact be compared to a new-born baby who is being taught to drive a car. This comparison makes the achievements made in autonomous driving much more impressive. When in particular looking at how computers understand the behavior of humans one should note that humans have a large advantage compared to computers; we can transfer our knowledge of how our own bodies move to form expectations of the motions of pedestrians. Humans start to learn about their dynamics already before being born [279] and sensorimotor systems are developed further through sporadic movement during sleep [280]. Further, it can be expected that our brains are wired to be biased to understand human dynamics such that one can foresee his or her own body's motion in complex tasks with extremely little computation time [281]. Therefore computers are at a disadvantage compared to humans in pedestrian forecasting and effort is needed to catch up in pedestrian motion modelling with humans. Human motion prediction becomes more confined in physiologically detailed models such as the sensory-driven muscle-reflex model of human legs [282] or the neuromechanical models [283, 284] as some motion can be modelled by Central Pattern Generators (CPG). This thesis argues that by modelling interactive pedestrians with (at least) articulated motion in traffic we should be able to improve the AVs' safety in interactions with humans.

In general pedestrian motion can be expressed to be stochastic (when compared to cars for example) because there are a large number of unobservable factors that affect a pedestrian's motion. However, somehow humans are able to effectively predict the possible motions of pedestrians. In paper I we have made an attempt to model a number of, to our knowledge previously unmodelled, factors that clearly affect pedestrian motion in traffic. We overcome shortcomings in available data by combining prior knowledge and various datasets.

Secondly, we note that humans and vehicles are mathematically speaking playing a game when interacting in traffic. In this game, both pedestrians and vehicles are on the same team even though they do not share exactly the same objectives. The actions of pedestrians do affect the actions of AVs and vice versa. Therefore special care must be taken when modelling pedestrian motions in traffic. In particular, in AV testing pedestrians should be modelled as cooperatively interactive to identify the AV's shortcoming in interactions with realistic pedestrians. We show that utilizing cooperative pedestrians in AV testing is possible in papers II and III.

AVs should be tested in collision-prone scenarios before deployment. This requires the efficient generation of collision-prone scenarios in varied settings. But the scenarios where collisions are most likely to occur are often unrealistic, therefore realism and collision-proneness must be balanced. In difference to previous work papers II and III utilize realistic pedestrian behaviors making the search for collision-prone scenarios harder. To ensure that collisions occur the AV and pedestrians can be constrained by the environment by, for example, occluded views of the world. Paper III proposes an easily extendable framework that allows for varied constraining of the AV. This is important because as the AV (and possibly also the pedestrian model) improves more constraints need to be added to the AV to ensure that collision scenarios exist.

Detecting pedestrians from onboard a moving vehicle is hard. In particular, because pedestrians are often small or occluded. Further, a number of articulated human sensing methods fail to produce stable results on traffic data, because the models are commonly trained and benchmarked on datasets where humans are clearly visible. Detecting the 3D pose and the distance to the human from binocular images is further made harder by the fact that vehicles often move forward at speeds that cause a number of 3D reconstruction methods to fail. Paper V motivates why a dataset of articulated pedestrian motion has not yet been developed, due to the challenges in detecting and reconstructing humans from images captured from moving vehicles.

If articulated pedestrian data is available what kind of improvement in motion forecasting may be expected? Human pose forecasting is commonly treated as a timeseries problem. In paper IV we argue that the common benchmarks are not representative of real-world interactions, where an AI system in general has longer interactions with individuals than benchmarked. We argue that the existing benchmarks restrain methods from utilizing the personalized motion patterns present in each individual's motion and propose a simple timeseries method to remedy this.

In the following, we will provide a more detailed summary of contributions, shortcomings, and outlooks for each paper.

Paper 1 is the first work to capture pedestrian-vehicle interactions, pedestrian-pedestrian interactions, pedestrian-semantics interactions, and the pedestrian's dynamics by modeling human pose. Because a complete dataset of articulated pedestrian motion in traffic is not available this is achieved by combining different training methods and sub-models trained on different datasets. In particular, RL is used to extrapolate beyond the training data with a prior-knowledge motivated reward function, while simultaneously optimizing the supervised objective. Realistic human dynamics are enforced by a model trained on in-lab captured human motion data. This work shows that it is possible to capture the relations that we know exist in traffic even though a complete labeled dataset of pedestrian motion in traffic does not exist yet. Instead, this can be achieved by bridging learning between different datasets and prior knowledge.

Paper 1 takes the first step in modeling many of the effectors in pedestrian motion. The model could be improved in a number of ways. A multi-modal model should be able to better capture the possible changes in the direction of the trajectory or the different variations in human dynamics and behaviors. This could possibly be achieved for example by using *Info-GAIL* [285] instead of supervised learning. *Generative Adversarial Imitation Learning* (GAIL) [120] uses a discriminator as a reward function for the acting policy. Info-GAIL models each individual's behavior as a latent variable. This allows the variation in trajectories exhibited by even just one individual to vary from observation to observation depending on the individual's mood, which can encoded in the latent variable. Further, the learned reward of InfoGAIL would allow for the model to naturally be trained on scenes with no prior pedestrian motion data. Of course, other multi-modality modeling techniques may be used as well. Secondly, the model can further be grounded to ground truth motion data during trajectory forecasting by evaluating the generated poses' distance to the GT human semantic masks. This would possibly require a dense human model to fit on the pedestrians but would provide some feedback on the accuracy of the model-generated poses.

Since more near-collision traffic datasets have become available it would be of interest to learn the human behavior from these datasets and transfer the collision-avoiding behavior model of the pedestrian to a general traffic dataset. Further explicit modeling of interactions, human groups, human gaze, and goal location prediction could be added to the model in paper 1. Language models (LMs) could be used to model the pedestrian's decision process at street crossings. Since the release of paper 1 goal-driven [161, 162] and semantic pedestrian behavior modelling [18, 286] have gained popularity. Further, simulating human poses in semantically meaningful ways in indoor and outdoor environments has also gained popularity in the human pose modeling community [287–290].

Papers ii and iii show that AV testing can be treated as a visual problem, thus allowing for scene generalizable collision scenario generation. The papers show that it is possible to find collision scenarios with interactive and learned pedestrian models instead of the heuristics utilized in previous work. Paper ii shows that for one pedestrian and one AV we can find collisions if the problem is constrained enough and the pedestrian behavior is goal-conditioned. In paper iii the result is extended to multiple pedestrians and vehicles showing how to adapt paper ii to generate realistic collisions with multiple pedestrians. It is not straightforward to generate realistic-looking collision scenarios with multiple pedestrians because the most collision-prone scenarios are exponentially less likely the more abnormalities they contain. Note that the most collision-prone scenarios are abnormal. Paper iii provides a general framework that can be extended with further RL agents to test more advanced AV models.

Papers ii and iii are the first steps in utilizing realistic pedestrian models in AV testing and as such can be improved further. State-of-the-art (STOTA) AV models should be tested in the setup proposed by the papers. This is likely not trivial, because a STOTA AV is expected to drive well thus making collision generation harder. To counter this, a reasonable set of constraints imposed on the AV must be found by possibly extending the framework of paper iii with additional RL agents that pose further constraints on the AV. For example, if the scene contains enough occluded spaces (this could be attained by an RL agent) then the pedestrians could travel through occluded spaces making collision avoidance hard for the AV. A prior distribution for collisions could be learned from a collision dataset or generated from a knowledge base describing different collisions. Finally, weather and pedestrian dynamics, physique, and visuals should be varied.

Paper v points out that 3D reconstruction of on-vehicle gathered images is hard because many reconstruction methods cannot handle large forward motion combined with poor visual quality, moving objects, a lot of unstructured surfaces, and a lot of occlusions. This has led to the popularization of LiDAR to measure the distance to objects. Unfortunately, human sensing from LiDAR has not yet caught up with image-based models, likely because LiDAR pointclouds of humans are quite sparse. Paper v also highlights the discrepancy between articulated human sensing benchmarks and traffic data. Humans are often clearly visible, centered in images, and close to the camera in benchmarks, while in traffic humans often appear far away, are occluded, poorly visible, and off the center. This has since partially been remedied with per frame 2D pose datasets on traffic data such as [291–293]. Similar benchmarks with estimated 3D poses are being developed for LiDAR data [294]. Benchmarks that allow the evaluation of forecasting articulated pedestrians instead of bounding boxes in AV motion planning are yet to be developed.

In paper IV we show that popular human motion forecasting benchmarks are not representative of realistic human-robot interactions, since most human-robot interactions are longer than benchmarked. During long interactions, the prediction model should adapt its prediction to the individual as the interaction continues. We utilize the knowledge that each individual has unique motion patterns to personalize neural human motion predictions with timeseries analysis. Timeseries analysis allows online adaption of the existing model predictions with few parameters. AR models can only improve predictions on a short time horizon because the prediction power of timeseries models decreases exponentially with time. To obtain longer-time horizon improvements, meta-learning [295, 296] could be used to learn the correction of the average motion model for new individuals. In key pose base models [297, 298], target pose prediction should be personalized with meta-learning to allow for personalized long-term predictions. In general human motion datasets are still too small to capture the variation in physique in the population of humans. To scale this up consistent human pose estimation methods that do not require access to a lab are needed. Steps have been taken in the right direction but still more work is needed to attain temporally smooth 3D pose estimates in the wild.

Data gathering of traffic data is currently still expensive, so until this can be scaled up we need to rely on priors to make learning efficient because the existing publicly available datasets are not large enough to capture the variability in the world needed to ensure safe driving. The main issue in data gathering is that we unfortunately currently rely on supervised learning for a large number of tasks (this guarantees that what the models predict is correct) such as articulated human motion forecasting. Therefore to scale up learning unsupervised methods for articulated human sensing should be explored further to popularize articulated human motion modelling in AV planning and testing.

Here we have noted discrepancies in the human models utilized for motion prediction in AV's, pedestrian trajectory prediction, human motion prediction and gait recognition. This suggests that more interdisciplinary work would lead to improved models in all fields. This could hopefully be achieved with joint conferences and benchmarks, as currently a lot of related work is not communicated across different research communities.

Finally, this thesis has shown that it is not trivial to find realistic, collision-prone scenarios for AVs. But it is in fact possible with visual semantically reasoning frameworks as suggested in Paper III. Modeling pedestrians accurately in autonomous vehicle testing is crucial if we wish to ensure the safety of real pedestrians in deployment.

7.1 Annotation

7.1.1 Mathematical Annotation

Symbol	Meaning
α	phase in PFNN
$\boldsymbol{\alpha}$	vector of control phases in PFNN
$\alpha_1, \alpha_2, \alpha_3, \alpha_4$	control phases in PFNN
$\mathbf{A}$	matrix describing a pointcloud Procrutes
$\mathbf{A}_\Theta$	learnable adjacency matrix in GCN
A_{π^*}	advantage function
$\mathbf{a}$	action in RL
a	parameter is PreLU
a_i	parameters in timeseries analysis
$\mathbf{a}_0^i$	action at timestep 0 in the i-th samples trajectory
$\mathbf{a}_1^i$	action at timestep 1 in the i-th samples trajectory
$\mathbf{a}_t^i$	action at timestep t in the i-th samples trajectory
$\mathbf{a}_{T_s}^i$	action at timestep T_s in the i-th samples trajectory
$\mathbf{a}_t$	action at timestep t in RL
β	scaling in Procrustes
β^*	optimal scaling in Procrustes
β_1	update rate of moving average in ADAM
β_2	update rate of second moment estimator in ADAM
$\mathbf{B}$	matrix describing a pointcloud Procrutes
b	baseline the distance between two cameras on a binocular rig.
$\mathbf{b}_1$	per joint belief map in human pose reconstruction at stage 1
$\mathbf{b}_t$	per joint belief map in human pose reconstruction at stage t
C_l	number of limbs
$\mathbf{c}$	constant in MLE
c	covariance function in timeseries
$\mathbf{c_i}$	the i-th class
$\mathbf{c_l}$	left camera's position in global coordinate system
$\mathbf{c_r}$	right camera's position in global coordinate system
$\mathbf{c_t}$	cell state in LSTM
c_t	element of $\mathbf{c_t}$
$\hat{c}_t$	temporary or updated cell state in LSTM
$\mathbf{c_x}$	camera's position in global coordinate system
$\mathbf{c_{x,y}}$	ground truth semantic class label at (x, y)
Δ_g	eigenvalues of $\mathbf{L}$
δ	Krockner's delta function
$\mathbf{D}$	diagonal matrix of the diagonal of $\mathbf{G}$
$\hat{\mathbf{D}}$	Diagonal matrix in Procrustes Analysis
$\mathcal{D}$	dataset
D	depth on an image in convolution
d	disparity
$\mathbf{d}_t$	semantic segmentation found by DilationalNet-10 of frame t
ϵ	machine zero in ADAM
ϵ_t	white noise in timeseries

η	learning rate
e_l	epipolar line in left image
e_r	epipolar line in right image
F	function value to be interpolated in bilinear interpolation
F_i	bilinear interpolation
$F_{i,1}$	bilinear interpolation along x-axis on first row
$F_{i,2}$	bilinear interpolation along x-axis on second row
f	focal length
$\mathbf{f}_\alpha$	phase function
f_γ	activation function pointnet
f_σ	nonlinearity in GCN
f_Θ	parametric function approximator in ML
f_a	activation function in perceptron
$\mathbf{f}_g$	filter in graph convolution
f_h	pointnet location encoding
f_l	likelihood
$\mathbf{f}_n$	observation vector of n timesteps in PEM
$\mathbf{f}_{\mathbf{NN}}$	the ANN that forecasts poses in PFNN
f_p	pointnet function
$\mathbf{f}_R$	function mapping to 3D joints DMHS.
f_r	reward frunction
$\mathbf{f}_{t-1,t}$	optical flow from frame $t-1$ to t
f_w	weights of interpolation in PFNN
$\boldsymbol{\Gamma}$	rotation in Procrustes
$\boldsymbol{\Gamma}^*$	optimal rotation in Procrustes
$\boldsymbol{\Gamma}_{\mathbf{o}}$	rotation and reflection in Procrustes
$\boldsymbol{\Gamma}_{\mathbf{o}}^*$	optimal rotation and reflection in Procrustes
γ	forgetting rate in RL
$\boldsymbol{\gamma}$	translation in Procrustes
$\boldsymbol{\gamma}^*$	optimal translation in Procrustes
$\mathbf{G}$	graph adjacency matrix
$G_{i,j}$	element of the matrix G
$\mathbf{g_t}$	classifier in inference machines
g_t^c	forget gating in LSTM
g_t^h	output gating in LSTM
g_t^u	input gating in LSTM
$g_{t,k}$	partial derivative with respect to the k-th parameter in the t-th optimization step
H	height an image in convolution
h	height of a window in SSD- disparity estimation
h_k	height of a convolution filter
$\mathbf{h_t}$	hidden layer in RNNS, the output of STGRU in GRFP.
h_t	output of a single RNN or LSTM unit, a component of $\mathbf{h_t}$
$\mathbf{I}$	identity matrix
I	image in convolution
I_t	frame t in GRFP
i	iterator
J	general ML loss
J_P	loss of Procrutes Analaysis
J_{RL}	RL loss
j	iterator

$\mathbf{K}$	camera intrinsic matrix
K	convolutional filter
k	iterator, number of anchor boxes in FRCNN
λ	forgetting factor in timeseries analysis
$\mathbf{L}$	Laplace matrix of a graph
L	negative log likelihood
l	layer in neural network
$\mathbf{l}_1$	PAF at inference stage 1
$\mathbf{l}_t$	PAF at inference stage t
$\mathbf{l}_{T_l}$	PAF at last inference stage
μ	initial state distribution in RL
M	dimension of input vector in perceptron, dimensionality of data in NNs
M_C	number of classes
M_c	dimension of cell state in LSTM
M_h	dimension of hidden layer in an RNN
M_i	number of 3D points in bundle adjustment
M_j	number of cameras in bundle adjustment
$m_{0,k}$	moving average of $g_{0,k}$
$m_{t,k}$	moving average of $g_{t,k}$
$\hat{m}_{t,k}$	unbiased mean estimator of $g_{t,k}$
M_s	number of samples
N	number of datapoints in pointcloud, number of RL agents in MARL
$\mathcal{N}$	Normal distribution
N_Θ	number of parameters
N_a	dimensionality of action space
N_B	number of body part segmentations
N_b	number of datapoints in a random batch
N_C	number of output channels in STS-GCN
N_c	number of input channels in STS-GCN
$N_\mathcal{D}$	number of datapoints in dataset
N_g	number of vertices in a graph
N_h	dimensionality of layer l's output
N_i	number of points in class c_i
N_J	number of joints
N_l	number of layers in a network
N_R	number of dimensions in 3D pose features
N_s	dimensionality of state space
N_u	dimensionality of layer l's input
N_t	number of timesteps in STS-GCN
N_y	dimensionality of labels $\mathbf{y}$
N_z	size of softmax function input
n	number of datapoints in online parameter estimation
n_p	number of points in a Pointnet group
π	policy in RL
π^*	optimal policy in RL
π_Θ	parametric policy in RL
ϕ_t	past observations in PEM
$\boldsymbol{\psi_t}$	classifier in openpose
$\boldsymbol{\psi'_t}$	classifier in DMHS
$\mathbf{P}$	camera matrix

P	order of an AR process
$\mathbf{P}_j$	j-th camera matrix
$\mathbf{P}_n$	inverted covariance matrix in online regression
p	probability
$\boldsymbol{p}$	A point in a dataset
$\mathbf{p}_1$	3D point
$\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$	four points on a grid
$p_{1,x}, p_{2,x}, p_{3,x}, p_{4,x}$	x-coordinates of $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$
$p_{1,y}, p_{2,y}, p_{3,y}, p_{4,y}$	y-coordinates of $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$
$\boldsymbol{p}_i$	A point in a random patch
Q	area of integration over trajectories in RL
Q_π	q-value function of policy π
Q_{π^*}	optimal q-value function
$\boldsymbol{\rho}_t$	classifier in openpose
$\boldsymbol{\rho}'_t$	classifier in dmhs
$\mathbf{R}$	camera's rotation matrix
$\mathbf{r}_t$	reconstruction features DMHS
r_t	reward at timestep t
r_t^i	reward at timestep t in the i-th trajectory
$\boldsymbol{\Sigma}$	covariance matrix in MLE
σ	sigmoid function
σ_l	variance in MLE
σ_w	variance of white noise in timeseries analysis
S	Number of steps of prediction in trajectory forecasting
$\mathbf{s}_0^i$	state at timestep 0 in the i-th samples trajectory
$\mathbf{s}_1^i$	state at timestep 1 in the i-th samples trajectory
$\mathbf{s}_{T_s+1}^i$	state at timestep $T_s + 1$ in the i-th samples trajectory
s_r	dilational stride rate
$\mathbf{s}_t$	state in RL
Θ	learnable parameters in a general ML loss
$\boldsymbol{\Theta}_a$	vector of AR parameters
Θ_{MLE}	MLE of the parameters Θ
$\boldsymbol{\Theta}_t$	online parameter estimate
Θ_t	parameters in SGD
$\Theta_{t,k}$	k-th parameter on the t-timestep in SGD
τ	a trajectory or roll-out in RL
T_B	number of stages of joint position estimation in DMHs
T_G	number of frames in a video
T_h	last timestep in RNNs
T_{in}	the number of past timesteps in human pose forecasting
T_J	number of stages in joint position estimation in OpenPose
T_L	number of stages in PAF estimation in OpenPose
T_M	number of stages in inference machine
T_{out}	the number of predicted timesteps in human pose forecasting
T_s	maximal number of timesteps in sequential RL, could be inf
t	iterator
$\mathbf{U}$	rotational matrix in Procrustes
U	the uniform distribution
$\mathbf{U_g}$	eigenvectors of Graph matrix
$\mathbf{U}_o$	rotational matrix in orthogonal Procrustes

$\mathbf{u}$	input to a perceptron
$\mathbf{V}$	rotational matrix in Procrustes
V_π	value function of policy π
V_{π^*}	optimal value function
$\mathbf{V}_g$	graph networks hidden layer
$\mathbf{V}_o$	rotational matrix in orthogonal Procrustes
$\mathbf{v}_g$	feature vectors of vertices of a graph
$v_{0,k}$	second moment estimator of $g_{0,k}$ in ADAM
$v_{t,k}$	second moment estimator of $g_{t,k}$ in ADAM
$\hat{v}_{t,k}$	unbiased second moment estimator of $g_{t,k}$ in ADAM
W	width of an image in convolution
$\mathbf{W}_\alpha$	control point weights in PFNN
$\mathbf{W}_\Theta$	learnable vertex weights in STS-GCN
$\mathbf{w}$	weight in a perceptron
$\hat{\mathbf{w}}$	weight of hidden layer in RNNs
w	width of a window in Disparity estimation
w_0	bias in perceptron
$\mathbf{w}_{\alpha_1}, \mathbf{w}_{\alpha_2}, \mathbf{w}_{\alpha_3}, \mathbf{w}_{\alpha_4}$	control point weights in PFNN
$\mathbf{w}_\mathbf{c}$	weight in forgetting gate LSTM
$\mathbf{w}_{\hat{\mathbf{c}}}$	weight in update state LSTM
$\hat{\mathbf{w}}_\mathbf{c}$	cell weight in forgetting gate LSTM
$\hat{\mathbf{w}}_{\hat{\mathbf{c}}}$	cell weight in update state LSTM
$w_{c,0}$	bias in forgetting gate LSTM
$w_{\hat{c},0}$	bias in forgetting gate LSTM
$\mathbf{w}_\mathbf{h}$	weight in output gate LSTM
$\hat{\mathbf{w}}_\mathbf{h}$	cell weight in output gate LSTM
$w_{h,0}$	bias in output gate LSTM
w_k	width of a convolution filter
$\mathbf{w}_l$	weight of l-th layer in an ANN
$\mathbf{w}_{NN}$	NN weights in PFNN
$\mathbf{w}_\mathbf{t}$	warped segmentation in GRFP
$\mathbf{w}_\mathbf{u}$	weight in input gate LSTM
$\hat{\mathbf{w}}_\mathbf{u}$	cell weight in input gate LSTM
$w_{u,0}$	bias in input gate LSTM
$\boldsymbol{\xi}_{\boldsymbol{t}}$	classifier in DMHS
$\mathbf{X}$	a 3D point
X	a 3D point, random variable
$\mathbf{X}'$	an incorrectly recovered 3D point
$\mathbf{X}^*$	ground truth joint 3D positions
$\hat{\mathbf{X}}$	estimated joint 3D positions
$\{\tilde{\mathbf{X}}_1 \dots \tilde{\mathbf{X}}_N\}$	estimated 3D points
$\mathbf{X}_\mathbf{i}$	a 3D point
$\mathbf{X}^*_\mathbf{i}$	ground truth i-th joint 3D position
$\hat{\mathbf{X}}_\mathbf{i}$	an estimated i-th joint 3D position
$\mathbf{X}_{\text{PFNN}}$	PFNN input vector
$\mathbf{x}$	a point in an image
x	pixel's horizontal coordinate in an image
$\mathbf{x}_1 \dots \mathbf{x}_{\mathbf{N_J}}$	positions of 2D joints
x_1	the object's x-coordinate in the left image in triangulation
x_2	the object's x-coordinate in the right image in triangulation

$\mathbf{x_b}$	image features in the first stage of DMHS
$\mathbf{x'_b}$	image features in DMHS
$\mathbf{x_{i,j}}$	a point in an image
$\mathbf{x_j}$	image features in OpenPose
$\mathbf{x_l}$	a point in the left image
$\mathbf{x_r}$	a point in the right image
$\mathbf{x'_r}$	projection of an incorrectly reconstructed point $\mathbf{X}'$
$\mathbf{x_z}$	image features in PoseMachines
$\mathbf{Y}$	output of convolution
$\mathbf{Y}_{\text{PFNN}}$	PFNN output vector
Y_1, Y_2	random variables in timeseries analysis
Y_t	random variable in timeseries analysis
$\mathbf{y}$	labels in MLE
y	pixel's vertical coordinate in an image
$\mathbf{y}_i$	i-th datapoints labels in MLE
y_n	n-th observation in PEM
y_p	output of a perceptron
y_t	realisation of Y_t
$\hat{y}_t$	one step prediction of Y_t
$\mathbf{z}$	a vector, input to softmax
z	depth from camera to object
z_i	vector $\mathbf{z}$ i-th element, input to softmax
z^i	depth of point $\mathbf{X}_i$
z_j	vector $\mathbf{z}$ j-th element, input to softmax
$\mathbf{z_t}$	semantic segmentation estimate at stage t DMHS
$\mathbf{z}_{x,y}$	GRFP's estimated confidence for different classes at (x, y).

7.1.2 Abbreviations

Symbol	Meaning
ANN	Artificial Neural Networks
AV	Autonomous Vehicles
BA	Bundle Adjustment
CE	Cross Entropy error
CNN	Convolutional Neural Network
CPG	Central Pattern Generators
DMHS	Deep Multitask Architecture for Fully Automatic 2D and 3D Human Sensing
ELU	Exponential Rectified Linear Unit
FRCNN	Faster RCNN
GAIL	Generative Adversarial Inverse Learning
GCN	Graph Convolutional Network
iou	intersection over union
LiDAR	Light Detection And Ranging Sensor
ML	Machine Learning
MLE	Maxmimum Likelihood Estimation
MoCap	Motion Capture
MVS	Multi view Stereo
PFNN	Phase Functional Neural Network

PreLU	Parametric Rectified Linear Unit
RANSAC	Random Sample Consensus
RCNN	Region Convolutional Neural Network
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RNN	Recurrent Neural Networks
SfM	Structure from Motion
SGD	Stochastic Gradient Descent
SIFT	Scale Invariant Feature Transform
Sim2Real	Simulated to Real
SLAM	Simultaneous Localization and Mapping
SNR	Signal to noise ratio
SSD	Sum of Squared Differences
STSGCN	Spatio-Temporal Separable Graph Convolutional Network
STOTA	state of the art
TCN	Temporal Convolutional Network
VRU	Vulnerable Road User

7.1.3 Abbreviations of Conferences

Symbol	**Meaning**
AAAI	Association for the Advancement of Artificial Intelligence
ACCV	Asian Conference on Computer Vision
ACM	Association for Computing Machinery
CoRL	Conference on Robot Learning
CVPR	Computer Vision and Pattern Recognition
ECCV	European Conference on Computer Vision
ICCAR	International Conference on Control, Automation and Robotics
ICCV	International Conference on Computer Vision
ICLR	International Conference on Learning Representations
ICML	International Conference on Machine Learning
ICPR	International Conference on Pattern Recognition
ICRA	International Conference on Robotics and Automation
IROS	International Conference on Intelligent Robots and Systems
ITSC	International Conference on Intelligent Transportation Systems
IV	Intelligent Vehicles Symposium
NeuRIPS	Conference on Neural Information Processing Systems
RO-MAN	International Symposium on Robot and Human Interactive Communication
TOG	Transactions On Graphics
WACV	Winter Conference on Applications of Computer Vision

Bibliography

1. Majecka, B. Statistical models of pedestrian behaviour in the forum. *Master's thesis, School of Informatics, University of Edinburgh* (2009).
2. Bae, I. & Jeon, H.-G. *A Set of Control Points Conditioned Pedestrian Trajectory Prediction* in *Proceedings of the AAAI Conference on Artificial Intelligence* **37** (2023), 6155–6165.
3. Shen, M., Habibi, G. & How, J. P. *Transferable pedestrian motion prediction models at intersections* in *2018 IEEE/RSJ IROS* (2018), 4547–4553.
4. Li, C., Meng, Y., Chan, S. H. & Chen, Y.-T. *Learning 3d-aware egocentric spatial-temporal interaction via graph convolutional networks* in *2020 IEEE ICRA* (2020), 8418–8424.
5. Rhinehart, N., McAllister, R., Kitani, K. & Levine, S. *Precog: Prediction conditioned on goals in visual multi-agent settings* in *Proceedings of the IEEE/CVF ICCV* (2019), 2821–2830.
6. Yao, Y., Atkins, E., Johnson-Roberson, M., Vasudevan, R. & Du, X. Bitrap: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation. *IEEE Robotics and Automation Letters* **6,** 1463–1470 (2021).
7. Huang, Z., Hasan, A., Shin, K., Li, R. & Driggs-Campbell, K. Long-term pedestrian trajectory prediction using mutable intention filter and warp LSTM. *IEEE Robotics and Automation Letters* **6,** 542–549 (2020).
8. Deo, N. & Trivedi, M. M. Trajectory forecasts in unknown environments conditioned on grid-based plans. *arXiv preprint arXiv:2001.00735* (2020).
9. Zhao, H. *et al. Tnt: Target-driven trajectory prediction* in *CoRL* (2021), 895–904.
10. Mangalam, K. *et al. It is not the journey but the destination: Endpoint conditioned trajectory prediction* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16* (2020), 759–776.
11. Li, J. *et al.* EvolveHypergraph: Group-Aware Dynamic Relational Reasoning for Trajectory Prediction. *arXiv preprint arXiv:2208.05470* (2022).

12. Chen, Y., Liu, C., Mei, X., Shi, B. E. & Liu, M. *HGCN-GJS: Hierarchical Graph Convolutional Network with Groupwise Joint Sampling for Trajectory Prediction* in *2022 IEEE/RSJ IROS* (2022), 13400–13405.

13. Djuric, N. *et al. Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving* in *Proceedings of the IEEE/CVF WACV* (2020), 2095–2104.

14. Liang, M. *et al. Pnpnet: End-to-end perception and prediction with tracking in the loop* in *Proceedings of the IEEE/CVF CVPR* (2020), 11553–11562.

15. Luo, Y., Cai, P., Lee, Y. & Hsu, D. Gamma: A general agent motion model for autonomous driving. *IEEE Robotics and Automation Letters* **7**, 3499–3506 (2022).

16. Ma, Y. *et al. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents* in *Proceedings of the AAAI Conference on Artificial Intelligence* **33** (2019), 6120–6127.

17. Zhu, Y. *et al.* Robust trajectory forecasting for multiple intelligent agents in dynamic scene. *arXiv preprint arXiv:2005.13133* (2020).

18. Sriram, N., Liu, B., Pittaluga, F. & Chandraker, M. *Smart: Simultaneous multi-agent recurrent trajectory prediction* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16* (2020), 463–479.

19. Zhao, T. *et al. Multi-agent tensor fusion for contextual trajectory prediction* in *Proceedings of the IEEE/CVF CVPR* (2019), 12126–12134.

20. Luo, K. *et al.* Safety-oriented pedestrian motion and scene occupancy forecasting. *arXiv preprint arXiv:2101.02385* (2021).

21. Fang, L., Jiang, Q., Shi, J. & Zhou, B. *Tpnet: Trajectory proposal network for motion prediction* in *Proceedings of the IEEE/CVF CVPR* (2020), 6797–6806.

22. Park, S. H. *et al. Diverse and admissible trajectory forecasting through multimodal context understanding* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16* (2020), 282–298.

23. Van der Heiden, T., Shankar-Nagaraja, N., Weiß, C. & Gavves, E. SafeCritic: Collision-Aware Trajectory Prediction.

24. Yang, T., Nan, Z., Zhang, H., Chen, S. & Zheng, N. *Traffic agent trajectory prediction using social convolution and attention mechanism* in *2020 IEEE IV* (2020), 278–283.

25. Cheng, H., Liao, W., Yang, M. Y., Rosenhahn, B. & Sester, M. Amenet: Attentive maps encoder network for trajectory prediction. *ISPRS Journal of Photogrammetry and Remote Sensing* **172**, 253–266 (2021).

26. Giuliari, F., Hasan, I., Cristani, M. & Galasso, F. *Transformer networks for trajectory forecasting* in *2020 25th ICPR* (2021), 10335–10342.

27. Anderson, C., Vasudevan, R. & Johnson-Roberson, M. Off the Beaten Sidewalk: Pedestrian Prediction in Shared Spaces for Autonomous Vehicles. *IEEE Robotics and Automation Letters* **5**, 6892–6899 (2020).

28. Salzmann, T., Ivanovic, B., Chakravarty, P. & Pavone, M. *Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data* in *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XVIII* **12363** (Springer, 2020), 683–700.

29. Hamandi, M., D'Arcy, M. & Fazli, P. *DeepMoTIon: Learning to Navigate Like Humans* in *28th IEEE RO-MAN 2019, New Delhi, India, October 14-18, 2019* (IEEE, 2019), 1–7.

30. Yao, X., Zhang, J. & Oh, J. Following Social Groups: Socially Compliant Autonomous Navigation in Dense Crowds. *arXiv preprint arXiv:1911.12063* (2019).

31. Chen, Y., Liu, C., Shi, B. E. & Liu, M. Robot Navigation in Crowds by Graph Convolutional Networks With Attention Learned From Human Gaze. *IEEE Robotics and Automation Letters* **5**, 2754–2761 (2020).

32. Ivanovic, B., Leung, K., Schmerling, E. & Pavone, M. Multimodal Deep Generative Models for Trajectory Prediction: A ConditionalVariational Autoencoder Approach. *IEEE Robotics and Automation Letters* **6**, 295–302 (2021).

33. Girgis, R. *et al. Latent Variable Sequential Set Transformers for Joint Multi-Agent Motion Prediction* in *The Tenth ICLR 2022, Virtual Event, April 25-29, 2022* (OpenReview.net, 2022).

34. Li, L. L. *et al. End-to-end contextual perception and prediction with interaction transformer* in *2020 IEEE/RSJ IROS* (2020), 5784–5791.

35. Chou, F.-C. *et al. Predicting motion of vulnerable road users using high-definition maps and efficient convnets* in *2020 IEEE IV* (2020), 1655–1662.

36. Tang, B. *et al.* Collaborative uncertainty benefits multi-agent multi-modal trajectory forecasting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

37. Huang, X. *et al.* DiversityGAN: Diversity-aware vehicle motion prediction via latent semantic sampling. *IEEE Robotics and Automation Letters* **5**, 5089–5096 (2020).

38. Jiang, B. *et al.* Perceive, interact, predict: Learning dynamic and static clues for end-to-end motion prediction. *arXiv preprint arXiv:2212.02181* (2022).

39. Tang, C. & Salakhutdinov, R. R. Multiple futures prediction. *NeurIPS* **32** (2019).

40. Manish, Dohare, U. & Kumar, S. *A Survey of Vehicle Trajectory Prediction Based on Deep Learning Models* in *Proceedings of Third International Conference on Sustainable Expert Systems: ICSES 2022* (2023), 649–664.

41. Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B. & Moutarde, F. *THOMAS: Trajectory Heatmap Output with learned Multi-Agent Sampling* in *ICLR* (2022).

42. Zeng, W. *et al. DSDNet: Deep Structured Self-driving Network* in *Computer Vision – ECCV 2020* (Springer International Publishing, Cham, 2020), 156–172.

43. Casas, S., Gulino, C., Liao, R. & Urtasun, R. *Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data* in *2020 IEEE ICRA* (2020), 9491–9497.

44. Messaoud, K., Deo, N., Trivedi, M. M. & Nashashibi, F. *Trajectory prediction for autonomous driving based on multi-head attention with joint agent-map representation* in *2021 IEEE IV* (2021), 165–170.

45. Belkada, Y., Bertoni, L., Caristan, R., Mordan, T. & Alahi, A. Do Pedestrians Pay Attention? Eye Contact Detection in the Wild. *arXiv preprint arXiv:2112.04212* (2021).

46. Rasouli, A., Rohani, M. & Luo, J. *Bifold and semantic reasoning for pedestrian behavior prediction* in *Proceedings of the IEEE/CVF ICCV* (2021), 15600–15610.

47. Agrawal, P. & Brahma, P. P. Single shot multitask pedestrian detection and behavior prediction. *arXiv preprint arXiv:2101.02232* (2021).

48. Huynh, M. & Alaghband, G. AOL: Adaptive Online Learning for Human Trajectory Prediction in Dynamic Video Scenes. *In Proceedings of the 31st BMVC,* 262 (2020).

49. Piccoli, F. *et al. Fussi-net: Fusion of spatio-temporal skeletons for intention prediction network* in *2020 54th Asilomar Conference on Signals, Systems, and Computers* (2020), 68–72.

50. Ranga, A. *et al.* VRUNet: Multi-Task Learning Model for Intent Prediction of Vulnerable Road Users. *Electronic Imaging* **32,** 1–10 (2020).

51. Lorenzo, J. *et al. Rnn-based pedestrian crossing prediction using activity and pose-related features* in *2020 IEEE IV* (2020), 1801–1806.

52. Minguez, R. Q., Alonso, I. P., Fernández-Llorca, D. & Sotelo, M. A. Pedestrian path, pose, and intention prediction through gaussian process dynamical models and pedestrian activity recognition. *IEEE Transactions on Intelligent Transportation Systems* **20,** 1803–1814 (2018).

53. Kim, U.-H., Ka, D., Yeo, H. & Kim, J.-H. A real-time vision framework for pedestrian behavior recognition and intention prediction at intersections using 3d pose estimation. *arXiv preprint arXiv:2009.10868* (2020).

54. Mangalam, K., Adeli, E., Lee, K.-H., Gaidon, A. & Niebles, J. C. *Disentangling human dynamics for pedestrian locomotion forecasting with noisy supervision* in *Proceedings of the IEEE/CVF WACV* (2020), 2784–2793.

55. Meng, C., He, X., Tan, Z. & Luan, L. Gait recognition based on 3D human body reconstruction and multi-granular feature fusion. *The Journal of Supercomputing,* 1–20 (2023).

56. Sepas-Moghaddam, A. & Etemad, A. Deep gait recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **45,** 264–284 (2022).

57. Wan, C., Wang, L. & Phoha, V. V. A survey on gait recognition. *ACM Computing Surveys (CSUR)* **51,** 1–35 (2018).

58. Singh, J. P., Jain, S., Arora, S. & Singh, U. P. Vision-based gait recognition: A survey. *IEEE Access* **6,** 70497–70527 (2018).

59. Deng, J. *et al. Imagenet: A large-scale hierarchical image database* in *2009 IEEE CVPR* (2009), 248–255.

60. Radford, A. *et al. Learning transferable visual models from natural language supervision* in *ICML* (2021), 8748–8763.

61. Zhao, W. X. *et al.* A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

62. Brown, T. *et al.* Language models are few-shot learners. *NeurIPS* **33,** 1877–1901 (2020).

63. Sarzynska-Wawer, J. *et al.* Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Research* **304,** 114135 (2021).

64. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

65. Haykin, S. *Neural networks: a comprehensive foundation* (Prentice Hall PTR, 1998).

66. Dosovitskiy, A. *et al.* An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

67. Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. *NeurIPS* **25** (2012).

68. Goodfellow, I., Bengio, Y. & Courville, A. *Deep Learning* `http://www.deeplearningbook.org` (MIT Press, 2016).

69. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

70. Glorot, X. & Bengio, Y. *Understanding the difficulty of training deep feedforward neural networks* in *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (2010), 249–256.

71. Frankle, J. & Carbin, M. *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks* in *ICLR* (2018).

72. Zhang, C., Bengio, S., Hardt, M., Recht, B. & Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM* **64,** 107–115 (2021).

73. Nakkiran, P. *et al.* Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment* **2021,** 124003 (2021).

74. Bishop, C. M. & Nasrabadi, N. M. *Pattern recognition and machine learning* **4** (Springer, 2006).

75. Girshick, R., Donahue, J., Darrell, T. & Malik, J. *Rich feature hierarchies for accurate object detection and semantic segmentation* in *Proceedings of the IEEE CVPR* (2014), 580–587.

76. Ren, S., He, K., Girshick, R. & Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS* **28** (2015).

77. Simonyan, K. & Zisserman, A. *Very deep convolutional networks for large-scale image recognition* in *3rd ICLR* (2015).

78. Yang, Y.-H. *et al. Dense prediction with attentive feature aggregation* in *Proceedings of the IEEE/CVF WACV* (2023), 97–106.

79. Nilsson, D. & Sminchisescu, C. *Semantic video segmentation by gated recurrent flow propagation* in *Proceedings of the IEEE CVPR* (2018), 6819–6828.

80. Yu, F. & Koltun, V. *Multi-scale context aggregation by dilated convolutions* in *ICLR* (2016).

81. Ke, L. *et al. Video mask transfiner for high-quality video instance segmentation* in *ECCV* (2022), 731–747.

82. Minaee, S. *et al.* Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence* **44,** 3523–3542 (2021).

83. Ke, L. *et al. Mask-free video instance segmentation* in *Proceedings of the IEEE/CVF CVPR* (2023), 22857–22866.

84. Wang, W. *et al. Exploring cross-image pixel contrast for semantic segmentation* in *Proceedings of the IEEE/CVF ICCV* (2021), 7303–7313.

85. Ilg, E. *et al. Flownet 2.0: Evolution of optical flow estimation with deep networks* in *Proceedings of the IEEE CVPR* (2017), 2462–2470.

86. Liu, S., Qi, L., Qin, H., Shi, J. & Jia, J. *Path aggregation network for instance segmentation* in *Proceedings of the IEEE CVPR* (2018), 8759–8768.

87. Lin, T.-Y. *et al. Feature pyramid networks for object detection* in *Proceedings of the IEEE CVPR* (2017), 2117–2125.

88. Zhou, T., Porikli, F., Crandall, D. J., Van Gool, L. & Wang, W. A survey on deep learning technique for video segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **45,** 7099–7122 (2022).

89. Li, S., Danelljan, M., Ding, H., Huang, T. E. & Yu, F. *Tracking Every Thing in the Wild* in *ECCV* (2022).

90. Kirillov, A. *et al.* Segment anything. *arXiv preprint arXiv:2304.02643* (2023).

91. Li, S. *et al. OVTrack: Open-Vocabulary Multiple Object Tracking* in *IEEE/CVF CVPR* (2023).

92. Paul, M., Danelljan, M., Mayer, C. & Gool, L. V. *Robust Visual Tracking by Segmentation* in *ECCV* (2022).

93. Zou, X. *et al.* Segment everything everywhere all at once. *arXiv preprint arXiv:2304.-06718* (2023).

94. Zhang, H. *et al.* A Simple Framework for Open-Vocabulary Segmentation and Detection. *arXiv preprint arXiv:2303.08131* (2023).

95. Li, F. *et al.* Semantic-SAM: Segment and Recognize Anything at Any Granularity. *arXiv preprint arXiv:2307.04767* (2023).

96. Zou, X. *et al. Generalized decoding for pixel, image, and language* in *Proceedings of the IEEE/CVF CVPR* (2023), 15116–15127.

97. Liu, S. *et al.* Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499* (2023).

98. Peng, S. *et al. Openscene: 3d scene understanding with open vocabularies* in *Proceedings of the IEEE/CVF CVPR* (2023), 815–824.

99. Ding, R. *et al. PLA: Language-Driven Open-Vocabulary 3D Scene Understanding* in *Proceedings of the IEEE/CVF CVPR* (2023), 7010–7019.

100. Li, C. *et al.* Elevater: A benchmark and toolkit for evaluating language-augmented visual models. *NeurIPS* **35,** 9287–9301 (2022).

101. Qi, C. R., Yi, L., Su, H. & Guibas, L. J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeurIPS* **30** (2017).

102. Qi, C. R., Su, H., Mo, K. & Guibas, L. J. *Pointnet: Deep learning on point sets for 3d classification and segmentation* in *Proceedings of the IEEE CVPR* (2017), 652–660.

103. Sutton, R. S. & Barto, A. G. *Reinforcement learning: An introduction* (MIT press, 2018).

104. Mnih, V. *et al.* Playing Atari with Deep Reinforcement Learning. *arXiv preprint arXiv:1312.5602* (2013).

105. Mnih, V. *et al.* Human-level control through deep reinforcement learning. *Nature* **518,** 529–533 (2015).

106. Hasselt, H. V., Guez, A. & Silver, D. *Deep Reinforcement Learning with Double Q-Learning* in *AAAI Conference on Artificial Intelligence* (2015).

107. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347* (2017).

108. Peng, X. B., Abbeel, P., Levine, S. & Van de Panne, M. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM TOG* **37,** 1–14 (2018).

109. Yu, C. *et al. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games* in *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track* (2022).

110. Mnih, V. *et al. Asynchronous Methods for Deep Reinforcement Learning* in *ICML* (2016).

111. Williams, R. J. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* **8,** 229–256 (1992).

112. Zhang, K., Yang, Z. & Başar, T. Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. *Handbook of Reinforcement Learning and Control* (2019).

113. Dai, Q., Xu, X., Guo, W., Huang, S. & Filev, D. Towards a systematic computational framework for modeling multi-agent decision-making at micro level for smart vehicles in a smart world. *Robotics and Autonomous Systems* **144,** 103859 (2021).

114. Daskalakis, C., Golowich, N. & Zhang, K. *The complexity of markov equilibrium in stochastic games* in *The Thirty Sixth Annual Conference on Learning Theory* (PMLR, 2023), 4180–4234.

115. Filos, A. *et al. PsiPhi-Learning: Reinforcement Learning with Demonstrations using Successor Features and Inverse Temporal Difference Learning* in *ICML* (2021).

116. Lu, Y. *et al.* Imitation Is Not Enough: Robustifying Imitation with Reinforcement Learning for Challenging Driving Scenarios. *arXiv preprint arXiv:2212.11419* (2022).

117. Adams, S. C., Cody, T. & Beling, P. A. A survey of inverse reinforcement learning. *Artificial Intelligence Review* **55,** 4307–4346 (2022).

118. Russell, S. J. *Learning agents for uncertain environments (extended abstract)* in *COLT' 98* (1998).

119. Pomerleau, D. A. Efficient Training of Artificial Neural Networks for Autonomous Navigation. *Neural Computation* **3,** 88–97 (1991).

120. Ho, J. & Ermon, S. *Generative Adversarial Imitation Learning* in *NeurIPS* (2016).

121. Goodfellow, I. J. *et al. Generative Adversarial Nets* in *NeurIPS* (2014).

122. Jakobsson, A. *An introduction to time series modeling* (Studentlitteratur AB, 2019).

123. Ljung, L. *System identification : theory for the user.* (Prentice Hall, 1999).

124. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G. & Black, M. J. SMPL: A Skinned Multi-Person Linear Model. *ACM TOG* **34** (2015).

125. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G. & Black, M. J. in *Seminal Graphics Papers: Pushing the Boundaries, Volume 2* 1st ed. (Association for Computing Machinery, New York, NY, USA, 2023).

126. Xu, H. *et al. Ghum & ghuml: Generative 3d human shape and articulated pose models* in *Proceedings of the IEEE/CVF CVPR* (2020), 6184–6193.

127. Gärtner, E., Andriluka, M., Xu, H. & Sminchisescu, C. *Trajectory Optimization for Physics-Based Reconstruction of 3D Human Pose From Monocular Video* in *Proceedings of the IEEE/CVF CVPR* (June 2022), 13106–13115.

128. Zhang, Y., Black, M. J. & Tang, S. *We Are More Than Our Joints: Predicting How 3D Bodies Move* in *Proceedings of the IEEE/CVF CVPR* (June 2021), 3372–3382.

129. Yang, Z. *et al. S3: Neural Shape, Skeleton, and Skinning Fields for 3D Human Modeling* in *2021 IEEE/CVF CVPR* (2021), 13279–13288.

130. Rasouli, A. & Tsotsos, J. K. Autonomous Vehicles That Interact With Pedestrians: A Survey of Theory and Practice. *IEEE Transactions on Intelligent Transportation Systems* **21,** 900–918 (2020).

131. Camara, F. *et al.* Pedestrian Models for Autonomous Driving Part II: High-Level Models of Human Behavior. *IEEE Transactions on Intelligent Transportation Systems* **22,** 5453–5472 (2021).

132. Narayanan, V., Manoghar, B. M., Dorbala, V. S., Manocha, D. & Bera, A. *Proxemo: Gait-based emotion learning and multi-view proxemic fusion for socially-aware robot navigation* in *2020 IEEE/RSJ IROS* (2020), 8200–8207.

133. Ramakrishna, V., Munoz, D., Hebert, M., Andrew Bagnell, J. & Sheikh, Y. *Pose machines: Articulated pose estimation via inference machines* in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13* (2014), 33–47.

134. Wei, S.-E., Ramakrishna, V., Kanade, T. & Sheikh, Y. *Convolutional pose machines* in *Proceedings of the IEEE CVPR* (2016), 4724–4732.

135. Cao, Z., Simon, T., Wei, S.-E. & Sheikh, Y. *Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields* in *CVPR* (2017).

136. Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S. & Sheikh, Y. A. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).

137. Popa, A.-I., Zanfir, M. & Sminchisescu, C. *Deep multitask architecture for integrated 2d and 3d human sensing* in *proceedings of the IEEE CVPR* (2017), 6289–6298.

138. Kothari, P., Kreiss, S. & Alahi, A. Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Transactions on Intelligent Transportation Systems* **23,** 7386–7400 (2021).

139. Robicquet, A., Sadeghian, A., Alahi, A. & Savarese, S. *Learning social etiquette: Human trajectory prediction in crowded scenes* in *ECCV* **2** (2016), 5.

140. Zhou, B., Wang, X. & Tang, X. *Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents* in *2012 IEEE CVPR* (2012), 2871–2878.

141. Pellegrini, S., Ess, A., Schindler, K. & Van Gool, L. *You'll never walk alone: Modeling social behavior for multi-target tracking* in *2009 IEEE 12th ICCV* (2009), 261–268.

142. Helbing, D. & Molnar, P. Social force model for pedestrian dynamics. *Physical review E* **51,** 4282 (1995).

143. Alahi, A. *et al. Social LSTM: Human Trajectory Prediction in Crowded Spaces* in *Proceedings of the IEEE CVPR* (June 2016).

144. Xu, Y., Piao, Z. & Gao, S. *Encoding crowd interaction with deep neural network for pedestrian trajectory prediction* in *Proceedings of the IEEE CVPR* (2018), 5275–5284.

145. Becker, S., Hug, R., Hubner, W. & Arens, M. *Red: A simple but effective baseline predictor for the trajnet benchmark* in *Proceedings of the ECCV Workshops* (2018), 0–0.

146. Zhang, P., Ouyang, W., Zhang, P., Xue, J. & Zheng, N. *Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction* in *Proceedings of the IEEE/CVF CVPR* (2019), 12085–12094.

147. Zhong, J., Sun, H., Cao, W. & He, Z. Pedestrian motion trajectory prediction with stereo-based 3D deep pose estimation and trajectory learning. *IEEE access* **8,** 23480–23486 (2020).

148. Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S. & Alahi, A. *Social gan: Socially acceptable trajectories with generative adversarial networks* in *Proceedings of the IEEE CVPR* (2018), 2255–2264.

149. Sadeghian, A. *et al. Sophie: An attentive gan for predicting paths compliant to social and physical constraints* in *Proceedings of the IEEE/CVF CVPR* (2019), 1349–1358.

150. Liang, J., Jiang, L., Niebles, J. C., Hauptmann, A. G. & Fei-Fei, L. *Peeking into the future: Predicting future person activities and locations in videos* in *Proceedings of the IEEE/CVF CVPR* (2019), 5725–5734.

151. Liang, J., Jiang, L., Murphy, K., Yu, T. & Hauptmann, A. *The garden of forking paths: Towards multi-future trajectory prediction* in *Proceedings of the IEEE/CVF CVPR* (2020), 10508–10518.

152. Huang, Y., Bi, H., Li, Z., Mao, T. & Wang, Z. *Stgat: Modeling spatial-temporal interactions for human trajectory prediction* in *Proceedings of the IEEE/CVF ICCV* (2019), 6272–6281.

153. Mohamed, A., Qian, K., Elhoseiny, M. & Claudel, C. *Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction* in *Proceedings of the IEEE/CVF CVPR* (2020), 14424–14432.

154. Yu, C., Ma, X., Ren, J., Zhao, H. & Yi, S. *Spatio-temporal graph transformer networks for pedestrian trajectory prediction* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16* (2020), 507–523.

155. Yue, J., Manocha, D. & Wang, H. *Human trajectory prediction via neural social physics* in *ECCV* (2022), 376–394.

156. Lee, N. *et al. Desire: Distant future prediction in dynamic scenes with interacting agents* in *Proceedings of the IEEE CVPR* (2017), 336–345.

157. Ma, W.-C., Huang, D.-A., Lee, N. & Kitani, K. M. *Forecasting interactive dynamics of pedestrians with fictitious play* in *Proceedings of the IEEE CVPR* (2017), 774–782.

158. Xue, H., Huynh, D. Q. & Reynolds, M. *Bi-prediction: Pedestrian trajectory prediction based on bidirectional LSTM classification* in *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)* (2017), 1–8.

159. Amirian, J., Hayet, J.-B. & Pettré, J. *Social ways: Learning multi-modal distributions of pedestrian trajectories with gans* in *Proceedings of the IEEE/CVF CVPR Workshops* (2019), 0–0.

160. Liang, J., Jiang, L. & Hauptmann, A. *Simaug: Learning robust representations from simulation for trajectory prediction* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16* (2020), 275–292.

161. Dendorfer, P., Osep, A. & Leal-Taixé, L. *Goal-gan: Multimodal trajectory prediction based on goal position estimation* in *Proceedings of the ACCV* (2020).

162. Mangalam, K., An, Y., Girase, H. & Malik, J. *From goals, waypoints & paths to long term human trajectory forecasting* in *Proceedings of the IEEE/CVF ICCV* (2021), 15233–15242.

163. Schöller, C., Aravantinos, V., Lay, F. & Knoll, A. What the constant velocity model can teach us about pedestrian motion prediction. *IEEE Robotics and Automation Letters* **5,** 1696–1703 (2020).

164. Saadatnejad, S. *et al.* Are socially-aware trajectory prediction models really socially-aware? *Transportation research part C: emerging technologies* **141,** 103705 (2022).

165. Ridel, D. A., Deo, N., Wolf, D. & Trivedi, M. *Understanding pedestrian-vehicle interactions with vehicle mounted vision: An LSTM model and empirical analysis* in *2019 IEEE IV* (2019), 913–918.

166. Jayaraman, S. K., Tilbury, D. M., Yang, X. J., Pradhan, A. K. & Robert, L. P. *Analysis and prediction of pedestrian crosswalk behavior during automated vehicle interactions* in *2020 IEEE ICRA* (2020), 6426–6432.

167. Xuan, W., Ren, R. & Wang, C. *Multi-agent Interactive Prediction under Challenging Driving Scenarios* in *2021 7th ICCAR* (2021), 6–13.

168. Bai, J., Fang, X., Fang, J., Xue, J. & Yuan, C. *Deep virtual-to-real distillation for pedestrian crossing prediction* in *2022 IEEE 25th ITSC* (2022), 1586–1592.

169. Rasouli, A., Kotseruba, I. & Tsotsos, J. K. *Are they going to cross? a benchmark dataset and baseline for pedestrian crosswalk behavior* in *Proceedings of the IEEE ICCV Workshops* (2017), 206–213.

170. Serrano, S. M., Llorca, D. F., Daza, I. G. & Sotelo, M. Á. *Insertion of Real Agents Behaviors in CARLA Autonomous Driving Simulator* in *In Proceedings of the 6th International Conference on Computer-Human Interaction Research and Applications* (SCITEPRESS, 2022), 23–31.

171. Vasquez, R. & Farooq, B. *Multi-objective autonomous braking system using naturalistic dataset* in *In Proceedings of the 2019 IEEE ITSC* (2019), 4348–4353.

172. Schmitt, P. *et al.* nuReality: A VR environment for research of pedestrian and autonomous vehicle interactions. *arXiv preprint arXiv:2201.04742* (2022).

173. Nikdel, P., Mahdavian, M. & Chen, M. *DMMGAN: Diverse Multi Motion Prediction of 3D Human Joints using Attention-Based Generative Adversarial Network* in *2023 IEEE ICRA* (2023), 9938–9944.

174. Adeli, V. *et al. Tripod: Human trajectory and pose dynamics forecasting in the wild* in *Proceedings of the IEEE/CVF ICCV* (2021), 13390–13400.

175. Usman, M. & Zhong, J. Skeleton-based motion prediction: A survey. *Frontiers in Computational Neuroscience* **16,** 1051222 (2022).

176. Rudenko, A. *et al.* Human motion trajectory prediction: A survey. *The International Journal of Robotics Research* **39,** 895–935 (2020).

177. Martinez, J., Black, M. J. & Romero, J. *On human motion prediction using recurrent neural networks* in *Proceedings of the IEEE CVPR* (2017), 2891–2900.

178. Fragkiadaki, K., Levine, S., Felsen, P. & Malik, J. *Recurrent network models for human dynamics* in *Proceedings of the IEEE ICCV* (2015), 4346–4354.

179. Jain, A., Zamir, A. R., Savarese, S. & Saxena, A. *Structural-rnn: Deep learning on spatio-temporal graphs* in *Proceedings of the ieee CVPR* (2016), 5308–5317.

180. Dilokthanakul, N. *et al.* Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648* (2016).

181. Yan, X. *et al. Mt-vae: Learning motion transformations to generate multimodal human dynamics* in *Proceedings of the ECCV* (2018), 265–281.

182. Zhong, C., Hu, L., Zhang, Z., Ye, Y. & Xia, S. *Spatio-temporal gating-adjacency gcn for human motion prediction* in *Proceedings of the IEEE/CVF CVPR* (2022), 6447–6456.

183. Sofianos, T., Sampieri, A., Franco, L. & Galasso, F. *Space-time-separable graph convolutional network for pose forecasting* in *Proceedings of the IEEE/CVF ICCV* (2021), 11209–11218.

184. Zhang, S., Liu, S., Gao, F. & Chen, S. *Augmented Graph Attention with Temporal Gradation and Reorganization for Human Motion Prediction* in *Intelligent Robotics and Applications* (Springer International Publishing, Cham, 2022), 294–306.

185. Dang, L., Nie, Y., Long, C., Zhang, Q. & Li, G. *Msr-gcn: Multi-scale residual graph convolution networks for human motion prediction* in *Proceedings of the IEEE/CVF ICCV* (2021), 11467–11476.

186. Li, M. *et al. Skeleton-parted graph scattering networks for 3d human motion prediction* in *ECCV* (2022), 18–36.

187. Kipf, T. N. & Welling, M. *Semi-Supervised Classification with Graph Convolutional Networks* in *ICLR* (2017).

188. Bai, S., Kolter, J. Z. & Koltun, V. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv preprint arXiv:1803.01271* (2018).

189. Holden, D., Komura, T. & Saito, J. Phase-functioned neural networks for character control. *ACM TOG* **36,** 1–13 (2017).

190. Peng, X. B., Kanazawa, A., Malik, J., Abbeel, P. & Levine, S. Sfv: Reinforcement learning of physical skills from videos. *ACM TOG* **37,** 1–14 (2018).

191. Bergamin, K., Clavet, S., Holden, D. & Forbes, J. R. DReCon: data-driven responsive control of physics-based characters. *ACM TOG* **38,** 1–11 (2019).

192. Peng, X. B., Ma, Z., Abbeel, P., Levine, S. & Kanazawa, A. Amp: Adversarial motion priors for stylized physics-based character control. *ACM TOG* **40,** 1–20 (2021).

193. Barsoum, E., Kender, J. & Liu, Z. *Hp-gan: Probabilistic 3d human motion prediction via gan* in *Proceedings of the IEEE CVPR workshops* (2018), 1418–1427.

194. Mao, W., Liu, M. & Salzmann, M. *Generating smooth pose sequences for diverse human motion prediction* in *Proceedings of the IEEE/CVF ICCV* (2021), 13309–13318.

195. Barquero, G., Escalera, S. & Palmero, C. BeLFusion: Latent Diffusion for Behavior-Driven Human Motion Prediction. *arXiv preprint arXiv:2211.14304* (2022).

196. Yuan, Y. & Kitani, K. M. *Diverse Trajectory Forecasting with Determinantal Point Processes* in *ICLR* (2019).

197. Bie, X. *et al.* HiT-DVAE: Human Motion Generation via Hierarchical Transformer Dynamical VAE. *arXiv preprint arXiv:2204.01565* (2022).

198. Cai, Y. *et al. A unified 3d human motion synthesis model via conditional variational auto-encoder* in *Proceedings of the IEEE/CVF ICCV* (2021), 11645–11655.

199. Shao, H. *et al. ReasonNet: End-to-End Driving with Temporal and Global Reasoning* in *Proceedings of the IEEE/CVF CVPR* (2023), 13723–13733.

200. Wu, P. *et al.* Trajectory-guided Control Prediction for End-to-end Autonomous Driving: A Simple yet Strong Baseline. *arXiv preprint arXiv:2206.08129* (2022).

201. Codevilla, F., Müller, M., Dosovitskiy, A., López, A. M. & Koltun, V. End-to-End Driving Via Conditional Imitation Learning. *2018 IEEE ICRA,* 1–9 (2017).

202. Sauer, A., Savinov, N. & Geiger, A. Conditional Affordance Learning for Driving in Urban Environments. *arXiv preprint arXiv:abs/1806.06498* (2018).

203. Couto, G. C. K. & Antonelo, E. A. Generative Adversarial Imitation Learning for End-to-End Autonomous Driving on Urban Environments. *2021 IEEE SSCI,* 1–7 (2021).

204. Imamura, R., Seno, T., Kawamoto, K. & Spranger, M. Expert Human-Level Driving in Gran Turismo Sport Using Deep Reinforcement Learning with Image-based Representation. *arXiv preprint arXiv:2111.06449* (2021).

205. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A. & Koltun, V. *CARLA: An Open Urban Driving Simulator* in *In Proceedings of the 2017 CoRL* **78** (PMLR, 2017), 1–16.

206. Chen, D. & Krähenbühl, P. Learning from All Vehicles. *2022 IEEE/CVF CVPR,* 17201–17210 (2022).

207. Shao, H., Wang, L., Chen, R., Li, H. & Liu, Y. *Safety-Enhanced Autonomous Driving Using Interpretable Sensor Fusion Transformer* in *CoRL* (2022).

208. Åström, K. & Hägglund, T. *PID Controllers: Theory, Design, and Tuning* English (ISA - The Instrumentation, Systems and Automation Society, 1995).

209. Ma, W.-C. *et al.* Exploiting Sparse Semantic HD Maps for Self-Driving Vehicle Localization. *2019 IEEE/RSJ IROS,* 5304–5311 (2019).

210. Li, H. *et al.* Delving into the Devils of Bird's-eye-view Perception: A Review, Evaluation and Recipe. *arXiv preprint arXiv:2209.05324* (2022).

211. Singh, A. & Bankiti, V. Surround-View Vision-based 3D Detection for Autonomous Driving: ASurvey. *arXiv preprint arXiv:2302.06650* (2023).

212. Casas, S. *et al. Implicit Latent Variable Model for Scene-Consistent Motion Forecasting* in *ECCV* (2020).

213. Xiong, Y., Ma, W.-C., Wang, J. & Urtasun, R. *Learning Compact Representations for LiDAR Completion and Generation* in *Proceedings of the IEEE/CVF CVPR* (2023).

214. Tong, K., Ajanović, Z. & Stettinger, G. Overview of Tools Supporting Planning for Automated Driving. *2020 IEEE 23rd ITSC,* 1–8 (2020).

215. Li, L. L. *et al.* End-to-end Contextual Perception and Prediction with Interaction Transformer. *2020 IEEE/RSJ IROS,* 5784–5791 (2020).

216. Xuan, W., Ren, R. & Hu, Y. Multi-agent Interactive Prediction under Challenging Driving Scenarios. *2021 7th ICCAR,* 6–13 (2019).

217. Kargar, E. & Kyrki, V. Vision Transformer for Learning Driving Policies in Complex Multi-Agent Environments. *arXiv preprint arXiv:2109.06514* (2021).

218. Casas, S., Gulino, C., Suo, S. & Urtasun, R. The Importance of Prior Knowledge in Precise Multimodal Prediction. *2020 IEEE/RSJ IROS,* 2295–2302 (2020).

219. Can, Y. B., Liniger, A., Paudel, D. P. & Gool, L. V. Structured Bird's-Eye-View Traffic Scene Understanding from Onboard Images. *2021 IEEE/CVF ICCV,* 15641–15650 (2021).

220. Ren, X., Yang, T., Li, E. L., Alahi, A. & Chen, Q. Safety-aware Motion Prediction with Unseen Vehicles for Autonomous Driving. *2021 IEEE/CVF ICCV,* 15711–15720 (2021).

221. Thornton, S. M. Autonomous Vehicle Speed Control for Safe Navigation of Occluded Pedestrian Crosswalk. *arXiv preprint arXiv:1802.06314* (2018).

222. Kamran, D., Lopez, C. F., Lauer, M. & Stiller, C. Risk-Aware High-level Decisions for Automated Driving at Occluded Intersections with Reinforcement Learning. *2020 IEEE IV,* 1205–1212 (2020).

223. Hanna, J. P. *et al.* Interpretable Goal Recognition in the Presence of Occluded Factors for Autonomous Vehicles. *2021 IEEE/RSJ IROS,* 7044–7051 (2021).

224. Fukuda, T., Hasegawa, K., Ishizaki, S., Nobuhara, S. & Nishino, K. *BlindSpotNet: Seeing Where We Cannot See* in *ECCV Workshops* (2022).

225. Cao, Z. & Yun, J. Self-Awareness Safety of Deep Reinforcement Learning in Road Traffic Junction Driving. *arXiv preprint arXiv:2201.08116* (2022).

226. Wiederer, J., Bouazizi, A., Troina, M., Kressel, U. & Belagiannis, V. Anomaly Detection in Multi-Agent Trajectories for Automated Driving. *arXiv preprint arXiv:2110.07922* (2021).

227. Jha, S., Miao, Y., Kalbarczyk, Z. T. & Iyer, R. K. Watch out for the risky actors: Assessing risk in dynamic environments for safe driving. *arXiv preprint arXiv:2110.09998* (2021).

228. Mavrogiannis, C., DeCastro, J. A. & Srinivasa, S. S. Implicit Multiagent Coordination at Unsignalized Intersections via Multimodal Inference Enabled by Topological Braids. *arXiv preprint arXiv:2004.05205* (2020).

229. Wang, Y. *et al.* Multi-Modal 3D Object Detection in Autonomous Driving: A Survey. *International Journal of Computer Vision* **131,** 2122–2152 (2021).

230. Morales, E. S. *et al.* Parallel Multi-Hypothesis Algorithm for Criticality Estimation in Traffic and Collision Avoidance. *2019 IEEE IV,* 2164–2171 (2019).

231. Manivasagam, S. *et al.* LiDARsim: Realistic LiDAR Simulation by Leveraging the Real World. *2020 IEEE/CVF CVPR,* 11164–11173 (2020).

232. Klose, P. & Mester, R. Simulated autonomous driving in a realistic driving environment using deep reinforcement learning and a deterministic finite state machine. *Proceedings of the 2nd International Conference on Applications of Intelligent Systems* (2018).

233. Barrera, A., Beltr'an, J., Guindel, C., Iglesias, J. A. & Garc'ia, F. A. Cycle and Semantic Consistent Adversarial Domain Adaptation for Reducing Simulation-to-Real Domain Shift in LiDAR Bird's Eye View. *2021 IEEE ITSC,* 3081–3086 (2021).

234. Akhauri, S., Zheng, L., Goldstein, T. & Lin, M. C. Improving Generalization of Transfer Learning Across Domains Using Spatio-Temporal Features in Autonomous Driving. *arXiv preprint arXiv:2103.08116* (2021).

235. Ceccarelli, A. & Secci, F. RGB Cameras Failures and Their Effects in Autonomous Driving Applications. *IEEE Transactions on Dependable and Secure Computing* **20,** 2731–2745 (2020).

236. Li, Z. *et al. READ: Large-Scale Neural Scene Rendering for Autonomous Driving* in *AAAI Conference on Artificial Intelligence* (2022).

237. Yang, Z. *et al.* SurfelGAN: Synthesizing Realistic Sensor Data for Autonomous Driving. *2020 IEEE/CVF CVPR,* 11115–11124 (2020).

238. Dokania, S., Subramanian, A., Chandraker, M. & Jawahar, C. *TRoVE: Transforming Road Scene Datasets into Photorealistic Virtual Environments* in *ECCV* (2022).

239. Liang, J., Jiang, L. & Hauptmann, A. G. *SimAug: Learning Robust Representations from Simulation for Trajectory Prediction* in *ECCV* (2020).

240. Mildenhall, B. *et al.* NeRF. *Communications of the ACM* **65,** 99–106 (2020).

241. Tang, S. *et al.* A Survey on Automated Driving System Testing: Landscapes and Trends. *ACM Transactions on Software Engineering and Methodology* (2023).

242. Ploeg, J. *et al. Scenario-Based Safety Assessment Framework for Automated Vehicles* in *Proceedings of the 16th ITS Asia-Pacific Forum* (2018), 713–726.

243. Zhong, Z. *et al.* A survey on scenario-based testing for automated driving systems in high-fidelity simulation. *arXiv preprint arXiv:2112.00964* (2021).

244. Zhang, X. *et al.* Finding critical scenarios for automated driving systems: A systematic mapping study. *IEEE Transactions on Software Engineering* **49,** 991–1026 (2022).

245. Ding, W. *et al.* A survey on safety-critical driving scenario generation—A methodological perspective. *IEEE Transactions on Intelligent Transportation Systems* (2023).

246. Cao, Y. *et al.* Adversarial objects against lidar-based autonomous driving systems. *arXiv preprint arXiv:1907.05418* (2019).

247. Wan, Z. *et al.* Too afraid to drive: systematic discovery of semantic DoS vulnerability in autonomous driving planning under physical-world attacks. *arXiv preprint arXiv:2201.04610* (2022).

248. Rossolini, G. *et al.* On the real-world adversarial robustness of real-time semantic segmentation models for autonomous driving. *arXiv preprint arXiv:2201.01850* (2022).

249. Zhang, W. *et al. Learning to build high-fidelity and robust environment models* in *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part I 21* (2021), 104–121.

250. Sun, H., Feng, S., Yan, X. & Liu, H. X. Corner case generation and analysis for safety assessment of autonomous vehicles. *Transportation research record* **2675,** 587–600 (2021).

251. Ding, W., Xu, M. & Zhao, D. *Cmts: A conditional multiple trajectory synthesizer for generating safety-critical driving scenarios* in *2020 IEEE ICRA* (2020), 4314–4321.

252. Hanselmann, N., Renz, K., Chitta, K., Bhattacharyya, A. & Geiger, A. *King: Generating safety-critical driving scenarios for robust imitation via kinematics gradients* in *ECCV* (2022), 335–352.

253. Rempe, D., Philion, J., Guibas, L. J., Fidler, S. & Litany, O. *Generating useful accident-prone driving scenarios via a learned traffic prior* in *Proceedings of the IEEE/CVF CVPR* (2022), 17305–17315.

254. Liao, Z. *et al. ITGAN: An Interactive Trajectories Generative Adversarial Network Model for Automated Driving Scenario Generation* in *Society of Automotive Engineers (SAE)-China Congress* (2022), 554–566.

255. Abeysirigoonawardena, Y., Shkurti, F. & Dudek, G. *Generating adversarial driving scenarios in high-fidelity simulators* in *2019 ICRA* (2019), 8271–8277.

256. Almanee, S., Wu, X., Huai, Y., Chen, Q. A. & Garcia, J. scenoRITA: Generating Less-Redundant, Safety-Critical and Motion Sickness-Inducing Scenarios for Autonomous Vehicles. *arXiv preprint arXiv:2112.09725* (2022).

257. Zhong, Z., Kaiser, G. & Ray, B. Neural network guided evolutionary fuzzing for finding traffic violations of autonomous vehicles. *IEEE Transactions on Software Engineering* (2022).

258. Karunakaran, D., Worrall, S. & Nebot, E. Efficient falsification approach for autonomous vehicle validation using a parameter optimisation technique based on reinforcement learning. *arXiv preprint arXiv:2011.07699* (2020).

259. Ding, W., Chen, B., Xu, M. & Zhao, D. *Learning to collide: An adaptive safety-critical scenarios generating method* in *2020 IEEE/RSJ IROS* (2020), 2243–2250.

260. Karunakaran, D., Worrall, S. & Nebot, E. *Efficient statistical validation with edge cases to evaluate highly automated vehicles* in *2020 IEEE 23rd ITSC* (2020), 1–8.

261. Wang, J. *et al. Advsim: Generating safety-critical scenarios for self-driving vehicles* in *Proceedings of the IEEE/CVF CVPR* (2021), 9909–9918.

262. Hamdi, A., Müller, M. & Ghanem, B. *SADA: semantic adversarial diagnostic attacks for autonomous applications* in *Proceedings of the AAAI Conference on Artificial Intelligence* **34** (2020), 10901–10908.

263. Nishiyama, D. *et al. Discovering avoidable planner failures of autonomous vehicles using counterfactual analysis in behaviorally diverse simulation* in *2020 IEEE 23rd ITSC* (2020), 1–8.

264. Ghodsi, Z. *et al. Generating and characterizing scenarios for safety testing of autonomous vehicles* in *2021 IEEE IV* (2021), 157–164.

265. Klischat, M. & Althoff, M. *Generating critical test scenarios for automated vehicles with evolutionary algorithms* in *2019 IEEE IV* (2019), 2352–2358.

266. Lowe, G. Sift-the scale invariant feature transform. *Int. J* **2,** 2 (2004).

267. DeTone, D., Malisiewicz, T. & Rabinovich, A. *Superpoint: Self-supervised interest point detection and description* in *Proceedings of the IEEE CVPR workshops* (2018), 224–236.

268. Fischler, M. A. & Bolles, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* **24,** 381–395 (1981).

269. Szeliski, R. *Computer vision: algorithms and applications* (Springer Nature, 2022).

270. Zhao, C., Sun, Q., Zhang, C., Tang, Y. & Qian, F. Monocular depth estimation based on deep learning: An overview. *Science China Technological Sciences* **63,** 1612–1627 (2020).

271. Sarlin, P.-E., DeTone, D., Malisiewicz, T. & Rabinovich, A. *Superglue: Learning feature matching with graph neural networks* in *Proceedings of the IEEE/CVF CVPR* (2020), 4938–4947.

272. Wei, X., Zhang, Y., Li, Z., Fu, Y. & Xue, X. *Deepsfm: Structure from motion via deep bundle adjustment* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16* (2020), 230–247.

273. Xiao, Y., Xue, N., Wu, T. & Xia, G.-S. *Level-S ^2 fM: Structure From Motion on Neural Level Set of Implicit Surfaces* in *Proceedings of the IEEE/CVF CVPR* (2023), 17205–17214.

274. Zhu, Z. *et al. NICE-SLAM: Neural Implicit Scalable Encoding for SLAM* in *Proceedings of the IEEE/CVF CVPR* (June 2022), 12786–12796.

275. Sarlin, P.-E. *et al. Back to the Feature: Learning Robust Camera Localization From Pixels To Pose* in *Proceedings of the IEEE/CVF CVPR* (June 2021), 3247–3257.

276. Zhu, Z. *et al.* Nicer-slam: Neural implicit scene encoding for rgb slam. *arXiv preprint arXiv:2302.03594* (2023).

277. Gower, J. C. Generalized procrustes analysis. *Psychometrika* **40,** 33–51 (1975).

278. Schönemann, P. H. A generalized solution of the orthogonal procrustes problem. *Psychometrika* **31,** 1–10 (1966).

279. Borsani, E., Della Vedova, A. M., Rezzani, R., Rodella, L. F. & Cristini, C. Correlation between human nervous system development and acquisition of fetal skills: An overview. *Brain and Development* **41,** 225–233 (2019).

280. Blumberg, M. S., Coleman, C. M., Gerth, A. I. & McMurray, B. Spatiotemporal structure of REM sleep twitching reveals developmental origins of motor synergies. *Current Biology* **23,** 2100–2109 (2013).

281. Maffei, G., Herreros, I., Sanchez-Fibla, M., Friston, K. J. & Verschure, P. F. The perceptual shaping of anticipatory actions. *Proceedings of the Royal Society B: Biological Sciences* **284,** 20171780 (2017).

282. Geyer, H. & Herr, H. A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities. *IEEE Transactions on neural systems and rehabilitation engineering* **18,** 263–273 (2010).

283. Prilutsky, B. I. & Edwards, D. H. *Neuromechanical modeling of posture and locomotion* (Springer, 2015).

284. Song, S. *et al.* Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation. *Journal of neuroengineering and rehabilitation* **18,** 1–17 (2021).

285. Li, Y., Song, J. & Ermon, S. Infogail: Interpretable imitation learning from visual demonstrations. *NeurIPS* **30** (2017).

286. Ridel, D., Deo, N., Wolf, D. & Trivedi, M. Scene compliant trajectory forecast with agent-centric spatio-temporal grids. *IEEE Robotics and Automation Letters* **5,** 2816–2823 (2020).

287. Yi, H. *et al. MIME: Human-Aware 3D Scene Generation* in *Proceedings of the IEEE/CVF CVPR* (June 2023), 12965–12976.

288. Hassan, M. *et al.* Synthesizing Physical Character-Scene Interactions. *arXiv preprint arXiv:2302.00883* (2023).

289. Zhang, J. Y. *et al. Perceiving 3D Human-Object Spatial Arrangements from a Single Image in the Wild* in *Computer Vision – ECCV 2020* (Springer International Publishing, Cham, 2020), 34–51.

290. Bhatnagar, B. L. *et al. Behave: Dataset and method for tracking human object interactions* in *Proceedings of the IEEE/CVF CVPR* (2022), 15935–15946.

291. Pham, Q. *et al. A*3D Dataset: Towards Autonomous Driving in Challenging Environments* in *2020 IEEE ICRA, ICRA 2020, Paris, France, May 31 - August 31, 2020* (IEEE, 2020), 2267–2273.

292. Zhang, S., Benenson, R. & Schiele, B. *CityPersons: A Diverse Dataset for Pedestrian Detection* in *2017 IEEE CVPR, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017* (IEEE Computer Society, 2017), 4457–4465.

293. Zhang, S. *et al.* WiderPerson: A Diverse Dataset for Dense Pedestrian Detection in the Wild. *IEEE Trans. Multim.* **22,** 380–393 (2020).

294. Windbacher, F., Hödlmoser, M. & Gelautz, M. *Single-Stage 3D Pose Estimation of Vulnerable Road Users Using Pseudo-Labels* in *Image Analysis* (Springer Nature Switzerland, Cham, 2023), 401–417.

295. Gui, L., Wang, Y.-X., Ramanan, D. & Moura, J. M. F. *Few-Shot Human Motion Prediction via Meta-learning* in *ECCV* (2018).

296. Zhu, H., Zhang, L. & Fan, Z. *Personalized individual trajectory prediction via meta-learning* in *Proceedings of the 30th International Conference on Advances in Geographic Information Systems* (2022), 1–2.

297. Kiciroglu, S., Wang, W., Salzmann, M. & Fua, P. *Long Term Motion Prediction Using Keyposes* in *3DV* (2022).

298. Diller, C., Funkhouser, T. & Dai, A. Forecasting Characteristic 3D Poses of Human Actions (2022).

Scientific publications

Paper I

Semantic Synthesis of Pedestrian Locomotion

Maria Priisalu[1](✉), Ciprian Paduraru[2,3], Aleksis Pirinen[1],
and Cristian Sminchisescu[1,3,4]

[1] Department of Mathematics, Faculty of Engineering, Lund University,
Lund, Sweden
{maria.priisalu,aleksis.pirinen,cristian.sminchisescu}@math.lth.se
[2] The Research Institute of the University of Bucharest (ICUB), Bucharest, Romania
ciprian.paduraru@fmi.unibuc.ro
[3] Institute of Mathematics of the Romanian Academy, Bucharest, Romania
[4] Google Research, Lund, Sweden

Abstract. We present a model for generating 3d articulated pedestrian locomotion in urban scenarios, with synthesis capabilities informed by the 3d scene semantics and geometry. We reformulate pedestrian trajectory forecasting as a structured reinforcement learning (RL) problem. This allows us to naturally combine prior knowledge on collision avoidance, 3d human motion capture and the motion of pedestrians as observed e.g. in Cityscapes, Waymo or simulation environments like Carla. Our proposed RL-based model allows pedestrians to accelerate and slow down to avoid imminent danger (e.g. cars), while obeying human dynamics learnt from in-lab motion capture datasets. Specifically, we propose a hierarchical model consisting of a semantic trajectory policy network that provides a distribution over possible movements, and a human locomotion network that generates 3d human poses in each step. The RL-formulation allows the model to learn even from states that are seldom exhibited in the dataset, utilizing all of the available prior and scene information. Extensive evaluations using both real and simulated data illustrate that the proposed model is on par with recent models such as S-GAN, ST-GAT and S-STGCNN in pedestrian forecasting, while outperforming these in collision avoidance. We also show that our model can be used to plan goal reaching trajectories in urban scenes with dynamic actors.

1 Introduction

Pedestrian trajectory prediction is an important sub-problem for safe autonomous driving. Recent 3d traffic datasets [1–6] focus on bounding box detection and prediction of cars and pedestrians. Bounding boxes are popular since they provide information on the location and velocity of the travelling object, and are relatively well suited to model cars, but neglect the detailed motion cues present in pedestrian

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-69532-3_29) contains supplementary material, which is available to authorized users.

H. Ishikawa et al. (Eds.): ACCV 2020, LNCS 12623, pp. 470–487, 2021.
https://doi.org/10.1007/978-3-030-69532-3_29

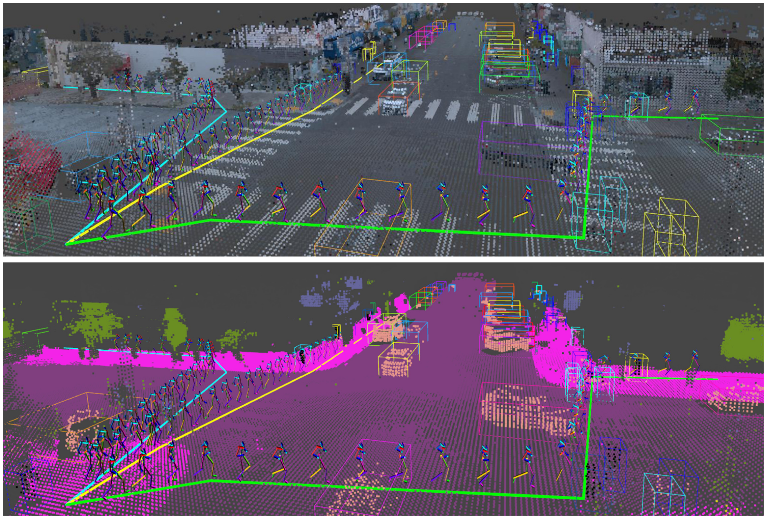


Fig. 1. Pedestrian trajectories and poses generated by our agent on a Waymo scene. RGB and semantic pointclouds of the scene are shown in the top and bottom images, respectively. A local neighborhood of these pointclouds are observed by the agent. Coloured lines on the ground show different trajectories taken by the agent when initialized with varying agent histories, cf. Sect. 2.2. The agent crosses the roads without collisions. Cars and other pedestrians in the scene are shown as positioned in the first frame and are surrounded by bounding boxes for clarity. (Color figure online)

posture. Pedestrian poses compactly model posture and motion cues and have been shown effective in pedestrian intent prediction [7–9]. However, to our knowledge there exists no large-scale datasets with ground truth annotations of pedestrian poses in traffic. Moreover, most previous work in pedestrian pose modelling has been performed without spatial reasoning [7,9] or using action-conditioned human models [8]. In contrast, we formulate pedestrian synthesis as a 3d scene reasoning problem that is constrained by human dynamics and where the generated motion must follow the scene's 3d geometric and semantic properties as seen in Fig. 1. To impose human dynamics, the articulated pose trajectories are conditioned on the current and past poses and velocities.

Specifically, we propose a *semantic pedestrian locomotion* (SPL) agent, a hierarchical articulated 3d pedestrian motion generator that conditions its predictions on both the scene semantics and human locomotion dynamics. Our agent first predicts the next trajectory location and then simulates physically plausible human locomotion to that location. The agent explicitly models the interactions with objects, cars and other pedestrians surrounding it, as seen in Fig. 2. We develop two different pedestrian locomotion generators – one without any restrictions that can roll forward from a given starting location, and one which is additionally conditioned on a target location. The former is useful for simulating generic pedestrian motion in traffic situations, while the latter can be used to control the simulation target, for example when generating high-risk scenarios. Moreover, our model can be used to augment existing traffic datasets with articulated poses. For example, the 3d poses generated by the SPL agent can

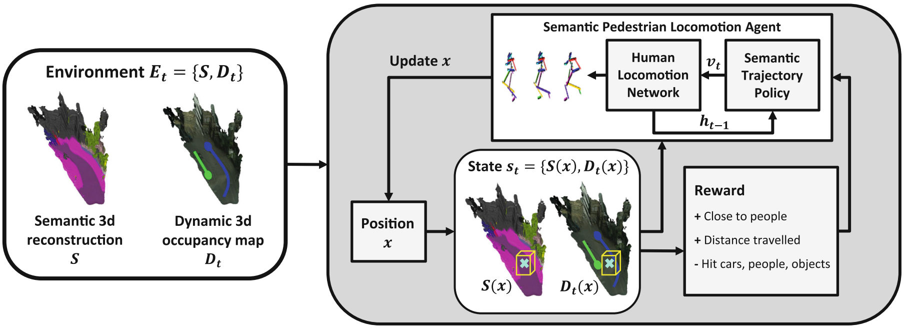

Environment $E_t = \{S, D_t\}$
Semantic 3d reconstruction S
Dynamic 3d occupancy map D_t
Update x
Semantic Pedestrian Locomotion Agent
Human Locomotion Network
v_t
Semantic Trajectory Policy
h_{t-1}
State $s_t = \{S(x), D_t(x)\}$
Position x
$S(x)$
$D_t(x)$
Reward
+ Close to people
+ Distance travelled
- Hit cars, people, objects

Fig. 2. Semantic pedestrian locomotion (SPL) agent and framework. The 3d environment $E_t = \{S, D_t\}$ consists of a semantic map S of static objects and a dynamic occupancy map D_t of cars and people at time t (shown as blue and green trajectories, ellipsoids indicate the positions at time t). The agent observes a top-view projection of a local crop (yellow box) of E_t. A velocity $\boldsymbol{v}_t$ is sampled from the semantic trajectory policy network (STPN). The human locomotion network (HLN) models the articulated movement of the step $\boldsymbol{v}_t$. Note that the STPN observes pose information via the previous hidden state $\boldsymbol{h}_{t-1}$ from the HLN. In training, a reward evaluating the subsequent state is given to the agent. (Color figure online)

be used to produce dense pedestrian predictions by applying a pose conditioned human mesh such as SMPL [10]. Augmented pedestrians can then be produced in semantic segmentation masks by projecting the dense pedestrian mesh onto the image plane. RGB images can be augmented similarly, but this may additionally require a photorealistic style transfer similar to [11,12]. Alternatively, LiDAR augmentations can be generated by sampling [13] from the dense bodies.

Learning to synthesize pedestrian motion is difficult, since the diversity among expert pedestrian trajectories is often limited in the training data, especially for high-risk scenarios. A trajectory generation model trained via imitation learning is unlikely to act reliably in situations that are not present in the training data. This implies e.g. that such an agent will likely behave poorly in near collision scenarios, as these are not present in existing datasets. Recent work on generative adversarial imitation learning (GAIL) [14] has recently gained popularity within trajectory forecasting [15–18] since it models the data distribution rather than cloning expert behaviour. GAIL is an inverse RL method where a policy tries to mimic the experts and the reward function aims to discriminate policy trajectories from expert trajectories. However, as for behaviour cloning, GAIL cannot learn reliable behaviour in situations that are highly different from those available in the training data, since the discriminator will in such cases be able to trivially distinguish between generated and expert trajectories.

To allow our SPL agent to learn also from states outside the training set, in Sect. 2 we pose the trajectory forecasting problem in the framework of reinforcement learning (RL). We extrapolate the learning signal with an optima-preserving reward signal that additionally involves prior knowledge to promote e.g. collision avoidance. We adapt the RL policy sampling process to simultaneously optimize the trajectory forecasting loss and maximize the reward.

Moreover, our analysis in Sect. 2 can be used to adapt any trajectory forecasting model into a robust articulated pedestrian synthesis model. By sampling initial positions of the agent in different locations, all of the spatio-temporal data in the driving dataset can be utilized. Because we train on a large number of different spatial locations and in near-collision scenarios, our motion synthesis model learns to generate plausible trajectories even in states that are far from expert trajectories such as near-collision scenarios. In summary, our contributions are as follows:

- We propose an articulated 3d pedestrian motion generator that conditions its predictions on both the scene semantics and human locomotion dynamics. The model produces articulated pose skeletons for each step along the trajectory.
- We propose and execute a novel training paradigm which combines the sample-efficiency of behaviour cloning with the open-ended exploration of the full state space of reinforcement learning.
- We perform extensive evaluations on Cityscapes, Waymo and CARLA and show that our model matches or outperforms existing approaches in three different settings: i) for pedestrian forecasting; ii) for pedestrian motion generation; and iii) for goal-directed pedestrian motion generation.

1.1 Related Work

In pedestrian trajectory forecasting, social interactions of pedestrians have been modelled with different GAN-based approaches [19,20], by social graphs [21–23], by recurrent networks [24,25] and by temporal convolutions [26,27]. Differently from us, these approaches only model the social interactions of pedestrians and ignore cars and obstacles. An attention model is used by [28] to forecast pedestrian trajectory given environmental features and GAN-based social modelling that neglects cars. Differently from [19–28] we utilize a locomotion model and therefore do not need to learn human dynamics from scratch. All of the mentioned supervised models in pedestrian forecasting can in principle be trained with our proposed methodology (cf. Sect. 2) to extend to unobserved states.

Our model does not rely on action detection (e.g. “walking” or “standing”) of the expert dataset for trajectory forecasting, as opposed to action conditioned intention detection networks [8,29] and motion forecasting models [30,31]. Instead the pedestrian’s future trajectory is conditioned on its past trajectory. A benefit of our approach is thus that it avoids dealing with temporal ambiguities associated with action detection. Recently it has been shown that pedestrian future augmentation can improve pedestrian forecasting features [32]. Our generator produces articulated 3d trajectories on real data, and in comparison to [33] we do not require the recreation of the full dataset in a simulation environment. We note that the goal-reaching version of our model could be utilized with a goal proposal method [34] to provide multiple future augmentations to data.

Human synthesis models for still images [11,18,35–37] aim to synthesize poses in semantically and geometrically plausible ways in images, and have no temporal modelling, but could be used to initialize the SPL's pose trajectories. The works [36,37] model likely locations for humans in images. The models in [11,18,35] synthesize pedestrians with 3d models, but do this only in static scenes. Similar to us, the affordance model of [38] explicitly incorporates 3d scene semantics to propose plausible human poses, but only for static scenes. Synthetic videos are generated in [39] by cropping humans from sample videos and pasting them into target videos followed by visual smoothing with a GAN, but this approach does not guarantee semantic plausibility.

The majority of 3d human pose forecasting models concentrate on predicting future poses given only the past pose history [40–42]. In [43] human pose futures are predicted on a static dataset by forecasting a trajectory, to which poses are fitted by a transformer network. Differently from our work the reasoning is performed in 2d which leads to geometrically implausible failure cases. [44] forecast pedestrian motion by combining a pose predicting GRU [45] with social pooling and a 2d background context layer. Both [43,44] are not readily applicable to driving datasets as they lack modelling of cars and require access to high quality 2d human poses which are in general hard to obtain in driving datasets.

2 Methodology

The pedestrian trajectory forecasting problem on a dataset $\mathcal{D}$ of pedestrian trajectories can be formulated as follows. Let $\boldsymbol{x}_0, \dots, \boldsymbol{x}_t, \boldsymbol{x}_{t+1}, \dots \boldsymbol{x}_T$ be a pedestrian trajectory[1] of length T in $\mathcal{D}$. Given the trajectory $\boldsymbol{x}_0, \dots, \boldsymbol{x}_t$ up to timestep t we would like to predict the pedestrian's position in the next timestep $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{v}_t$, where $\boldsymbol{v}_t$ is the step taken by the pedestrian from $\boldsymbol{x}_t$ to $\boldsymbol{x}_{t+1}$. Each position $\boldsymbol{x}_t$ is associated with a state $\boldsymbol{s}_t$, described in detail in Sect. 2.1, that includes the pedestrian's past trajectory and other relevant scene information at position $\boldsymbol{x}_t$. We denote the density function of the random variable $\boldsymbol{v}_t$ conditioned on $\boldsymbol{s}_t$ as $p(\boldsymbol{v}_t|\boldsymbol{s}_t)$. The prediction task is to estimate $p(\boldsymbol{v}_t|\boldsymbol{s}_t)$ by a parametric function $p_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t)$ where the step forecast is $\hat{\boldsymbol{v}_t} = \max_{\boldsymbol{v}_t} p_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t)$. The maximum likelihood estimate of the model parameters Θ is then given by

$$\Theta^* = \arg\max_{\Theta} \log \mathcal{L}(\Theta|\mathcal{D}) = \arg\max_{\Theta} \sum_{\mathcal{D}} \sum_{t=0}^{T-1} \log p_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t) \tag{1}$$

From the RL perspective on the other hand, an agent has an initial position $\boldsymbol{x}_0$ and takes steps by sampling from a parametric policy: $\boldsymbol{v}_t \sim \pi_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t)$. After taking a step $\boldsymbol{v}_t$ the agent finds itself in a new location $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{v}_t$ and in training receives a reward $R(\boldsymbol{s}_t, \boldsymbol{v}_t)$. The objective is to find a policy π_Θ that maximizes the expected cumulative reward,

$$J(\Theta) = \mathbb{E}_{\pi_\Theta}\left[\sum_{t=0}^{T-1} R(\boldsymbol{s}_t, \boldsymbol{v}_t)\right] \tag{2}$$

[1] The $\boldsymbol{x}_t$ are 2d locations in the movement plane.

Comparing the RL perspective with the standard forecasting formulation, we first note that $\pi_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t) = p_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t)$. Furthermore, the optima of (1) is unchanged if it is multiplied by a function $R(\boldsymbol{s}_t, \boldsymbol{v}_t)$ that obtains its maximum at all $(\boldsymbol{s}_t, \boldsymbol{v}_t) \in \mathcal{D}$, i.e. on the expert trajectories. Thus, assuming that the actions taken by the pedestrians in $\mathcal{D}$ are optimal in the reward function R, we can rewrite the maximum likelihood objective (1) as a Monte Carlo estimate of the policy gradient objective [46], sampled from the expert trajectories $(\boldsymbol{s}_t, \boldsymbol{v}_t) \in \mathcal{D}$:

$$\Theta^* = \arg\max_{\Theta} \sum_{\mathcal{D}} \sum_{t=0}^{T-1} \log \pi_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t) R(\boldsymbol{s}_t, \boldsymbol{v}_t) \tag{3}$$

We can now unify the policy gradient objective (3) and the supervised objective (1) by sampling respectively from $(\tilde{\boldsymbol{s}}_t, \tilde{\boldsymbol{v}}_t) \sim \pi_\Theta$ and $(\boldsymbol{s}_t, \boldsymbol{v}_t) \in \mathcal{D}$. Optimizing (3) while sampling from both the expert trajectories and the current parametric policy equates to iteratively optimizing the policy gradient objective and the maximum likelihood objective. Thus we have shown that (1) can be rewritten as a policy gradient objective assuming a reward function that obtains its optima on $\mathcal{D}$. In Sect. 2.4 we construct a reward function that fulfills this criteria.

By posing the supervised learning problem of pedestrian trajectory forecasting as an RL problem, the detailed human dynamics model HLN becomes part of the observable environment dynamics and does not need to be modelled explicitly in the trajectory prediction model π_Θ. This is a natural way of combining accurate human motion models trained on in-laboratory motion capture data [47,48] with trajectories available in autonomous driving datasets [1–6].

In the following subsections we present our SPL agent, which performs human 3d motion synthesis within two modules. First a semantic pedestrian locomotion network (STPN) samples a step $\boldsymbol{v}_t$ based on $\boldsymbol{s}_t$, and then a human locomotion network (HLN) generates realistic body joint movements to the next position $\boldsymbol{x}_{t+1}$. The HLN is first trained in a supervised fashion (see Sect. 2.3). Then the STPN and HLN modules are combined, and the STPN is trained by alternating[2] between sampling from expert trajectories and from arbitrary states, following the objective (3). Figure 2 provides an overview of the SPL model.

2.1 States and Actions

The agent acts in the voxelized 3d environment $E_t = \{S, D_t\}$ over the time horizon $\{0, \ldots, T\}$, where E_t is a 3d pointcloud reconstruction of a scene with resolution 20 cm × 20 cm × 20 cm. The reconstruction E_t consists of stationary objects S and a dynamic occupancy map D_t of moving objects. Specifically, the dynamic occupancy map marks the timestamps of voxel occupancies by other pedestrians and cars (in separate channels) in the time horizon $\{0, \ldots, T\}$. For past timesteps $0 - t$ the dynamic occupancy map contains the past trajectories of cars and pedestrians, while a constant velocity model is used to predict the future $t+1, \ldots, T$. Further details of D_t are in the supplement. Each 3d point in

[2] See details of the alternating training in the supplement.

E_t is described by a semantic label l and an RGB-color label $\boldsymbol{c}$. We let $E_t(\boldsymbol{x}_t) = \{S(\boldsymbol{x}_t), D_t(\boldsymbol{x}_t)\}$ denote a 5 m × 5 m × 1.8 m rectangular 3d crop of E_t centered at the agent's current position $\boldsymbol{x}_t$ and touching the ground.

The agent's state at time t consists of its external semantic state $\boldsymbol{s}_t$ and the internal locomotion state $\boldsymbol{l}_t$. The external semantic state is defined as

$$\boldsymbol{s}_t = \{E_t^{2d}(\boldsymbol{x}_t), \boldsymbol{v}_{t-N}, \ldots, \boldsymbol{v}_{t-1}, \boldsymbol{d}_v, \boldsymbol{h}_{t-1}\} \tag{4}$$

where $E_t^{2d}(\boldsymbol{x}_t)$ is a top-view projection of $E_t(\boldsymbol{x}_t)$, $\boldsymbol{v}_{t-N}, \ldots, \boldsymbol{v}_{t-1}$ constitute the agent's movement history for the past $N = 12$ timesteps, $\boldsymbol{d}_v$ is the displacement[3] to the closest vehicle, and $\boldsymbol{h}_{t-1}$ is the hidden layer of the HLN (cf. Sect. 2.3) which informs about the agent's posture, pose and acceleration. The locomotion state

$$\boldsymbol{l}_t = \{\boldsymbol{x}_{t-M}, \ldots, \boldsymbol{x}_{t-1}, \boldsymbol{x}_t, \boldsymbol{g}_{t-M}, \ldots \boldsymbol{g}_{t-1}, \boldsymbol{g}_t, \boldsymbol{j}_t, \boldsymbol{i}_t, \boldsymbol{x}_{t+1}, |\boldsymbol{v}_t|\} \tag{5}$$

consists of the past positions $\boldsymbol{x}_{t-M}, \ldots, \boldsymbol{x}_{t-1}$ of the agent ($M = 11$), the current position $\boldsymbol{x}_t$, the past gait characteristics $\boldsymbol{g}_{t-M}, \ldots, \boldsymbol{g}_{t-1}$, the next step's gait $\boldsymbol{g}_t$, the joint positions and velocities $\boldsymbol{j}_t$ and $\boldsymbol{i}_t$, the next trajectory position $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{v}_t$, and the speed $|\boldsymbol{v}_t|$. The gait characteristic $\boldsymbol{g}_t$ is a binary vector indicating if the agent is standing, walking or jogging and is regressed from $|\boldsymbol{v}_t|$. The joint positions $\boldsymbol{j}_t$ are the 3d positions of the root-joint centered 30 BVH joints of the CMU motion capture data [49].

2.2 Semantic Trajectory Policy Network (STPN)

The STPN is a neural network that parametrizes $\pi_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t)$, the velocity distribution of the agent in position $\boldsymbol{x}_t$ with state $\boldsymbol{s}_t$. We factorize $\pi_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t)$ into a Gaussian distribution over speed $|\boldsymbol{v}_t|$, and a multinomial distribution over discretized unit directions $\boldsymbol{u}_t$. Since the agent is acting and observing the world in a regular voxel grid, the movement directions are discretized into the eight directions North (N), North-East (NE) and so on: N, NE, E, SE, S, SW, W, NW, as well as a no-move action. After the velocity $\boldsymbol{v}_t$ is sampled, the agent's next position $\boldsymbol{x}_{t+1}$ is given by the HLN in Sect. 2.3. The new position is often close to $\boldsymbol{x}_t + \boldsymbol{v}_t$ but could be adjusted by the HLN to ensure physical plausibility.

The policy $\pi_\Theta(\boldsymbol{v}_t|\boldsymbol{s}_t)$ is parameterized by a neural network, consisting of a convolutional features extractor, an agent history encoder and two parallel fully connected (FC) layers. The convolutional features extractor consists of two convolutional layers of size $(2, 2, 1)$ with ReLU activations and max pooling. The agent history encoder is a 32-unit LSTM [50] that extracts a temporal feature vector $\boldsymbol{f}_t$ from the agent's past trajectory $\boldsymbol{v}_{t-N}, \ldots, \boldsymbol{v}_{t-1}$. The parallel FC layers both receive[4] as input the convolutional features, the temporal features $\boldsymbol{f}_t$, the displacement vector $\boldsymbol{d}_t$ and the hidden state[5] $\boldsymbol{h}_{t-1}$ of the HLN. The previous unit direction $\boldsymbol{u}_{t-1}$ is added as a prior to the output of the first FC layer,

[3] This is comparable to a pedestrian being aware of cars in its vicinity.

[4] The goal-directed agent additionally includes the direction to the goal at this stage.

[5] The previous hidden state is used, as the HLN is executed after the STPN.

and the result is then fed through a softmax activation to output a probability distribution over the unit directions $\boldsymbol{u}_t$. The second FC layer is activated by the sigmoid function which is scaled with the maximal speed 3 m/s to produce μ_t, the mean of the normal distribution that models the speed taken at time t. Hence $|\boldsymbol{v}_t| \sim \mathcal{N}(\mu_t, \sigma)$, where σ is exponentially decreased from 2 to 0.1 in training. Finally, the sampled velocity $\boldsymbol{v}_t$ is given by $\boldsymbol{v}_t = |\boldsymbol{v}_t|\boldsymbol{u}_t$.

2.3 Human Locomotion Network (HLN)

The HLN produces 3d body joint positions to take a step $\boldsymbol{v}_t$ from $\boldsymbol{x}_t$. The HLN is adapted from [51] with the addition of a velocity regression layer that estimates $\boldsymbol{g}_t$ in (5) from $\boldsymbol{v}_t$. Network weights are learnt following the data and procedure in [51]. The HLN is a phase function network that is conditioned on the walking phase of the body at time t, where the phase varies from 0 to 2π for a full cycle from the right foot touching the ground until the next occurrence of the right foot touching the ground. The HLN regresses $\boldsymbol{j}_{t+1}, \boldsymbol{i}_{t+1} = h(\boldsymbol{l}_t)$, i.e. the joint positions $\boldsymbol{j}_{t+1}$ and velocities $\boldsymbol{i}_{t+1}$, conditioned on the current state $\boldsymbol{l}_t$ (see Sect. 2.1).

The next position $\boldsymbol{x}_{t+1}$ of the agent is set to the plane coordinates of the pelvis joint in $\boldsymbol{j}_{t+1}$ at timestep $t+1$ (the agent is not allowed to move through objects). The HLN architecture consist of three fully connected layers with 512 hidden units per layer and an exponential rectified linear function [52] as the activation function. The last hidden layer $\boldsymbol{h}_t$ is observed by the STPN in the next timestep, informing it of the agent's current posture. Network weights are trained for different walking phases by augmenting surface curvature for constant feet to ground distances from motion capture data as reported in [51].

2.4 Reward Signal

In training the agent's state is evaluated by the reward function $R_t = R(\boldsymbol{x}_t, \boldsymbol{v}_t)$ at each step. We wish to estimate the optimal policy $\pi_{\boldsymbol{\Theta}^\star}(\boldsymbol{v}_t|\boldsymbol{s}_t)$ that maximizes the total expected reward. The reward function is designed so that its maximal value occurs on the expert trajectories, as discussed in Sect. 2. A reward $R_d = 1$ is given for visiting a pedestrian trajectory in the dataset $\mathcal{D}$, otherwise $R_d = 0$. The reward is given only for newly visited locations to promote the agent to move. We also encourage the agent to move close to positions where pedestrians tend to appear. To approximate a pedestrian density map from $\mathcal{D}$ we apply an exponential kernel on the trajectory locations in $\mathcal{D}$, i.e.

$$R_k(\boldsymbol{x}_t, \boldsymbol{v}_t) = \log\left\{\frac{1}{b}\sum_{\boldsymbol{x}^i \in D}\sum_{t'=0}^{T} \exp\{-\|\boldsymbol{x}^i_{t'} - \boldsymbol{x}_{t+1}\|\}\right\} \tag{6}$$

where b is the bandwidth (we set $b = 0.0001$) and the sum is over all pedestrian trajectory positions $\boldsymbol{x}^i$ in the dataset $\mathcal{D}$. We gather the terms that encourage the agent to stay near trajectories in $\mathcal{D}$ as $R_{ped}(\boldsymbol{x}_t, \boldsymbol{v}_t) = R_k(\boldsymbol{x}_t, \boldsymbol{v}_t) + R_d(\boldsymbol{x}_t, \boldsymbol{v}_t)$.

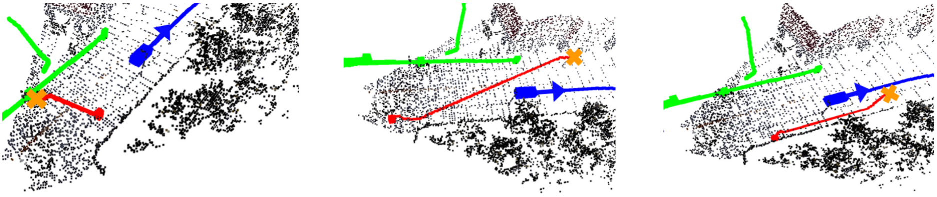


Fig. 3. Several 1-min trajectories of our SPL-goal agent reaching its goal location in orange (maximum distance to goal: 120 m) on the CARLA test set. Car, person and agent trajectories are shown in blue, green and red respectively. *Left:* Agent sharply but safely crossing the street to reach a goal. *Middle:* Agent safely crossing the street as no cars are approaching. *Right:* Agent safely moving along the pavement when given a goal on the road. The shortest path to the goal would involve walking on the road for a longer amount of time, so the agent balances its desire to reach the goal with the risk of being on the road. (Color figure online)

To penalize collisions, let R_v, R_p and R_s be negative indicator functions that active if the agent collides with vehicles, pedestrians and static objects, respectively. The terms are gathered as $R_{coll}(\boldsymbol{x}_t, \boldsymbol{v}_t) = R_v(\boldsymbol{x}_t, \boldsymbol{v}_t) + R_p(\boldsymbol{x}_t, \boldsymbol{v}_t) + R_s(\boldsymbol{x}_t, \boldsymbol{v}_t)$. Note that $R_{ped}(\boldsymbol{x}_t, \boldsymbol{v}_t)$ is only given when $R_p(\boldsymbol{x}_t, \boldsymbol{v}_t) = 0$.

To encourage smooth transitions between the exhibited poses and to penalize heavy effort motions, we penalize the average yaw ϕ (in degrees) of the joints in the agent's lower body as $R_\phi(\boldsymbol{x}_t, \boldsymbol{v}_t) = \max(\min(\phi - 1.2, 0), 2.0)$. Thus the full reward[6] is $R(\boldsymbol{x}_t, \boldsymbol{v}_t) = R_{coll}(\boldsymbol{x}_t, \boldsymbol{v}_t) + R_{ped}(\boldsymbol{x}_t, \boldsymbol{v}_t) + R_\phi(\boldsymbol{x}_t, \boldsymbol{v}_t)$.

When the agent is given a goal location, every step taken towards the goal should provide a reward for the improvement made relative to the initial goal distance. Thus, given a goal location $\boldsymbol{x}_g$ we define

$$R_g(\boldsymbol{x}_t, \boldsymbol{v}_t) = \begin{cases} 1 & \text{if } \|\boldsymbol{x}_{t+1} - \boldsymbol{x}_g\| < \epsilon \\ 1 - \frac{\|\boldsymbol{x}_{t+1} - \boldsymbol{x}_g\|}{\|\boldsymbol{x}_t - \boldsymbol{x}_g\|} & \text{otherwise} \end{cases} \tag{7}$$

where ϵ defines the distance from the goal location to the agent center.[7] The full reward[8] of the goal reaching agent is $R(\boldsymbol{x}_t, \boldsymbol{v}_t) = R_{coll}(\boldsymbol{x}_t, \boldsymbol{v}_t) + R_{ped}(\boldsymbol{x}_t, \boldsymbol{v}_t) + R_\phi(\boldsymbol{x}_t, \boldsymbol{v}_t) + R_g(\boldsymbol{x}_t, \boldsymbol{v}_t)$. Note that the goal-driven reward does not necessarily reach its optima on expert trajectories, as the it is not assumed that $\boldsymbol{x}_g \in \mathcal{D}$.

2.5 Policy Training

With a finite sequence length T, a large number of states are in practice unreachable for the agent with an initial location $\boldsymbol{x}_0$. However, thanks to the RL reformulation the agent can be initialized in any location. By regularly choosing

[6] Each term weighted with the respective weights, $\lambda_v = 1$, $\lambda_p = 0.1$, $\lambda_s = 0.02$, $\lambda_k = 0.01$, $\lambda_d = 0.01$, $\lambda_\phi = 0.001$.

[7] We set $\epsilon = 20\sqrt{2}$ cm, i.e. the agent must overlap the goal area.

[8] The weights except for $\lambda_v = 2$, $\lambda_g = 1$ are the same. The fraction term of R_g is weighted by 0.001.

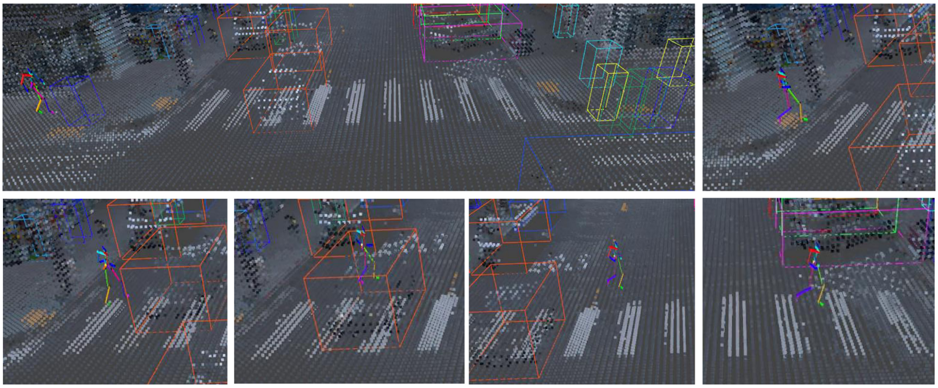


Fig. 4. Subsampled pose sequence in a Waymo test scene, showing the SPL agent walking behind a car (indicated with an orange 3d bounding box) to avoid a collision, and then returning to the crosswalk. A zoomed out view of the scene at the beginning of the agent's trajectory is shown in the top left. (Color figure online)

information dense $\boldsymbol{x}_0$, the number of samples needed to learn critical behaviours such as collision avoidance can be reduced, and thus the agent is initialized in front of cars, near pedestrians, randomly, on pavement and on pedestrians. Agents are trained in Tensorflow [53] using Adam [54] with a batch size of 30 trajectories, learning rate of 10^{-3}, and a discount rate of 0.99.

3 Experiments

The proposed pedestrian motion generation agent is evaluated on both simulated and real data, with and without target goals. The goal-free and goal-directed agents are denoted SPL and SPL-goal, respectively. Since the human locomotion network (HLN) described in Sect. 2.3 imposes realistic human dynamic constraints, we present all results with the HLN performing joint transformations along the trajectories. We compare SPL with the following methods:

- *Behaviour cloning (BC)* is an imitation learning baseline. BC is trained with the same network structure as SPL, but by only sampling from $\mathcal{D}$, i.e. max-likelihood forecasting. The same hyperparameters as for SPL are used.
- *Constant velocity (CV)* models pedestrian motion with a constant velocity, which as shown in [55] is surprisingly effective in many cases. When initialized on a pedestrian it continues with the last step velocity of that pedestrian. When initialized elsewhere, a Gaussian with $\mu = 1.23$ and $\sigma = 0.3$ (same as [56]) is used to estimate speed and the direction is drawn at random.
- *S-GAN* is the Social-GAN [19] used for pedestrian forecasting.
- *S-STGCNN* (*S-STG* in tables) the Social Spatio-Temporal Graph Convolutional Neural Network [23], a pedestrian trajectory forecasting network.
- *ST-GAT* is the Spatial-Temporal Graph Attention Network [22], another recent pedestrian trajectory forecasting network.

- *CARLA-simulated (GT)* are the pedestrians simulated in CARLA, here considered ground truth. These pedestrians follow hand-designed trajectories.

S-GAN, S-STGCNN and ST-GAT are trained with default hyperparameters from the official implementations. We compare SPL-goal with the following:

- *Goal direction (GD)* takes the shortest Euclidean path to the goal.
- *Collision avoidance with deep RL (CADRL)* [57] walks towards the goal location while avoiding moving objects around itself. CADRL is a learning based model for collision avoidance with dynamic obstacles.

3.1 Datasets

Simulated Data from CARLA. The CARLA package [58] is a simulator for autonomous driving. RGB images, ground truth depth, 2d semantic segmentations and bounding boxes of pedestrians and cars are collected from the simulator at 17 fps. Town 1 is used to collect training and validation sets, with 37 and 13 scenes, respectively. The test set consists of 37 scenes from Town 2.

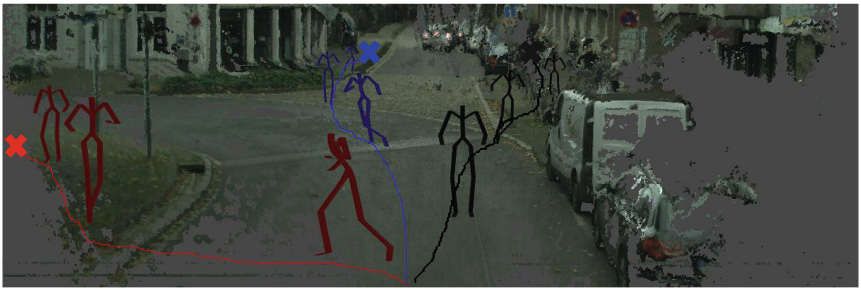


Fig. 5. Multiple SPL-goal agent trajectories generated from the same initial position in Cityscapes. The agent can be seen reaching different goals (marked by crosses). The agent chooses to walk on pavement when nearby.

3D Reconstructions from Cityscapes. This dataset [59] consists of on-board stereo videos captured in German cities. The videos are 30 frames long with a frame rate of 17 fps (video length: 1.76 s). We use GRFP [60] to estimate the semantic segmentation of all frames. The global reconstructions are computed by COLMAP [61] assuming a stereo rig with known camera parameters. The density of the dense reconstructions from COLMAP varies; an example reconstruction can be seen in Fig. 5. Cars and people are reconstructed frame-by-frame from PANnet [62] 2d bounding boxes and instance level segmentation masks. Triangulation is used to infer 3d positions from 2d bounding boxes. The dataset consists of 200 scenes; 100 for training, 50 for validation and 50 for testing.

LiDAR Waymo. The Waymo dataset [2] consist of 200 frame 10 Hz LiDAR 3d scans, traffic agent trajectories and RGB images in 5 directions from the top of a data gathering car. We subsample a dataset of the 100 most pedestrian dense scenes in 50 m radius to the collecting car. We use 70, 10 and 20 scenes for training, validation and testing, respectively. The images are segmented by [63] and the semantic labels are mapped to the 3d scans by the mapping between LiDAR and cameras provided by the Waymo dataset.

3.2 Training and Evaluation Details

In CARLA and Waymo the training sequence length is 30 timesteps, and in testing 300 timesteps (≈17 s). The agents are trained for 20 epochs, 10 of which are STPN-pretraining without the HLN, and 10 of which are further refinements with the HLN attached (cf. Sect. 2.2 and Sect. 2.3). Agents tested on Cityscapes are first trained on CARLA for 10 epochs and refined on Cityscapes for 22 epochs. Agents that are given a goal are trained with a sequence length of 10 timesteps for the first 5 epochs, after which the sequence length is increased to 30. The SPL-goal agents are refined from the weights of goal-free SPL agent that

Table 1. *Left:* Evaluation of pedestrian motion generation with 17s rollouts on the CARLA test set. The SPL (goal-free) agent is compared to the behaviour cloning (BC), constant velocity (CV) heuristics, as well as to to S-GAN [19], ST-GAT [22] and S-STG(CNN) [23]. The average of five different starting scenarios is shown (on pedestrian, random, close to a car, near a pedestrian, or on pavement). Our SPL agent collides with objects and people (f_o) and cars (f_v) less frequently than any other method, while travelling (d) only slightly shorter than ST-GAT. *Right:* Our SPL-goal agent outperforms or matches the goal direction (GD) heuristic and CADRL in success rate (f_s), while colliding much less (f_v, f_o).

	SPL	**BC**	**CV**	**S-GAN**	**ST-GAT**	**S-STG**
f_o	**0.02**	0.03	0.13	0.14	0.14	0.02
f_v	**0.07**	0.13	0.16	0.16	0.15	0.12
d	7.0	1.6	3.7	5.1	**7.9**	0.47

	f_o	f_v	f_s
SPL-goal	**0.09**	**0.01**	**0.78**
CADRL	0.24	0.08	0.75
GD	0.14	0.07	**0.78**

Table 2. *Left:* Average displacement error (m) for pedestrian forecasting on CARLA and Waymo. Our SPL agent receives the second lowest forecasting error on both datasets. The ST-GAT outperforms SPL on the CARLA dataset but yields the worst results on the Waymo dataset. On the Waymo dataset our SPL and BC models outperform the others with a large margin. *Right:* Our SPL agent avoids more collisions ($f_o + f_v$), walks further (d) and stays more on pavements (f_p) than ground truth simulated pedestrians (GT) on CARLA. The SPL agent is initialized on the same positions as the simulated pedestrians.

	SPL	**BC**	**S-GAN**	**ST-GAT**	**S-STG**
CARLA	0.11	0.22	0.16	**0.09**	0.12
WAYMO	0.06	**0.03**	0.11	0.13	0.11

	f_o	f_v	d	f_p
SPL	**0.00**	**0.0**	**17.0**	**0.46**
GT	0.08	**0.0**	16.0	0.35

was trained on CARLA, with the addition of a feature indicating the direction to the goal. Each test scene is evaluated for 10 episodes with different spatial and agent history initializations to compute mean metrics.

3.3 Results

Evaluation metrics are adapted from the benchmark suite of CARLA and are:

- f_o, average frequency of collisions with static objects and pedestrians;
- f_v, average frequency of collisions with vehicles;
- d, average Euclidean distance (in meters) between agent's start and end location in episodes;
- f_p, average frequency of the agent being on pavements;
- f_s, success rate in reaching a goal (only applicable for goal reaching agents).

CARLA. Table 1 (left) shows that our SPL agent generates long trajectories and yields significantly fewer collisions than the compared methods. The SPL average trajectory length 7.0 m is 11% less than the furthest travelling ST-GAT of 7.9 m, but the SPL agent collides 53% less with vehicles and 86% less with objects and pedestrians. As shown in Table 2 (right), SPL even outperforms the CARLA-simulated (GT) trajectories in collision avoidance, and learns to stay on the sidewalk more, despite GT being the experts. To show the effect on the loss (1) of training on states outside of the expert trajectories, we compute the average negative log-likelihood loss (NLL) with respect to expert trajectories on the test set for the STPN module of SPL and of the BC baseline, obtaining losses of 0.009 and 0.013, respectively. The lower NLL of STPN indicates that training on states outside the expert trajectories provides more informative features and a model that acts more similar to ground truth data (i.e. expert trajectories). Finally, the SPL agent obtains the second lowest one-step trajectory forecasting error, or average displacement error (ADE), as seen in Table 2 (left).

Table 3. *Left:* The SPL agent has learnt to avoid collisions with cars and pedestrians significantly better than BC, CV, S-GAN, ST-GAT and S-STG(CNN) on the Waymo data. *Right:* SPL-goal outperforms CADRL and GD on all metrics on Cityscapes. SPL-goal can reach goals while avoiding cars even in noisy scenes.

	SPL	**BC**	**CV**	**S-GAN**	**ST-GAT**	**S-STG**
f_o	**0.07**	0.16	0.22	0.60	0.71	0.15
f_v	**0.03**	0.06	0.10	0.26	0.12	0.07
d	1.4	**4.0**	1.2	2.9	2.5	0.34

	f_o	f_v	f_s
SPL-goal	**0.23**	**0.03**	**0.71**
CADRL	0.28	0.10	0.70
GD	0.28	0.09	0.70

To the right in Table 1 the SPL-goal agent is compared to CADRL and to the goal direction (GD) heuristic when given a goal at a distance of 6 m. Our SPL-goal agent achieves a slightly higher success rate (f_s) than CADRL while being on par with GD. Moreover, SPL-goal is significantly better at avoiding collisions with cars, people and obstacles than the compared methods. In Fig. 3, the SPL-goal agent can be seen safely crossing streets to reach its goals.

Cityscapes. The 3d reconstructions of moving objects in the Cityscapes data can be noisy due to errors in depth estimation in frame-by-frame reconstruction, as well as noise in bounding boxes and semantic segmentation. Therefore the goal reaching task is harder in Cityscapes than in CARLA. Agents are initialized on pavement, near cars or randomly. The SPL-goal agent outperforms the GD heuristic and CADRL in collision avoidance as seen in Table 3 (right). Sample trajectories of our agent can be seen in Fig. 5.

Waymo. In Table 3 (left), our SPL agent, BC, CV, S-GAN, ST-GAT and ST-GCNN are evaluated on 4 second trajectories. The SPL agent is significantly better at collision avoidance than any other model that is only trained on expert pedestrian trajectories. It should be noted that the collision-aware SPL agent travels slower than BC to avoid collisions, which results in shorter trajectories on average. However SPL's trajectories are three times longer than S-STG(CNN) with half of the collisions. The SPL model has the second lowest ADE after BC (which shares SPL's architecture) on the Waymo dataset as seen in Table 2 (left). The SPL model is the only model to perform well on trajectory forecasting on both simulated and real data, while outperforming all models in collision avoidance. Qualitative examples of the SPL agent (without goals) are shown in Fig. 1, Fig. 6 and frame-by frame car avoidance in Fig. 4.

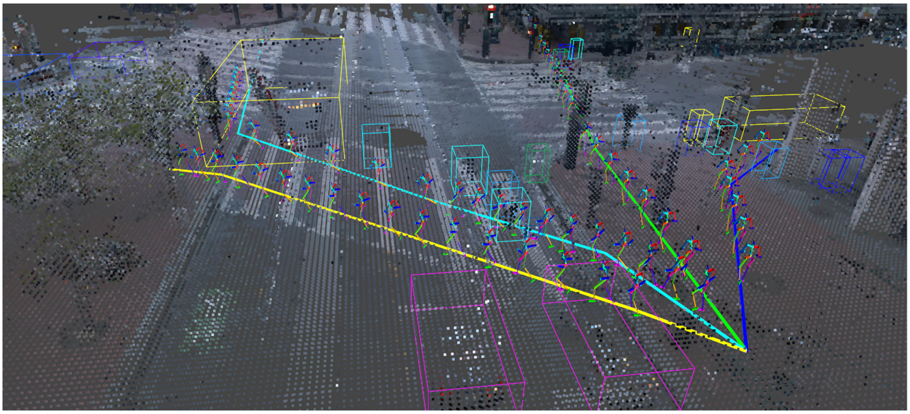


Fig. 6. SPL agent trajectories on the Waymo dataset, showing the pedestrian taking a number of different paths depending on how the agent history is initialized (cf. Sect. 2.2). Cars and other pedestrians are indicated with 3d bounding boxes.

4 Conclusions

We have introduced a novel hierarchical 3d pedestrian locomotion generation model, based on explicit 3d semantic representations of the scene and 3d pedestrian locomotion model. By training the generator with a unified reward and likelihood maximization objective, the model learns to forecast well on both real and simulated data, while outperforming even expert trajectories in collision avoidance. More generally, our formulation can be used to adapt or refine any maximum likelihood-based trajectory forecasting method to simultaneously handle collision avoidance and forecasting. Our formulation also enables the use of articulated human models to enforce human dynamics on the trajectory forecasting model. Finally, the proposed pedestrian motion generator can also be refined to plausibly navigate among other pedestrians and traffic to specific goals. Future work includes studying finer grained agent-scene interactions, for example modelling traffic signs, crossroads, and other relevant objects in urban scenes.

Acknowledgments. This work was supported by the European Research Council Consolidator grant SEED, CNCS-UEFISCDI PN-III-P4-ID-PCE-2016-0535 and PCCF-2016-0180, the EU Horizon 2020 Grant DE-ENIGMA, and the Swedish Foundation for Strategic Research (SSF) Smart Systems Program.

References

1. Chang, M.F., et al.: Argoverse: 3d tracking and forecasting with rich maps. In: CVPR (2019)
2. Sun, P., et al.: Scalability in perception for autonomous driving: Waymo open dataset (2019)
3. Caesar, H., et al.: nuScenes: a multimodal dataset for autonomous driving. In: CVPR (2020)
4. Behley, J., et al.: SemanticKITTI: a dataset for semantic scene understanding of lidar sequences. In: ICCV (2019)
5. Huang, X., et al.: The apolloscape dataset for autonomous driving. In: CVPR Workshops (2018)
6. Kesten, R., et al.: Lyft level 5 AV dataset 2019, vol. 2, p. 5 (2019). https.level5.lyft.com/dataset
7. Mangalam, K., Adeli, E., Lee, K.H., Gaidon, A., Niebles, J.C.: Disentangling human dynamics for pedestrian locomotion forecasting with noisy supervision. In: The IEEE Winter Conference on Applications of Computer Vision, pp. 2784–2793 (2020)
8. Mínguez, R.Q., Alonso, I.P., Fernández-Llorca, D., Sotelo, M.Á.: Pedestrian path, pose, and intention prediction through Gaussian process dynamical models and pedestrian activity recognition. IEEE Trans. Intell. Transp. Syst. **20**, 1803–1814 (2018)
9. Rasouli, A., Kotseruba, I., Tsotsos, J.K.: Pedestrian action anticipation using contextual feature fusion in stacked RNNs. arXiv preprint arXiv:2005.06582 (2020)

10. Bogo, F., Kanazawa, A., Lassner, C., Gehler, P., Romero, J., Black, M.J.: Keep it SMPL: automatic estimation of 3D human pose and shape from a single image. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9909, pp. 561–578. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46454-1_34
11. Zanfir, M., Oneata, E., Popa, A.I., Zanfir, A., Sminchisescu, C.: Human synthesis and scene compositing. In: AAAI, pp. 12749–12756 (2020)
12. Wang, M., et al.: Example-guided style-consistent image synthesis from semantic labeling. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
13. Cheng, S., et al.: Improving 3d object detection through progressive population based augmentation. arXiv preprint arXiv:2004.00831 (2020)
14. Ho, J., Ermon, S.: Generative adversarial imitation learning. In: NIPS (2016)
15. Rhinehart, N., Kitani, K.M., Vernaza, P.: R2p2: a reparameterized pushforward policy for diverse, precise generative path forecasting. In: ECCV (2018)
16. Li, Y.: Which way are you going? Imitative decision learning for path forecasting in dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
17. van der Heiden, T., Nagaraja, N.S., Weiss, C., Gavves, E.: SafeCritic: collision-aware trajectory prediction. In: British Machine Vision Conference Workshop (2019)
18. Zou, H., Su, H., Song, S., Zhu, J.: Understanding human behaviors in crowds by imitating the decision-making process. ArXiv abs/1801.08391 (2018)
19. Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A.: Social GAN: socially acceptable trajectories with generative adversarial networks. In: CVPR (2018)
20. Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I., Rezatofighi, H., Savarese, S.: Social-BiGAT: multimodal trajectory forecasting using bicycle-GAN and graph attention networks. In: NeurIPS (2019)
21. Zhang, L., She, Q., Guo, P.: Stochastic trajectory prediction with social graph network. CoRR abs/1907.10233 (2019)
22. Huang, Y., Bi, H., Li, Z., Mao, T., Wang, Z.: STGAT: modeling spatial-temporal interactions for human trajectory prediction. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
23. Mohamed, A., Qian, K., Elhoseiny, M., Claudel, C.: Social-STGCNN: a social spatio-temporal graph convolutional neural network for human trajectory prediction. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
24. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Li, F., Savarese, S.: Social LSTM: human trajectory prediction in crowded spaces. In: CVPR (2016)
25. Lee, N., Choi, W., Vernaza, P., Choy, C.B., Torr, P.H., Chandraker, M.: Desire: distant future prediction in dynamic scenes with interacting agents. In: CVPR (2017)
26. Luo, W., Yang, B., Urtasun, R.: Fast and furious: real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In: CVPR (2018)
27. Zhao, T., et al.: Multi-agent tensor fusion for contextual trajectory prediction. In: CVPR (2019)
28. Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, H., Savarese, S.: SoPhie: an attentive GAN for predicting paths compliant to social and physical constraints. In: CVPR (2019)
29. Malla, S., Dariush, B., Choi, C.: Titan: future forecast using action priors. In: CVPR (2020)

30. Tanke, J., Weber, A., Gall, J.: Human motion anticipation with symbolic label. CoRR abs/1912.06079 (2019)
31. Liang, J., Jiang, L., Niebles, J.C., Hauptmann, A.G., Fei-Fei, L.: Peeking into the future: predicting future person activities and locations in videos. In: CVPR (2019)
32. Liang, J., Jiang, L., Murphy, K., Yu, T., Hauptmann, A.: The garden of forking paths: towards multi-future trajectory prediction. In: CVPR (2020)
33. Liang, J., Jiang, L., Hauptmann, A.: SimAug: learning robust representations from 3d simulation for pedestrian trajectory prediction in unseen cameras. arXiv preprint arXiv:2004.02022 (2020)
34. Makansi, O., Cicek, O., Buchicchio, K., Brox, T.: Multimodal future localization and emergence prediction for objects in egocentric view with a reachability prior. In: CVPR (2020)
35. Zhang, Y., Hassan, M., Neumann, H., Black, M.J., Tang, S.: Generating 3d people in scenes without people. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6194–6204 (2020)
36. Hong, S., Yan, X., Huang, T.S., Lee, H.: Learning hierarchical semantic image manipulation through structured representations. In: Advances in Neural Information Processing Systems, pp. 2708–2718 (2018)
37. Chien, J.T., Chou, C.J., Chen, D.J., Chen, H.T.: Detecting nonexistent pedestrians. In: CVPR (2017)
38. Li, X., Liu, S., Kim, K., Wang, X., Yang, M.H., Kautz, J.: Putting humans in a scene: learning affordance in 3d indoor environments. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 12368–12376 (2019)
39. Lee, D., Pfister, T., Yang, M.H.: Inserting videos into videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 10061–10070 (2019)
40. Wang, B., Adeli, E., Chiu, H.K., Huang, D.A., Niebles, J.C.: Imitation learning for human pose prediction. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 7123–7132 (2019)
41. Wei, M., Miaomiao, L., Mathieu, S., Hongdong, L.: Learning trajectory dependencies for human motion prediction. In: ICCV (2019)
42. Du, X., Vasudevan, R., Johnson-Roberson, M.: Bio-LSTM: a biomechanically inspired recurrent neural network for 3-d pedestrian pose and gait prediction. IEEE Robot. Autom. Lett. **4**, 1501–1508 (2019)
43. Cao, Z., Gao, H., Mangalam, K., Cai, Q.-Z., Vo, M., Malik, J.: Long-term human motion prediction with scene context. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12346, pp. 387–404. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_23
44. Adeli, V., Adeli, E., Reid, I., Niebles, J.C., Rezatofighi, H.: Socially and contextually aware human motion and pose forecasting. IEEE Robot. Autom. Lett. **5**, 6033–6040 (2020)
45. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
46. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn. **8**, 229–256 (1992)
47. Hodgins, J.: CMU graphics lab motion capture database (2015)
48. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: Human3. 6m: large scale datasets and predictive methods for 3d human sensing in natural environments. IEEE Trans. Pattern Anal. Mach. Intell. **36**, 1325–1339 (2013)
49. Joo, H., et al.: Panoptic studio: a massively multiview system for social motion capture. In: ICCV (2015)

50. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
51. Holden, D., Komura, T., Saito, J.: Phase-functioned neural networks for character control. ACM Trans. Graph. **36**, 42:1–42:13 (2017)
52. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289 (2015)
53. Abadi, M., et al.: Tensorflow: a system for large-scale machine learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2–4, 2016 (2016)
54. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
55. Schöller, C., Aravantinos, V., Lay, F., Knoll, A.: What the constant velocity model can teach us about pedestrian motion prediction. IEEE Robot. Autom. Lett. **5**, 1696–1703 (2020)
56. Chandra, S., Bharti, A.K.: Speed distribution curves for pedestrians during walking and crossing. Procedia-Soc. Behav. Sci. **104**, 660–667 (2013)
57. Everett, M., Chen, Y.F., How, J.P.: Motion planning among dynamic, decision-making agents with deep reinforcement learning. In: IROS (2018)
58. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: an open urban driving simulator. In: CoRL (2017)
59. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
60. Nilsson, D., Sminchisescu, C.: Semantic video segmentation by gated recurrent flow propagation. In: CVPR (2018)
61. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR (2016)
62. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: CVPR (2018)
63. Zhou, B., et al.: Semantic understanding of scenes through the ADE20K dataset. IJCV **127**, 302–321 (2018)

Semantic Synthesis of Pedestrian Locomotion Supplementary Material

Maria Priisalu[1], Ciprian Paduraru[2,3], Aleksis Pirinen[1], and Cristian Sminchisescu[1,3,4]

[1] Department of Mathematics, Faculty of Engineering, Lund University
[2] The Research Institute of the University of Bucharest (ICUB), Romania
[3] Institute of Mathematics of the Romanian Academy
[4] Google Research
{maria.priisalu,aleksis.pirinen, cristian.sminchisescu}@math.lth.se
ciprian.paduraru@fmi.unibuc.ro

1 Supplementary Videos and Further Visualizations

In the file pedestrian_synthesis_in_waymo.mp4, example trajectories generated by our goal-free SPL agent on the Waymo test set are shown. The agent is visualized as a skeleton, which is generated by the human locomotion network (HLN), cf. §2.3 in the main paper. Also, 3d bounding boxes of cars and other pedestrians are shown. The agent can be seen to wait for a bus before crossing and starting to run when cars close to it start moving. The agent can also be seen crossing the street while walking behind a car and crossing a busy intersection. When the ground truth trajectories end, the bounding boxes of cars and pedestrians stand still, but the agent continues to produce plausible trajectories. In Fig. 1 a small qualitative example of the agent cutting corners and leaving the crosswalk can be seen. Such plausible but perhaps imperfect behaviour from the point of view of traffic laws would need to be explicitly modelled to be achieved by classical planners (based on methods such as visibility graphs or spatial navigation meshes).

In the file pedestrian_synthesis_in_cityscapes.mp4, example trajectories generated by our SPL-goal agent on the Cityscapes test set are shown. Two observed pedestrians visualized as the blue and yellow bounding box can be seen walking along the pavement. A car is approaching marked by the red bounding box. It is unclear where the pedestrians are heading, as they may decide to cross the street on the crosswalk, continue along the curb or continue forward. Our agent can be used to rollout the motion of the pedestrian in yellow in all of these cases. The agent is initialized with the past movements of the pedestrian in yellow after which it generates movement in multiple plausible directions, shown as the moving skeleton (which again is produced by the HLN module). In the second scene we show the temporal visualization of Fig. 5 from the main paper.

Frame by frame visualizations from the CARLA test set in Fig. 3 shows that the HLN produces smooth poses even when accelerating from standing to running or when running away from cars.

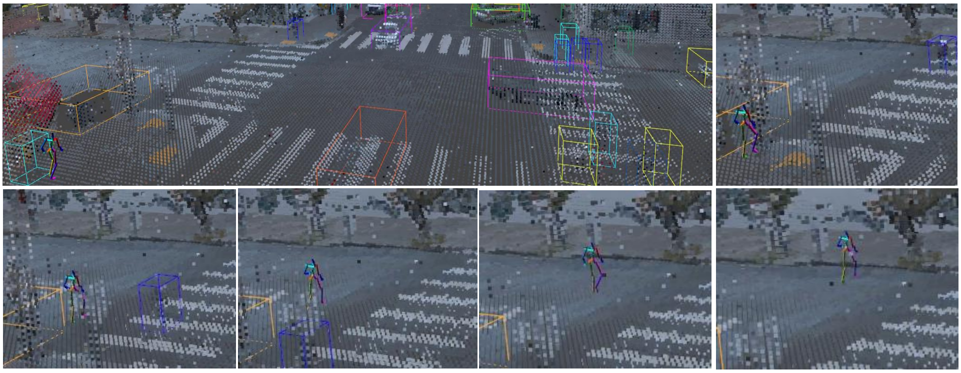


Fig. 1: A subsampled pose sequence in Waymo showing the SPL agent cutting corners and choosing to avoid the crosswalk. This kind of imperfect but plausible traffic behaviour would not be modelled by classical planning methods.To the *top left:* it can be seen that no cars are approaching the pedestrian. The pedestrian continues to the left once on the pavement.

Further visualizations of the SPL agent on the CARLA test set can be seen in Fig. 2. In these visualization we show the SPL agent trained only with RL, i.e. without any alternation between imitation learning and RL (cf. Algorithm 1).

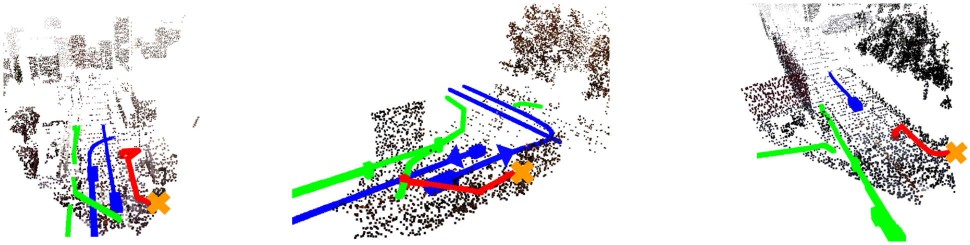


Fig. 2: Additional 1-minute trajectories of our goal-free SPL agent on the CARLA test. Note that in these examples we show an SPL agent that was trained without any objective that encourages it to track another pedestrian – it was trained only to walk safely and plausibly in the environment based on the reward signal in §2.4 of the main paper. Car and person trajectories are shown in blue and green, respectively. Red indicates the agent's trajectory, with terminal position indicated by an orange cross. *Left:* Agent walking along a pavement. The pavement also continues to the right close to the initial location, and the agent initially walks a few steps in that direction. It then then turns to move towards the camera, which indicates the inherent multi-modality of plausible paths to take. *Middle:* When initialized on the road on a trajectory of a crossing pedestrian, the agent moves away from the road onto the pavement parallel to to the road. Blue arrows indicate the directions of the nearby cars. Note that the agent is not hit by any of the cars (the bottom car has moved away before the agent reaches that trajectory). The agent produces a plausible but different trajectory from the one it was initialized on. *Right:* Agent moving to the sidewalk when initialized on the road. The agent follows the curvature of the pavement in the end of its trajectory.

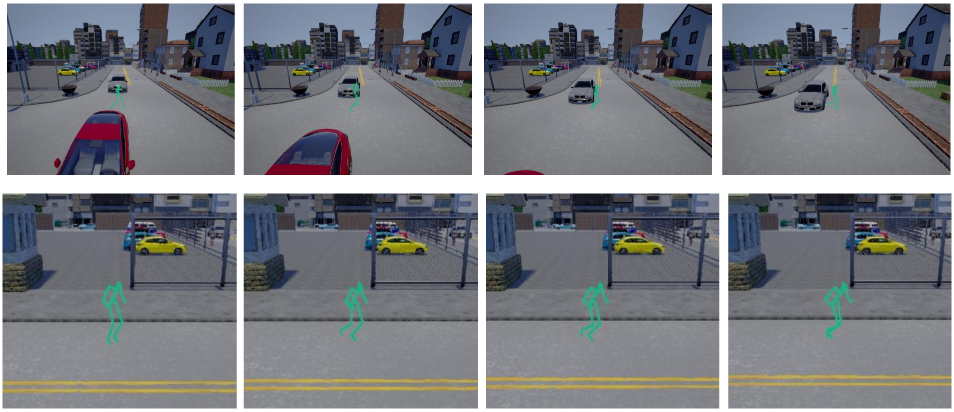


Fig. 3: *Top:* A subsampled (every 4th frame) pose sequence in the CARLA test set showing the agent successfully running to avoid two cars. *Bottom:* The agent starts running from a standing initial position to leave the road. The movements are temporally smooth even when the agent accelerates from standing.

2 Details of our Model Architecture

In Fig. 5 we show the network architecture of the proposed model. The semantic and RGB top-view projections as well as the dynamic occupancy map (constructed as shown in Fig. 4) are processed by the convolutional features extractor, which consists of two convolutional layers with ReLU activation and followed by max pooling. The agent's past trajectories (i.e. the agent history) are processed by an LSTM. Finally, two independent fully connected layers process the HLN's locomotion feature $\boldsymbol{h}_{t-1}$, the agent's displacement $\boldsymbol{d}_t$ to the closest vehicle, the encoded agent history $\boldsymbol{f}_t$, and the convolutional features to produce two feature vectors $\boldsymbol{f}_u$ and $\boldsymbol{f}_\mu$. The multinomial distribution over unit directions is given by $\pi_\Theta(\boldsymbol{U}_t|\boldsymbol{s}_t) = \mathrm{softmax}(\boldsymbol{f}_u + \boldsymbol{u}_{t-1})$. The addition of $\boldsymbol{u}_{t-1}$ acts as a prior to make the agent move in the same general direction unless motivated by the current state $\boldsymbol{s}_t$ to change direction. The mean speed is given by $\mu_t = \sigma(\boldsymbol{f}_u)$, where σ is the sigmoid function. Finally, the agent's velocity $\boldsymbol{v}_t$ for the current timestep is given by $\boldsymbol{v}_t = |\boldsymbol{v}_t|\boldsymbol{u}_t$.

3 Semantic 3d Reconstruction of Cityscapes

The 3d reconstruction of the environment E_t is performed in two parts – one global reconstruction for static objects S and a frame-by-frame stereo reconstruction for moving objects in D_t. Semantic segmentation is used to mask out people, cars and bikes, when performing sparse 3d reconstruction. Dynamic objects are represented by cuboids in D_t, which are found by performing 2d object detection followed by triangulation.

A semantic segmentation network is used to estimate the semantic mask of each frame. The points in the found 2d semantic masks are triangulated to find the corresponding 3d points in the world. A 3d point $\boldsymbol{p_i}$ which is visible in

Construction of the Relative Dynamic Occupancy Map

Current agent position

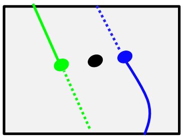
$\boldsymbol{x}_t = (0, 0)$

Car trajectory with constant velocity estimates of future positions

$\ldots \boldsymbol{x}_{t-1}, \boldsymbol{x}_t, \boldsymbol{x}_t + (\boldsymbol{x}_t - \boldsymbol{x}_{t-1}) \ldots = \ldots (2, 0), (1, 1), (0, 2) \ldots$

Pedestrian trajectory with constant velocity estimates of future positions

$\ldots \boldsymbol{x}_{t-1}, \boldsymbol{x}_t, \boldsymbol{x}_t + (\boldsymbol{x}_t - \boldsymbol{x}_{t-1}) \ldots = \ldots (-2, 0), (-1, -1), (0, -2) \ldots$



Occupancy Timestamps of Agent's Neighborhood

		t+1		
			t	
t-1		t		t-1
	t			
		t+1		

-

Current Agent Timestamp x Dynamic Occupancy

		t		
			t	
t		t		t
	t			
		t		

= ➡

Relative Dynamic Occupancy Map

		1		
			0	
-1		0		-1
	0			
		+1		

Note: Car and Pedestrian positions at timestep **t** are input as separate semantic channels

Fig. 4: Construction of the dynamic occupancy map. The pedestrian and car trajectories are extended by constant velocity estimates at each timestep, as indicated by the dashed lines in the top-right rectangle. Timestamps of future and past occupancy in the agent's neighborhood are then mapped into an occupancy map, centered around the agent. Finally, the current timestamp t is subtracted to provide a relative dynamic occupancy map centered around 0. Note that the pedestrians and cars present are represented in separate channels in the semantic 3d reconstruction input E.

frames $i, .., j$ has the 2d projections $\boldsymbol{u_i}, .., \boldsymbol{u_j}$. Each 2d point $\boldsymbol{u_i}$ corresponds to a semantic label estimate l_i. The mode of $l_i, ..., l_j$ is assigned as the label of p_i. Alternatively, a 3d semantic segmentation method could be used. The resulting pointclouds are regularized by a voxel grid G. In the regularization, the color of a voxel is determined by the mode color of the points in the voxel. Semantic class is determined in the same fashion. The voxelized static environment is filtered with a median filter with size $(1, 4, 4)$. Finally, the height of the ground plane is estimated as the mode of points in the 3d semantic pointcloud with the semantic labels for ground, road, sidewalk, parking, terrain and rail tracks.

4 Training Procedure for the SPL and SPL-goal Agents

The SPL and SPL-goal agents are trained to simultaneously minimize the pedestrian trajectory forecasting error by maximizing the model likelihood on the pedestrian trajectories in the data, and to maximize the respective reward functions. In §2 of the main paper it is shown that the two objectives can be rewritten as a single one, with two sampling techniques (sampling agent locations and moves from the expert trajectories, or from the current policy). Given expert trajectories $(\boldsymbol{x}_0^i, \ldots \boldsymbol{x}_T^i) \in \mathcal{D}$, which can also be described by the state-velocity

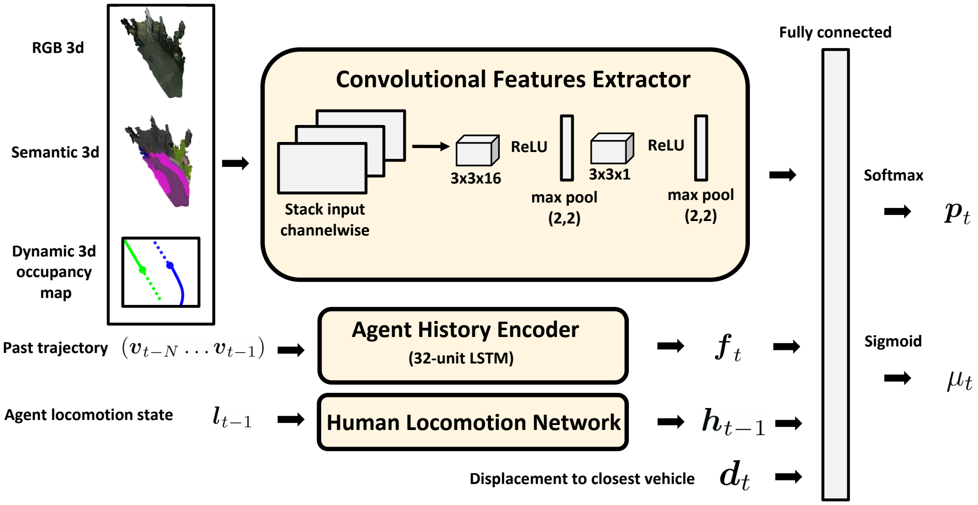


Fig. 5: Semantic trajectory policy network (STPN) architecture. A top-view of an agent-centered local crop of the 3d pointcloud with RGB and semantic labels, together with a top-view of the dynamic occupancy map, is processed by the convolutional features extractor. The N last agent velocities $\boldsymbol{v}_{t-N}, \dots, \boldsymbol{v}_{t-1}$ are fed to a 32-unit LSTM, whose hidden state is denoted $\boldsymbol{f}_t$. The human locomotion network (HLN) extracts from the previous locomotion state $\boldsymbol{l}_{t-1}$ the locomotion feature $\boldsymbol{h}_{t-1}$ (recall that the STPN is executed *before* the HLN, and thus the STPN only has access to the *previous* hidden state $\boldsymbol{h}_{t-1}$ of the HLN). Finally, the convolutional features, agent history feature $\boldsymbol{f}_t$, locomotion feature $\boldsymbol{h}_{t-1}$ and displacement vector $\boldsymbol{d}_t$ to the closest vehicle are concatenated and fed to two separate fully connected layers. The first layer feeds into a softmax to produce a distribution $\boldsymbol{p}_t$ over the unit movement directions (unit velocities), and the second layer feeds into a sigmoid to produce the mean speed μ_t for a normal distribution from which a speed $|\boldsymbol{v}_t|$ is sampled. The agent's velocity $\boldsymbol{v}_t$ for the current timestep is finally given by $\boldsymbol{v}_t = |\boldsymbol{v}_t|\boldsymbol{u}_t$.

pairs $(\boldsymbol{s}_0^i, \boldsymbol{v}_0^i) \dots (\boldsymbol{s}_T^i, \boldsymbol{v}_T^i)$, and given the set of agent initializations[5] $\mathcal{I}$, the full optimization can be performed by following Algorithm 1 that samples gradients from the expert pedestrian (initialized from $(\boldsymbol{x}, v) \in \mathcal{D}$) and the policy trajectories (initialized from $\boldsymbol{x}_0 \in \mathcal{I}$, and thereafter following $\boldsymbol{v}_t \sim \pi_\Theta(\boldsymbol{v}|\boldsymbol{s}_t)$ where $\boldsymbol{x}_{t+1}$ is given by HLN step towards $\boldsymbol{x}_t + \boldsymbol{v}_t$), respectively.

In all experiments the semantic trajectory policy network (STPN, cf §2.2 in the main paper) is trained first without the HLN, cf. Table 1. The SPL agent is then refined with the HLN executing the steps provided by the STPN, as visualized in Fig. 3 of the main paper. During this training step the HLN weights are kept frozen (please refer to §2.5 for training procedure and evaluation). The CARLA goal driven agent SPL-goal-CARLA (Table 1 in main) is initialized from

[5] $\mathcal{I}$ contains initializations near cars, near pedestrians, on pavement and at random.

Algorithm 1 Unified Batch Policy Gradient and Maximum Likelihood Estimation

```
for $N$ epochs do
  for pedestrian $i$ in $\mathcal{D}$ do
    Initialize batch gradient direction $\boldsymbol{g} = 0$
    Initialize agent on the $i$th pedestrian's initial state $\boldsymbol{s}_0 = \boldsymbol{s}_0^i$
    for $t$ in $[0, T]$ do
      Take expert action $\boldsymbol{v} = \boldsymbol{v}_t^i$
      Evaluate gradient $\boldsymbol{g} = \boldsymbol{g} + \partial_\Theta R(\boldsymbol{s}, \boldsymbol{v}) \log(\pi_\Theta(\boldsymbol{s}|\boldsymbol{v}))$
    end for
    for Initialization $j$ in $\mathcal{I}$ do
      Initialize agent on the $j$th initial state $\boldsymbol{s} = \boldsymbol{s}_0^j$
      for $t$ in $[0, T]$ do
        Sample action $\boldsymbol{v}_t^j \sim \pi_\Theta(\boldsymbol{v}|\boldsymbol{s})$
        Take action $\boldsymbol{v} = \boldsymbol{v}_t^j$
        Evaluate gradient $\boldsymbol{g} = \boldsymbol{g} + \partial_\Theta R(\boldsymbol{s}, \boldsymbol{v}) \log(\pi_\Theta(\boldsymbol{v}|\boldsymbol{s}))$
      end for
    end for
    Update $\Theta$ with gradient step in direction $\boldsymbol{g}$.
  end for
end for
```

the pretrained weights of SPL-CARLA noted here for clarity as STPN-CARLA to indicate that no training with the HLN is performed. Since the Cityscapes dataset does not contain tracking and therefore trajectories, the STPN-Cityscapes is initialized from STPN-CARLA, and is trained on the Cityscapes dataset for 14 epochs. The STPN-Cityscapes is the initialization of the SPL-goal-Cityscapes presented in Table 3. Finally the SPL Waymo agent in Table 3 is initialized from the weights of STPN Waymo. The initialization and training epochs are gathered in Table 1. The agent history is initialized randomly when the agent is not initialized on top of a pedestrian in the data. This provides a random initial direction and promotes exploration in early stages of learning.

Table 1: Number of epochs trained for different models presented. The number of epochs of pretraining shows the number of epochs STPL is trained without the HLN, and No of epochs is the number of epochs the two modules are trained together.

Model	Dataset	Weight initialization	No. of epochs pretraining	No. of epochs training
SPL	CARLA	Random	10	8
SPL-goal	CARLA	(STPN-CARLA)	21	14
SPL	Waymo	Random	1	1
STPN	Cityscapes	(STPN-CARLA)	14	-
SPL-goal	Cityscapes	(STPN-Cityscapes)	22	17

5 Ablation Study of Reward Function

Table 2: Ablations of the STPN with an average over all of the presented initializations. The ablation results are on the CARLA validation set with a maximal episode length of 5.8s. f_d is the frequency on locations occupied by pedestrians, f_{pav} is the frequency on pavement. The full model balances between collisions, travelling far and implicitly learns to stay on pavement without explicitly staying on pedestrian trajectories.

Model	$f_{o,p}$	f_{car}	d	f_d	f_{pav}
$R_{dist} + R_{coll}$	0.01	0	4.7	0.14	0.61
R_{ped}	0.55	0	9.6	0.24	0.48
$R_{coll} + R_d$	0.03	0.1	13	0.45	0.79
$R_{coll} + R_k$	0.02	0	16	0.04	0.45
$R_{coll} + R_{ped}, \lambda_p = \lambda_v = \lambda_s$	0.04	0	15	0.31	0.38
$R_{coll} + R_{ped}$	0.04	0	8.5	0.32	0.77

To show the need for the different reward components we present a short ablation study in table Table 2 on the CARLA test set. The different models are the STPLN agent trained with the listed reward weights. The R_{ped} is trained with a positive reward for being on pedestrian trajectory $\lambda_d = 0.01$ and $\lambda_k = 0.01$. This leads to an agent that is oblivious to collisions and collides with a frequency of 0.55. As a baseline, we present an agent trained without the pedestrian reward components and instead a positive reward $R_{dist} + R_{coll}$ for moving further from the start location and a negative reward for collisions ($\lambda_{dist} = 0.001$). This agent $R_{dist} + R_{coll}$ visits the pedestrian occupancy map seldom $\boldsymbol{f}_d = 0.14$. Further only including one of the pedestrian reward components $R_{coll} + R_d$,$R_{coll} + R_k$ leads to an agent that ignores cars when only rewarded for staying on expert trajectories, and to an agent that seldom visits the pedestrian occupancy map when only rewarded for staying on the pedestrian heatmap. Finally the the model trained with the full reward $R_{coll} + R_{ped}$ balances between all of these metrics. To show the effect of the different collision components of the reward, we set the penalty for colliding with cars, pedestrians and obstacles equal to 1 $R_{coll} + R_{ped}, \lambda_p = \lambda_v = \lambda_s$ and observe that this leads to an agent that is on pavementf_{pav} half of the times compared to the proposed model $R_{coll} + R_{ped}$.

Paper II

Generating Scenarios with Diverse Pedestrian Behaviors for Autonomous Vehicle Testing

Maria Priisalu[1], **Aleksis Pirinen**[2*], **Ciprian Paduraru**[3,4], **and Cristian Sminchisescu**[1,5]

[1]Lund University, [2]RISE Research Institutes of Sweden, [3]University of Bucharest, [4]Institute of Mathematics of the Romanian Academy, [5]Google Research

Abstract: There exist several datasets for developing self-driving car methodologies. Manually collected datasets impose inherent limitations on the variability of test cases and it is particularly difficult to acquire challenging scenarios, e.g. ones involving collisions with pedestrians. A way to alleviate this is to consider automatic generation of safety-critical scenarios for autonomous vehicle (AV) testing. Existing approaches for scenario generation use heuristic pedestrian behavior models. We instead propose a framework that can use state-of-the-art pedestrian motion models, which is achieved by reformulating the problem as learning where to place pedestrians such that the induced scenarios are collision prone for a given AV. Our pedestrian initial location model can be used in conjunction with any goal driven pedestrian model which makes it possible to challenge an AV with a wide range of pedestrian behaviors – this ensures that the AV can avoid collisions with any pedestrian it encounters. We show that it is possible to learn a collision seeking scenario generation model when both the pedestrian and AV are collision avoiding. The initial location model is conditioned on scene semantics and occlusions to ensure semantic and visual plausibility, which increases the realism of generated scenarios. Our model can be used to test any AV model given sufficient constraints.

Keywords: Autonomous Vehicles, AV Testing, Reinforcement Learning

1 Introduction

Research on autonomous vehicle (AV) models has gained momentum in recent years [1]. There exist both end-to-end AV models which make decisions directly based on visual sensor outputs [1–6], and hierarchical models which require intermediate processing (such as pedestrian detection) of sensor outputs for decision making [7, 8]. To ensure traffic safety, e.g to avoid fatal collisions [9], there is a need to evaluate the various AV models in safety-critical situations. In this paper we consider safety testing of the full pipeline of perceptive AV models – from sensor inputs (e.g. images) to steering. There exist several datasets [10–17] for developing and evaluating AV models, but manually collected data is typically gathered from traffic scenarios that seldom exhibit collision and near-collision scenarios. This shortcoming has lead to recent developments of safety-critical test case generation methods [18–32] for AV models. These existing approaches resort to simulated pedestrians which are not representative of the rich and varied behavior of real pedestrians [33] – either the pedestrian trajectories are handcrafted, or the pedestrian models are trained to behave in unnatural ways (e.g. pedestrian agents which are adversarially trained to collide with vehicles). Thus these methods may provide insufficient insights on how the AV would act in scenarios involving real pedestrians. At the same time, there exist a large number of state-of-the-art pedestrian behavior models [34–49] which learn, from real traffic scenarios, how pedestrians interact with the world.

Different from [18–32], we reformulate the problem of generating challenging scenarios as one of learning the distribution μ of pedestrian initial locations x_0 which are likely to induce collisions between the pedestrian and the AV, for a given pedestrian behavior model π. This reformulation allows the use of state-of-the-art goal driven pedestrian behavior models π in AV test case generation,

[*] Work partially done while at Lund University.

5th Conference on Robot Learning (CoRL 2021), London, UK.

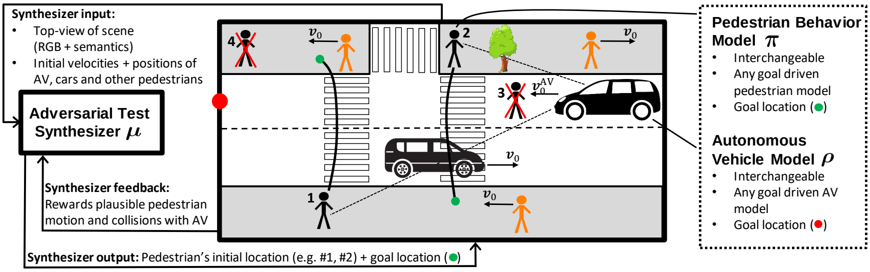

Synthesizer input:
Top-view of scene (RGB + semantics)
Initial velocities + positions of AV, cars and other pedestrians
Adversarial Test Synthesizer μ
Synthesizer feedback:
Rewards plausible pedestrian motion and collisions with AV
Synthesizer output: Pedestrian's initial location (e.g. #1, #2) + goal location (●)
Pedestrian Behavior Model π
Interchangeable
Any goal driven pedestrian model
Goal location (●)
Autonomous Vehicle Model ρ
Interchangeable
Any goal driven AV model
Goal location (●)
1
2
3
4
v_0
v_0^{AV}

Figure 1: Overview of the proposed safety-critical test case generation model for AVs. The *Adversarial Test Synthesizer (ATS)* is trained to position a pedestrian with behavior model π such that the induced scenario is likely to yield a collision with the AV ρ (car to the right). Four example scenarios are shown. Scenarios #3 and #4 are visually implausible, as a pedestrian cannot simply appear from nowhere into the line of sight of an AV. Scenarios #1 and #2 are both plausible and challenging, as the pedestrian is close to the AV and not in the line of sight of the AV due to the occlusions.

which means that the AV can be stress tested in more realistic scenarios compared to prior works. There exist three types of pedestrian behavior models – collision seeking, collision ignorant, and collision avoiding – each of which gives rise to a distinct optimization problem in our framework. We are the first to show empirically that a non-trivial solution exists when the pedestrian model is collision avoiding. Different from previous work [18–32], we explicitly model scene semantics to learn to generate semantically and visually plausible traffic scenarios. Our model can be used to augment existing data by adding simulated safety critical pedestrians to real traffic scenarios.

In real traffic an AV can be expected to encounter pedestrians with a range of different behaviors. Some individuals follow traffic rules and plan their movements based on the surroundings; others are inattentive and take risks. Independently of the pedestrian's overall behavior, an AV should be able to avoid collisions with the pedestrian when it appears from an occluded space. To ensure this, AV test scenarios should cover the true variation of different pedestrian behaviors. Previous works [18–32] either assume that the pedestrian motion can be modelled by a simple constant velocity model, or that the pedestrian motion is adversarial to the AV. In reality however, collisions do not occur only when a pedestrian has a perfectly predictable path (e.g. constant velocity), or when the pedestrian is actively seeking to get hit by the AV (e.g. adversarial pedestrian model). Quite the contrary – most collisions occur because pedestrians are distracted, due to occlusions or noise. To alleviate the previous unrealistic assumptions on pedestrian motion in generative AV testing, we separate the problem of *finding* the pedestrian location distribution μ from the *modelling* of the pedestrian behavior. The main problem is then to find a location distribution μ such that the number of collisions between a black-box AV and a black-box pedestrian is maximal in expectation. In AV testing the proposed approach should be used with as many different pedestrian behavior models as possible, as an AV should be seeking to avoid collisions with all (even collision seeking) pedestrians.

The pedestrian location distribution μ, shown in Fig. 1, is conditioned on scene semantics, distance to the AV, as well as a dynamic occupancy and occlusion map. Occlusions can cause an AV to miss a pedestrian (or vise versa) [50] and can thus cause collisions, therefore affecting the shape of μ. Furthermore, μ is likely to be shaped by the scene semantics (e.g., pedestrians are more likely to reside on sidewalks than on grass) [33, 51, 52]. In previous works, test case generation for AVs has been treated as a reinforcement learning problem [20–24, 32] or as a black-box optimization problem solved by bayesian optimization (BO) [18, 21]. BO [53] cannot be used to learn μ as μ is inherently discontinuous – in realistic scenarios pedestrians can only appear from occluded spaces [51] (cf. Fig. 2). Reinforcement learning (RL) on the other hand does not assume that the policy μ is continuous, and avoids the curse of dimensionality (that occurs in classical control and planning methods) in problems, like ours, with large state spaces with unknown world dynamics [54]. We thus propose the *Adversarial Test Synthesizer (ATS)*, an RL agent which positions pedestrians in a given scene (see Fig. 1). It selects initial locations for the pedestrian according to its policy μ, which is optimized to increase the number of collisions. For the ATS agent, the uncontrollable external dynamics include the scene, the AV, and all other pedestrians and cars. We model μ as a heatmap over the scene, parametrized by a deep convolutional neural network. Our pedestrian initial location model μ allows collision seeking scenario generation with any goal driven pedestrian behavior model and any AV model. This allows for more varied and more realistic testing of the AV. We show that near-collision scenario generation with a collision avoiding pedestrian gives rise to a previously unstudied optimization problem in AV testing. We show that this problem has a solution.

1.1 Related Work

Previous works [21, 23, 24] have studied the generation of full pedestrian trajectories $(x_0, \dots, x_T)$ for AV testing, such that the trajectory is adversarial to the AV ρ. This leads to the pedestrian only behaving in a suicidal manner. This is unnecessarily limiting as typically it is not only the AV which aims to avoid collisions. Ultimately in testing we wish to ensure that the AV can avoid collisions with adversarial as well as collision avoiding pedestrians. In [18] pedestrians are modelled with constant velocity and are initialized from a set of predefined positions. The AV is retrained in a loop with the test case generator. In [21] existing trajectories are adapted to become adversarial. The suggested method requires a varied ground truth dataset and the generated data is dependent on the variability of the existing dataset. Similarly, [25, 26, 28–31] augment existing datasets in a latent or trajectory space, which again requires a large and varied ground truth dataset.

When only testing the vehicle control of a hierarchical AV system, the set of initial locations that cause collisions can be found by the Hamiltonian-Jacobi reachability set [55]. This is not possible in our setup since we consider the full stack of the AV, not only the control problem. Moreover, in our framework the pedestrian is not necessarily adversarial to the AV, and the scene dynamics cannot be described by a differential game. Our proposed approach allows the testing of AV models with pedestrian models that are semantically aware, collision avoiding, goal reaching and articulated. We do not use robust control methods because we utilize explicit pedestrian behavior models.

There are a number of recent studies which explore visual relations in data from the AV's perspective [51, 52, 56]. Makansi et al. [51] learn a visual prior for where pedestrians and other objects can appear from the perspective of a camera mounted on an AV. A similar problem of realistic object placement in LiDAR scenes is studied in [56]. Finally, [52] show that visual cues from an on-board camera can be used to learn walkable areas in a scene. The results of [51, 52, 56] indicate that realistic data contains strong correlations between the scene's semantic structure as well as the the presence and behavior of pedestrians and stationary obstacles. We thus include such semantic cues and occlusions in our proposed pedestrian location distribution model μ, as described in §2.

2 Methodology

In our framework, the ATS μ and the AV model ρ play an indirect constrained minimax game, and no assumptions are made about the pedestrian behavior model π. Thus π can be cooperative with either the AV ρ or ATS μ, or be ignorant with respect to both of these. The problem then becomes a constrained indirect three-agent minimax game with up to two agents per team. The study of the equilibrium [57, 58] is beyond the scope of this paper. However, it is clear that if the AV ρ, pedestrian behavior π and pedestrian location distribution μ are unconstrained, then the minimax problem has a trivial solution. If the pedestrian is always initialized arbitrarily close to the front of the AV (when the AV's initial velocity is forward), then this will always lead to a collision. If the pedestrian and the AV always stand still or always move in opposite directions, then there are never any collisions. To avoid trivial solutions, sufficient constraints are needed.

To illustrate the minimax problem, assume that the loss functions for the AV ρ, the pedestrian π, and the ATS μ are respectively given by a sum of the expectation of the number of collisions and other loss components. Let the number of collisions between a given AV and pedestrian be measured by an indicator function I that is 1 if a collision occurs and 0 otherwise. The AV model ρ and the pedestrian location distribution model μ are learnt by minimizing the loss functions J_ρ and J_μ respectively,

$$\min_{\rho} J_\rho = \min_{\rho} \left(\mathbb{E}_{\mu,\rho,\pi}[I] + f_\rho(\rho)\right) \text{ s.t. } \rho \in B_\rho \tag{1}$$

$$\min_{\mu} J_\mu = \max_{\mu} \left(\mathbb{E}_{\mu,\rho,\pi}[I] - f_\mu(\mu)\right) \text{ s.t. } \mu \in B_\mu, \tag{2}$$

where f_ρ and f_μ are loss components of J_ρ and J_μ, respectively. And B_ρ and B_μ describe the model constraints of ρ, and μ respectively. Equations (1) - (2) express the general optimization problem when the pedestrian behavior π is independent of $\mathbb{E}[I]$ (for example constant velocity π). If π is collision avoiding then (1) - (2) together with the following equation describe the general problem

$$\min_{\pi} J_{\pi} = \min_{\pi} \left(\mathbb{E}_{\mu,\rho,\pi}[I] + f_{\pi}(\mu) \right) \text{ s.t. } \pi \in B_{\pi}, \tag{3}$$

where f_{π} is the loss component of J_{π}, and B_{π} describes the constraints on the model π. If the pedestrian behavior model π is adversarial then (3) will be replaced by $\max_{\pi} \left(\mathbb{E}_{\mu,\rho,\pi}[I] + f_{\pi}(\mu) \right)$ s.t. $\pi \in B_{\pi}$. It is clear that the choice of the behavior policy π changes the optimization problem and affects the solutions of μ and ρ. Depending on the choice of π, the set of applied constraints B_{ρ} and B_{μ} may need to be adjusted to ensure that none of the models converge to a trivial solution. Previous works [18–32] have considered the cases where π is adversarial or a constant velocity model. In our experiments we illustrate that with sufficient constraints on μ, π and ρ, a non-trivial solution exists for (2) when π is collision avoiding. The classical existence conditions of a solution of zero-sum game cannot be applied [57, 59] because the problem at hand is not a zero-sum game, as the pedestrian has loss terms f_{π} that are not present in the AV's loss function J_{ρ}.

2.1 Special Case: Three Reinforcement Learning Agents

We view the problem of learning the pedestrian initial location distribution as a reinforcement learning (RL) problem with three agents: the pedestrian, the AV and the ATS. At timestep $t \in \{0, T-1\}$ the pedestrian and the AV move in the scene by taking actions a_t^{π} and a_t^{ρ}, respectively; we gather these in a joint vector $a_t = (a_t^{\pi}, a_t^{\rho})$. The pedestrian's action is sampled from the pedestrian policy $a_t^{\pi} \sim \pi(.|s_t^{\pi})$ conditioned on its observation s_t^{π} of the scene which includes the AV. Similarly, the AV chooses actions as $a_t^{\rho} \sim \rho(.|s_t^{\rho})$ where s_t^{ρ} is the AV's observation of the scene which includes the pedestrian. The states s_t^{π} and s_t^{ρ} respectively contain the pedestrian's and the AV's final goal location. We join s_t^{π} and s_t^{ρ} as a vector $s_t = (s_t^{\pi}, s_t^{\rho})$. The unknown world model $p(s_{t+1}|s_t, a_t)$ provides the transition probabilities from state s_t to state s_{t+1} when the pedestrian and the AV take the joint action a_t. The pedestrian's and the AV's actions are evaluated by the reward functions $r_{\pi}(s_t, a_t, s_{t+1})$ and $r_{\rho}(s_t, a_t, s_{t+1})$, respectively. The policies π and ρ are trained to maximize the respective expected discounted cumulative future rewards (i.e. the utility) at each state s_t.

We assume that the AV's initial location y_0, initial velocity v_0^{ρ} and final goal location are given. Before the 0th timestep the ATS observes $s^{\mu} = (S, D, OP)$, where S is the top view RGB and semantic images of the scene, with constant velocity predicted dynamic occupancy D of the AV(calculated from y_0 and v_0^{ρ}), external cars and external pedestrians in the scene, and OP is the elementwise product between the occlusion map from the AV's perspective O and μ's prior distribution P . The prior P is a heuristic of μ which assigns high probability to pedestrian initial locations that are close to the AV and that can lead to a collision assuming constant motion v_0^{ρ} of the AV. The ATS agent samples an initial pedestrian location x_0 from the policy $OP\mu(s^{\mu})$, which is the product between OP and the learnable policy μ. To reduce notational clutter we will in §2.1 omit the notation OP from $OP\mu$ and let μ denote the policy of ATS. The pedestrian with an initial location x_0 is given a goal location g^{π} and velocity v_0^{π} such that the pedestrian's path to g^{π} coincides with the AV's assuming both move with constant velocity. After sampling the pedestrian's initial location x_0 we simulate the pedestrian at the location x_0 with velocity v_0^{π}. Next we can simulate the pedestrian's and the AV's observation of the world s_0^{π}, s_0^{ρ} at $t = 0$. Our aim is to find the the initial distribution μ – i.e. the policy of the ATS agent – which leads to the highest utility for the reward function $r_{\mu}(s_t, a_t, s_{t+1})$, where r_{μ} attains its highest value when the AV and pedestrian collide.

In our experiments the learnable ρ, π, μ are modelled by policy gradient models and share the loss

$$J = \mathbb{E}_{x_0 \sim \mu_{\Theta}(.|s^{\mu}), s^{\mu} \sim q, a_t^{\pi} \sim \pi, a_t^{\rho} \sim \rho, s_t \sim p(.|s_t, a_t)} \left[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t^{\pi}, a_t^{\rho}, s_{t+1}) \right], \tag{4}$$

where $r = (r_{\mu}, r_{\pi}, r_{\rho})$. The loss functions' dependence on I is expressed in the different reward functions. To simplify notations let the state-action history $\tau = (a_0, s_1, ..., a_{T-1}, s_T)$, and the discounted cumulative reward $R = \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t, s_{t+1})$. We can express (4) as $\mathbb{E}[R] = \int_{s^{\mu}} \int_{x_0} \int_{\tau} R(x_0, \tau) q(s^{\mu}) \mu(x_0|s^{\mu}) p_{\tau}(\tau|x_0) d\tau dx_0 ds^{\mu}$, where p_{τ} is the probability density function of τ given x_0, and q is the probability density function of s^{μ}. Let Ω be the set of allowed values for (τ, x_0, s^{μ}) then for finite T (4) can be rewritten to reveal the relationship between μ and r,

$$J = \int_{\Omega} q(s^{\mu}) \mu(x_0|s^{\mu}) \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t, s_{t+1}) \prod_{k=0}^{t} \pi(a_k^{\pi}|s_k^{\pi}) \rho(a_k^{\rho}|s_k^{\rho}) p(s_{k+1}|s_k, a_k) \mathrm{d}\tau \mathrm{d}x_0 \mathrm{d}s^{\mu}. \tag{5}$$

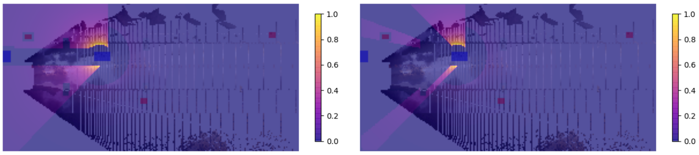


Figure 2: *Left:* A top view image of a sample prior P of μ. In red are other pedestrians, and in blue are cars. The prior implies a higher likelihood of pedestrian initial placement which are close to the AV. *Right:* The same prior after a multiplication with the occlusion map O.

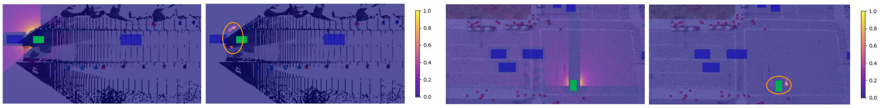


Figure 3: Top-view of two different scenarios. The AV is green, external cars and pedestrians are blue and dark red, respectively. In each of the two examples, the left image shows the prior distribution and the right image shows the final initial distribution. *Left example:* The prior P induces a high likelihood for initializing a pedestrian close to the AV, but the probability map is very smeared out. The final distribution $P\mu$ is much less scattered than P and more peaked close to the AV (indicated also with an external orange ellipsoid, to more clearly show where the probability mass is). *Right example:* We see a similar phenomenon as in the left example, in the dense dataset (see supplement).

From the above it is clear that μ's one step reward is $R_\mu = \sum_{t=0}^{T-1} \gamma^t r_\mu(s_t, a_t, s_{t+1})$. We use REINFORCE [60] to find μ. The models ρ and μ can be learnt simultaneously as shown in the supplement. If the environment model p is known, we can try to find the closed form solution of (5). Using Bellman equations would then allow for a white-box treatment of the AV and the pedestrian.

2.2 Adversarial Test Synthesizer

The ATS is a policy gradient agent, with policy μ. Its objective is to provide an initial position x_0 to the pedestrian agent such that the pedestrian collides with the AV. To do so the ATS needs to find locations near the AV where pedestrians and their motion are difficult to detect for the AV. To this end the ATS gets an input consisting of a top view image of the scene, the prior P of μ that depends on the distance to the AV, and the temporal mapping of dynamic objects D. The initial distribution μ depends also on the scene semantics S, as ATS should learn that pedestrians are more likely to reside near certain semantic classes such as pavement. The policy μ is conditioned on the state s^μ of size ($128\times256\times C$), where $C = 17$ is the number of channels. Due to the success of neural networks in vision tasks and the visual nature of the input μ is modelled by a two layered convolutional neural network with bi-linear interpolations and a softmax output layer (see Fig.1 in supplement). The output of the network $\mu(s^\mu)$ is a heatmap of size (128×256). The heatmap is multiplied by the prior P (see Fig. 2) to avoid sampling x_0 that cannot possibly lead to a collision. To enforce visual feasibility the ATS can be required to sample x_0 only from locations that are occluded for the AV. This can be done by sampling $x_0 \sim OP\mu(s^\mu)$, *i.e.*the product of the occlusion map O, the prior P and $\mu(s^\mu)$.

The ATS's reward r_μ evaluates at timestep t the pedestrian's behavior at position x_t. Collisions with all external objects, cars and pedestrians are penalized but collisions with the AV are given a positive reward. Steps x_t taken in areas often visited by pedestrians are rewarded. Steps a_t towards the goal g^π are rewarded. The reward r_μ is adapted from r_π §2.3.

2.3 Pedestrian Model

The collision avoiding pedestrian behavior policy π is the goal driven Semantic Pedestrian Locomotion model (CARLA SPL) [46]. The π is a policy gradient agent that is trained by alternatively optimizing π for the maximum likelihood objective of pedestrian trajectory forecasting and for the policy gradient objective of collision avoidance. The reward function r_π (see supplement) of π encourages motion in pedestrian dense areas with the reward term R_{ped} and penalizes collisions with cars (including the AV), other pedestrians and static objects in the reward term R_{coll}. The reward component R_g encourages movement towards the goal location g^π, and R_ϕ penalizes unnaturally large motions.

The model observes a local crop $S_t(x_t)$ of size $5m \times 5m$ of the semantic labels and RGB top view image of scene S and a local crop $D_t(x_t)$ of the dynamic occupancy map D_t. Further the state s^π contains a history of past actions and poses taken by the pedestrian in the past $N = 12$ timesteps, the displacement to the closest car d_t^x and the displacement to the goal g^π. In summary $s_t^\pi = (S_t(x_t), D_t(x_t), a_{t-1}^\pi \dots a_{t-N}^\pi, d_t^x, \|x_t - g^\pi\|)$. The policy gradient model takes a step a_t^π consisting of a direction and a speed. The step a_t^π is articulated by the Human Locomotion Network.

Unless otherwise stated the pedestrian models weights are kept constant to not deviate from the learnt pedestrian motion. The CARLA SPL model is trained to avoid collisions with the external cars. The external cars have a lower average speed than the highest possible speed for ρ. This implies that π expects ρ to always have the same dynamics as its surrounding cars.

2.4 Autonomous Vehicle Model

The AV model ρ is intentionally simple to illustrate the framework empirically and to avoid making constraining assumptions about the AV. The focus of this work is to show that collision avoiding pedestrian behavior models can be successfully used in autonomous AV test case generation given enough constraints on the problem. The AV is a policy gradient model with the state $s_t^\rho = (\|x_t - y_t\|, d_t, \delta_t)$ at timestep t; where $\|x_t - y_t\|$ is the AV's distance to the pedestrian agent, d_t is the AV's distance to the closest car, and δ_t is the AV's intersection with the sidewalk. The AV's speed c_t is sampled from $\mathcal{N}(\text{sigmoid}(w^T s_t^\rho + b), \sigma_\rho)$, where w, b are learnt weights, and $\sigma_\rho = 0.1$. The sampled speed c_t is then scaled by the maximal speed of 70km/h. The AV's initial position y_0 and direction are chosen randomly among the external cars' constant velocity future trajectories.

The AV ρ is assumed to have a constant direction and the policy gradient model controls the speed of the AV. Speed control can be enough to avoid collisions, as the AV can stop or accelerate to avoid a collision. Extending the AV's model to allow directional changes complicates the learning as the AV receives two conflicting objectives: to move to a goal location further ahead and to avoid collisions. The research on AVs deals with balancing such conflicting objectives, and in the future we aim to replace the minimal AV model with a state-of-the-art AV model. Replacing the current AV model with a state-of-the-art AV model requires additional constraints to avoid 0 gradients in early training (as the trained AV model may outperform the untrained ATS).

The reward function r_ρ penalizes the AV for collisions with cars, people and static objects. The AV is penalized for driving on the sidewalk proportionally to the AV's overlap with the sidewalk. To motivate the AV to move, a positive reward is given at the end of the episode for the distance travelled $\|y_0 - y_T\|$. The full reward function r_ρ is given in the supplementary.

3 Experiments

We experiment on a dataset gathered from CARLA[61]. Training data is collected from Town 1 and consists of 100 training and 50 validation scenes. The test set consists of 37 scenes from Town 2. For each scene a 3d reconstruction of RGB and semantic segmentation is created from a AV's perspective. In all experiments the scenes $51m \times 25.6m$ are voxelized into $20cm$ cube voxels. All of the tested ATS models are evaluated and trained with the *base AV model*. During initial experimentation it was noted that the AV model ρ had trouble learning collision avoidance without an initializer μ. The *base AV model* is trained on two scenes for 200 epochs with a μ that is trained on the training dataset for 10 epochs. A trajectory length of $T = 30$ is used to train the AV model, and $T = 100$ is used to train the pedestrian initial distribution models. Each scene is evaluated for 10 episodes with $T = 100$. During testing the pedestrian and AV models perform the mode and mean actions respectively. The action of the ATS model is sampled. The models are evaluated with three different random seeds and the average and the standard deviation (stdev) of the three runs are reported. The reported metrics are

– #. collisions - number of collisions the AV model ρ has with pedestrians on average.
– π-entropy - entropy of the pedestrian policy during the length of an episode.

In Table 1 *left* the proposed pedestrian initial distribution models $OP\mu$ and $P\mu$ from §2.2 generate more than twice as many collisions (std=0.01) as sampling x_0 from the priors P and the occlusion-

Table 1: *Left:* The proposed $OP\mu$ and $P\mu$ generate more than twice the collisions compared to the baselines; the heuristics the priors P and OP and the random initialization from occluded spaces *Random* O. *Right:* An ablation studying the effect of the prior during the training of μ shows that the μ is robust to changes in the prior during training as $OP\mu$ and $P\mu$ trained with the priors OP and P respectively, and tested with the prior OP, have indistinguishable collision rates (stdev 0.01).

	Random O	Prior P	Prior OP	$OP\mu$	$P\mu$	Testing prior OP	$P\mu$	$OP\mu$
#. collisions	0.06	0.10	0.09	**0.22**	**0.24**	#. collisions	**0.21**	**0.22**
π-entropy	0.85	0.60	0.65	**0.25**	**0.24**	π-entropy	0.29	**0.25**

Table 2: Collision rates of the $P\mu$ model trained with collision avoiding, distracted, collision seeking and constant velocity pedestrians. The pedestrian model does not affect the collision rate of the proposed μ, as long as the pedestrian model is not the constant velocity model.

	Collision avoiding SPL	Distracted SPL+ϵ	Adversarial SPL A.	Adversarial STPN	Constant velocity HLN
#. collisions	**0.21**($\pm$0.02)	**0.22**($\pm$0.03)	**0.22**($\pm$0.01)	**0.19**($\pm$0.01)	0.11($\pm$0.02)
π-entropy	0.29($\pm$0.01)	0.25($\pm$0.02)	0.029($\pm$0.001)	0.53($\pm$0.03)	**0**

masked prior OP. This confirms that μ learns and improves beyond the initial prior distribution, and that $OP\mu$ produces more collisions than the hand-designed heuristics P and OP. The baseline *Random O* the random initialization of pedestrians from occluded spaces with 360°field of view has the lowest collision frequency. This is likely because occluded spaces may be far from the AV. The proposed $OP\mu$ has a lower π-entropy than the prior OP suggesting that $OP\mu$ has learnt to initialize the pedestrian such that the pedestrian's direction of movement is as predictable as possible. With low π-entropy μ has more control over π's trajectory. To the *left* in Fig. 3 the prior P and the corresponding scene's $P\mu$ distribution are visually compared. The $P\mu$ has learnt decisively to initialize the pedestrian near the AV, and with a higher probability towards the sidewalk than the road.

In Table 1 *left* the models $OP\mu$ (i.e. μ trained and tested with the prior OP) and $P\mu$ (i.e. μ trained and tested with prior P) showed no significant difference. Showing that a 90°view occlusion map does not significantly affect μ. Further applying the occlusion mask O only in testing does not affect the number of collisions, as seen when comparing $P\mu$ to $OP\mu$ in Table 1 *right*. This suggests that curriculum learning may be used to enforce larger changes to the prior P to facilitate 360°field of view occlusion masks (for LiDAR data). A visual comparison of P and OP can be seen in Fig. 2.

In Table 2 the following pedestrian behavior policies are used to train μ,

– *Collision avoiding SPL* - the goal reaching collision avoiding pedestrian model described in §2.3
– *Distracted SPL+*ϵ - a distracted SPL pedestrian. With a 0.3 probability at each timestep the pedestrian will not notice the AV for $m \sim Poisson(2)$ timesteps.
– *Adversarial SPL A.* - an adversarial SPL agent. The *SPL* model that is finetuned with the R^{μ} reward. *SPL A.* is trained simultaneously with μ (see supplementary Algorithm 1 with $\alpha_{\rho} = 0$).
– *Adversarial STPN A.* - an adversarial agent that has the Semantic Trajectory Policy Network architecture [46] i.e. the SPL architecture without the Human Locomotion Network (HLN). The *STPN A.* is trained from random weights simultaneously with μ to maximize the the number or collisions ($R_{STPN} = I$ from §2). *STPN A.* is not trained to maximize the negative log-likelihood of pedestrian trajectories like the *SPL* models, and it is the only model without locomotion.
– *Constant velocity CV* - constant velocity motion articulated by [62]. The agent moves towards the goal with a speed drawn from a Gaussian with $\mu = 1.23\text{ms}^{-1}$ and $\sigma = 0.3$ [63].

The models $P\mu$ trained with the collision avoiding *SPL*, the distracted *SPL+*ϵ, the adversarial finetuned pedestrian policy *SPL A.* and the adversarial *STPN A.* (most similar to previous work) are on-par, showing that μ can learn to control the collision avoiding *SPL* as well as an adversarial pedestrian model. The collision seeking *STPN A.* does not outperform the collision avoiding *SPL* likely due to *STPN A.*'s high entropy that makes *STPN A.* hard to control for μ. *STPN A.* has no motion priors and can get hit by the AV with motions that have a low likelihood in real pedestrian trajectories, such as zigzagging in the middle of the road. Even though the *CV* model is the most controllable, the initializer trained to control *CV* results in the lowest collision rate because the AV has an easy time avoiding collisions with the *CV*. The μ trained on *SPL+*ϵ distracted pedestrian could be expected to have a higher collision rate than *SPL*, as π has a noisier estimate of the AV's position. Unfortunately μ does not learn to utilize this unnaturally unstructured (and thus unpredictable) noise.

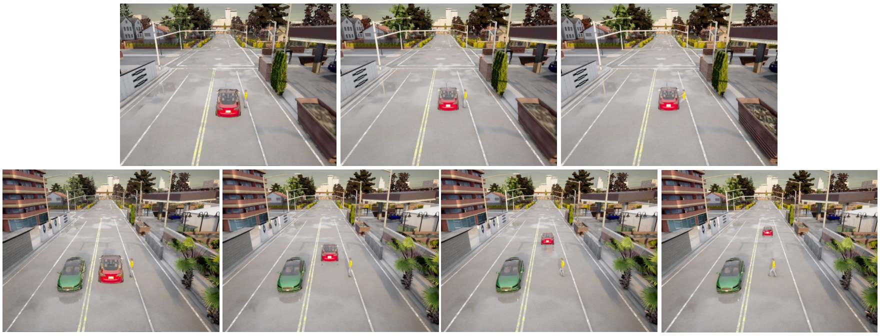


Figure 4: Sample trajectories of the *Simultaneous-μ, ρ* model, sub-sampled at 5 frames from frame 0. *First row:* The AV changes speed thus causing the pedestrian to incorrectly estimate the AV's motion and walk into the AV. *Second row:* The pedestrian waits for the AV to pass before crossing the road.

This illustrates the need for a realistic noise model. Natural noise in the pedestrian's observation of the AV could be expected to be more structured, for example high noise levels are expected near occluded spaces. Some sample trajectories of the simultaneously trained μ and ρ (see supplement) are shown in Fig. 4 - a collision prone initialization, and an initialization that does not lead to a collision because the AV speeds away.

4 Conclusions and Future Work

We are the first to utilize state-of-the-art pedestrian forecasting models in generative AV testing. We have proposed a general framework that is capable of stress testing the collision avoidance of AVs with a wide range of pedestrian behavior models. In practice we wish to ensure that an AV can avoid collisions with all pedestrians (intoxicated, law-obedient, children etc.), and thus should test the AV with as many different pedestrian behaviors as possible. Our empirical evaluations show that a goal driven pedestrian model with any behavior can be used in this framework. This is a significant result, as no prior work has shown that a collision avoiding pedestrian model can be used to generate collisions with a collision avoiding AV. To achieve this, we have proposed the *Adversarial Test Synthesizer (ATS)* which, given any goal driven pedestrian model, learns the pedestrian initial location distribution μ that maximizes the expected number of collisions with a given AV. The ATS is modelled by a neural network which receives as input the top view image of the scene, the scene semantics, the occupancy of dynamic objects, and outputs a distribution μ over pedestrian initial locations. We have shown that μ can learn to adversarially position a collision avoiding pedestrian model that has been trained on ground truth pedestrian data and obeys human locomotive dynamics. Our work, for the first time, shows that generative models of AV test scenarios can utilize state-of-the-art pedestrian motion models instead of the typically used models which do not resemble real pedestrian motion. Stress testing AVs with state-of-the-art pedestrian forecasting models decreases the statistical difference between tested and real pedestrian behaviors, which could reduce the likelihood of real life AV crashes.

We have shown that a learnable pedestrian initial location distribution μ exists for stress testing a basic AV model. Ultimately we wish to extend the result to state-of-the-art AV models. Since the model μ treats the AV and the pedestrian agent as black-boxes, μ can be trained to adjust to the dynamics of a more sophisticated AV as is. However, finding a non-trivial solution will require a careful readjustment of the choice of sufficient but realistic constraints. The problem can be constrained spatially by tight streets, occlusion dense scenes, lack of space due to traffic density, or by setting a limit on the pedestrian's maximal distance to the AV. Alternatively, the pedestrian model can be constrained by adjusting the noise level of the pedestrian's internal prediction of the AV's future motion, the noise level of the pedestrian's observation of the AV, or the pedestrian's goal, dynamics or personality. Similarly, the noise in the AV's observation of pedestrians and other traffic participants and their motion should be high enough to lead to collisions. As seen in the experiments, unstructured noise cannot be utilized by μ, thus careful modelling of the noise of the chosen AV's observations is required. In future work the pedestrian's internal prediction of the AV's motion could be impaired with a psychologically or physiologically inspired noise process.

Acknowledgments

This work was supported by the European Research Council Consolidator grant SEED, CNCS-UEFISCDI PCCF-2016-0180, and the Swedish Foundation for Strategic Research (SSF) Smart Systems Program. We would also like to thank the anonymous reviewers for their useful comments.

References

[1] Z. Zhu and H. Zhao. A survey of deep rl and il for autonomous driving policy learning. *arXiv preprint arXiv:2101.01993*, 2021.

[2] F. Ye, S. Zhang, P. Wang, and C.-Y. Chan. A survey of deep reinforcement learning algorithms for motion planning and control of autonomous vehicles. *arXiv preprint arXiv:2105.14218, accepted in IEEE Transactions on intelligent transportation systems.*, 2021.

[3] A. Sadat, M. Ren, A. Pokrovsky, Y.-C. Lin, E. Yumer, and R. Urtasun. Jointly learnable behavior and trajectory planning for self-driving vehicles. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3949–3956. IEEE, 2019.

[4] W. Zeng, S. Wang, R. Liao, Y. Chen, B. Yang, and R. Urtasun. Dsdnet: Deep structured self-driving network. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 156–172. Springer, 2020.

[5] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl. Learning by cheating. In *Proceedings of Conference on Robot Learning (CoRL)*, pages 66–75. PMLR, 2020.

[6] M. Toromanoff, E. Wirbel, and F. Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7153–7162, 2020.

[7] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1 (1):33–55, 2016.

[8] Q. Dai, X. Xu, W. Guo, S. Huang, and D. Filev. Towards a systematic computational framework for modeling multi-agent decision-making at micro level for smart vehicles in a smart world. *Robotics and Autonomous Systems*, 144:103859, 2021.

[9] P. Kohli and A. Chadha. Enabling pedestrian safety using computer vision techniques: A case study of the 2018 uber inc. self-driving car crash. *In Proceedings of Future of Information and Communication Conference (FICC) 2019, Lecture Notes in Networks and Systems, vol 69.*

[10] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[11] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *I. J. Robotics Res.*, 32(11):1231–1237, 2013.

[12] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov. Scalability in perception for autonomous driving: Waymo open dataset, 2019.

[13] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision (IJCV)*, 2018.

[14] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[15] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[16] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[17] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, et al. Lyft level 5 av dataset 2019. *level5. lyft. com/dataset*, 2:5, 2019.

[18] Y. Abeysirigoonawardena, F. Shkurti, and G. Dudek. Generating adversarial driving scenarios in high-fidelity simulators. In *Proceedings of International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pages 8271–8277. IEEE, 2019.

[19] A. Hamdi, M. Mueller, and B. Ghanem. SADA: semantic adversarial diagnostic attacks for autonomous applications. In *Proceedings of Thirty-Fourth Conference on Artificial Intelligence (AAAI)*, pages 10901–10908. AAAI Press, 2020.

[20] P. Gupta, D. Coleman, and J. Siegel. Towards safer self-driving through great pain (physically adversarial intelligent networks). *ArXiv*, abs/2003.10662, 2020.

[21] J. Wang, A. Pun, J. Tu, S. Manivasagam, A. Sadat, S. Casas, M. Ren, and R. Urtasun. Advsim: Generating safety-critical scenarios for self-driving vehicles. *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[22] D. Karunakaran, S. Worrall, and E. Nebot. Efficient falsification approach for autonomous vehicle validation using a parameter optimisation technique based on reinforcement learning, 2020.

[23] D. Karunakaran, S. Worrall, and E. M. Nebot. Efficient statistical validation with edge cases to evaluate highly automated vehicles. In *23rd IEEE International Conference on Intelligent Transportation Systems, ITSC 2020, Rhodes, Greece, September 20-23, 2020*, pages 1–8. IEEE, 2020.

[24] W. Ding, B. Chen, M. Xu, and D. Zhao. Learning to collide: An adaptive safety-critical scenarios generating method. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021*, pages 2243–2250. IEEE, 2020.

[25] W. Ding, B. Chen, B. Li, K. J. Eun, and D. Zhao. Multimodal safety-critical scenarios generation for decision-making algorithms evaluation. *IEEE Robotics and Automation Letters*, 6(2):1551–1558, 2021.

[26] P. Parashar, A. Cosgun, A. Nakhaei, and K. Fujimura. Modeling preemptive behaviors for uncommon hazardous situations from demonstrations. *CoRR*, abs/1806.00143, 2018.

[27] M. Wen, J. Park, and K. Cho. A scenario generation pipeline for autonomous vehicle simulators. *Human-centric Computing and Information Sciences*, 10, 12 2020.

[28] A. Demetriou, H. Alfsvåg, S. Rahrovani, and M. Chehreghani. A deep learning framework for generation and analysis of driving scenario trajectories. *ArXiv*, abs/2007.14524, 2020.

[29] D. Nishiyama, M. Y. Castro, S. Maruyama, S. Shiroshita, K. Hamzaoui, Y. Ouyang, G. Rosman, J. A. DeCastro, K. Lee, and A. Gaidon. Discovering avoidable planner failures of autonomous vehicles using counterfactual analysis in behaviorally diverse simulation. In *23rd IEEE International Conference on Intelligent Transportation Systems, ITSC 2020, Rhodes, Greece, September 20-23, 2020*, pages 1–8. IEEE, 2020.

[30] W. Ding, M. Xu, and D. Zhao. CMTS: A conditional multiple trajectory synthesizer for generating safety-critical driving scenarios. In *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*, pages 4314–4321. IEEE, 2020.

[31] Z. W. Sun X., Zhang Y. Building narrative scenarios for human-autonomous vehicle interaction research in simulators. *Advances in Simulation and Digital Human Modeling*, 1206, 2020.

[32] M. Koren and M. J. Kochenderfer. Efficient autonomy validation in simulation with adaptive stress testing. In *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019, Auckland, New Zealand, October 27-30, 2019*, pages 4178–4183. IEEE, 2019.

[33] A. Rasouli and J. K. Tsotsos. Autonomous vehicles that interact with pedestrians: A survey of theory and practice. *IEEE transactions on intelligent transportation systems*, 21(3):900–918, 2019.

[34] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras. Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, 39(8): 895–935, 2020.

[35] P. Dendorfer, A. Osep, and L. Leal-Taixe. Goal-gan: Multimodal trajectory prediction based on goal position estimation. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020.

[36] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[37] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, H. Rezatofighi, and S. Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In *NeurIPS*, 2019.

[38] L. Zhang, Q. She, and P. Guo. Stochastic trajectory prediction with social graph network. *CoRR*, abs/1907.10233, 2019.

[39] Y. Huang, H. Bi, Z. Li, T. Mao, and Z. Wang. Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[40] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[41] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, F. Li, and S. Savarese. Social LSTM: human trajectory prediction in crowded spaces. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[42] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[43] W. Luo, B. Yang, and R. Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[44] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu. Multi-agent tensor fusion for contextual trajectory prediction. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[45] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[46] M. Priisalu, C. Paduraru, A. Pirinen, and C. Sminchisescu. Semantic synthesis of pedestrian locomotion. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020.

[47] P. Kothari, S. Kreiss, and A. Alahi. Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[48] C. Yu, X. Ma, J. Ren, H. Zhao, and S. Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 507–523. Springer, 2020.

[49] Y. Liu, Q. Yan, and A. Alahi. Social nce: Contrastive learning of socially-aware motion representations. *arXiv preprint arXiv:2012.11717*, 2020.

[50] X. Ren, T. Yang, L. E. Li, A. Alahi, and Q. Chen. Safety-aware motion prediction with unseen vehicles for autonomous driving. *In Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021.

[51] O. Makansi, Ö. Çiçek, K. Buchicchio, and T. Brox. Multimodal future localization and emergence prediction for objects in egocentric view with a reachability prior. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4353–4362. IEEE, 2020.

[52] J. Sun, H. Averbuch-Elor, Q. Wang, and N. Snavely. Hidden footprints: Learning contextual walkability from 3d human trails. In A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, editors, *In Proceedings of European Conference on Computer Vision (ECCV)*, volume 12363 of *Lecture Notes in Computer Science*, pages 192–207. Springer, 2020.

[53] J. Mockus. On bayes methods for seeking an extremum. *Avtomatika i Vychislitelnaja Technika*, 3:53–62, 1972.

[54] D. Bertsekas. *Reinforcement and Optimal Control*. Athena Scientific, 2019.

[55] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin. Hamilton-jacobi reachability: A brief overview and recent advances. In *Proceedings of the IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2242–2253. IEEE, 2017.

[56] S. Manivasagam, S. Wang, K. Wong, W. Zeng, M. Sazanovich, S. Tan, B. Yang, W. Ma, and R. Urtasun. Lidarsim: Realistic lidar simulation by leveraging the real world. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11164–11173. IEEE, 2020.

[57] K. Zhang, Z. Yang, and T. Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pages 321–384, 2021.

[58] Q. Dai, X. Xu, W. Guo, S. Huang, and D. Filev. Towards a systematic computational framework for modeling multi-agent decision-making at micro level for smart vehicles in a smart world. *Robotics and Autonomous Systems*, 144:103–859, 2021. ISSN 0921-8890.

[59] E. Smolyakov. Necessary and sufficient conditions for the existence of a saddle point in parametrized programmed differential games. *Cybernetics and Systems Analysis*, 33(5):686 – 693, 1997. ISSN 10600396.

[60] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.

[61] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: An open urban driving simulator. In *Proceedings of Conference on Robotic Learning (CoRL)*, 2017.

[62] D. Holden, T. Komura, and J. Saito. Phase-functioned neural networks for character control. *ACM Trans. Graph.*, 36(4):42:1–42:13, 2017.

[63] S. Chandra and A. K. Bharti. Speed distribution curves for pedestrians during walking and crossing. *Procedia-Social and Behavioral Sciences*, 104:660–667, 2013.

Supplementary Material of Generating Scenarios with Diverse Pedestrian Behaviors for Autonomous Vehicle Testing

Maria Priisalu[1], **Aleksis Pirinen**[*2], **Ciprian Paduraru**[3,4], **and Cristian Sminchisescu**[1,5]

[1]Lund University, [2]RISE Research Institutes of Sweden, [3]University of Bucharest, [4]Institute of Mathematics of the Romanian Academy, [5]Google Research

1 Additional visualizations: supplementary videos

In the attached video `Supplemenatry_Scenarios_with_Diverse_Pedestrian_Behaviors_for_AV_Testing.mp4` a number of sample trajectories are shown. The video contains sample trajectories where the pedestrian initial positions are sampled from the prior P, the model Pμ-D from Table 2 ($P\mu$-D is trained on the dense dataset D) and the model Simultaneous-μ, ρ from Table 3. The trajectories sampled from the prior P and from the model Pμ-D are shown with an untrained AV ρ, as this is what the model Pμ-D is trained on. The pedestrian initial position is sampled from the respective model and the pedestrian behaviour model described in §2.3 of the main paper is used to roll out the pedestrian trajectory. The visualizations show sample trajectories with a length of at most 100 steps (they are edited to stop when the pedestrian and AV collide). The trajectories are performed on a visualization scene that is gathered in the same fashion as the dense dataset, but is not a part of the dense dataset. The first frame is kept still for 6s to ease detecting the pedestrian's and AV's initial positions.

The video shows three sample trajectories where the pedestrian model π is initialized by sampling from the prior $x_0 \sim P$. It can be seen that even when the pedestrian model is initialized near the AV it seeks to reach a sidewalk, or walks along the middle of the road avoiding collisions to reach its goal. Note that goals are placed out as before: by reflecting the pedestrian's position x_0 in the constant velocity prediction of AV's trajectory. These sample trajectories visualize that it is not trivial where to place the pedestrian to ensure a collision. The model μ (see Fig. 1), has initially random weights. Therefore samples drawn from the initial $P\mu$ strongly resemble the samples of prior P.

The next three sample trajectories are from the model Pμ-D trained on the dense dataset. The samples show that the model has learnt to initialize the pedestrian model π such that the pedestrian misjudges the AV's motion and gets hit by the AV. Note that the AV could have 0 speed already at the first timestep/frame and thus avoid collisions by standing. The fourth sample trajectory from the Pμ-D model illustrates a failure case for the initialization model. In the fourth trajectory of the Pμ-D model it can be seen that if the AV and the pedestrian are initialized far away from one another then the initializer has little control over the pedestrian's trajectory and it is harder for the initializer to enforce a collision. Note the collision avoidance behaviour of the pedestrian model. In this trajectory the pedestrian curves around the AV to increase its distance to the AV.

Finally we see a sample failure case for the Simultaneous-μ, ρ model where both μ and ρ are trained simultaneously. The AV and the pedestian model are initialized close by, but both the the pedestrian and AV are good enough at collision avoidance to avoid a collision. This trajectory visualizes that problem of initializing a pedestrian such that it gets hit by a AV is not trivial. Even though the pedestrian is running towards the AV, the AV has learnt to accelerate to avoid collisions.

[*]Work partially done while at Lund University.

5th Conference on Robot Learning (CoRL 2021), London, UK.

2 The proposed μ model

In the following the details of the pedestrian initial distribution model μ are provided. Its objective is to model the distribution of initial positions x_0 of the pedestrian agent locations from where the pedestrian collides with the AV ρ. A sample trajectory of the Monte Carlo estimate of the gradient of J_π is evaluated by sampling the initial position of the pedestrian x_0 from μ, the pedestrian actions $a^\pi \sim \pi$ and the AV actions from $a^\rho \sim \rho$. The roll-outs are evaluated by a reward function r_μ that rewards collisions between the AV with position y_t and the pedestrian x_t, where $t \in [0, T]$ is the timestep. Since the ATS cannot control the actions of the behaviour policy π beyond the first timestep, the model μ does not receive a new state in response to the chosen action, only a reward r_μ. The pedestrian's initial position $x_0 \sim OP\mu(s^\mu)$ is considered to be the action taken by the policy gradient agent μ.

2.1 Model input s^μ

The pedestrian distribution model μ observes the scene as $s^\mu = (S, D, OP)$. Here S contains the top view RGB image and semantic labels of the scene (possibly constructed from a reconstruction). The same semantic labels are used as in the pedestrian model π. The dynamic mapping D contains the constant velocity predictions of the external cars, the AV and the external pedestrians. The dynamic map D is the reciprocal of the dynamic map used in the pedestrian behaviour model π, and contains a separate channel for cars and pedestrians. Finally μ observes the product of the occlusion map and the prior OP.

The proposed model μ takes as input a tensor s^μ of size (128×256×C) where $C = 17$ is the number of channels. The input channels contain the RGB channels of the top view of the scene at timepoint $t = 0$. The s^μ contains 9 channels for the semantic segmentation of the static objects in the scene. The s^μ contains two channels for the inverted dynamic occupancy map D and two channels containing the occupancy of the AV and external pedestrians and cars at timestep 0. Finally the occlusion-map masked prior OP is input as a separate channel to s^μ, to inform μ of which car to challenge.

2.1.1 The prior P

Given that the pedestrian has a maximal speed of $\|v^\pi_{max}\| = 3ms^{-1}$ there exists a cone of points h from which the pedestrian can reach the AV's constant velocity trajectory. The prior for the points in $x \in h$ is $\|x - y_0\|^{-1}$ where y_0 is the initial position of the AV. The prior $P(x)$ is 0 within the braking distance $\|v^\rho_0\|^2/(2g * 0.8)$ of the AV assuming dry road conditions (0.8 as friction coefficient), and v^ρ_0 is the AV's initial velocity. This is to avoid sampling from the trivial initializations within the AV's braking distance, thus leading to an inevitable collision. The points x that are on the constant velocity estimate of the AV's trajectory receive a 0 prior. Finally for all other points the prior is $||x - y_0||^{-2}$. The prior can be summarized as follows,

$$P(x) = \begin{cases} \|x - y_0\|^{-1} & \text{if } x \in h \\ 0 & \text{if } \|x - y_0\| < \frac{(v^\rho_0)^2}{250*0.8} \\ 0 & \text{if } x \text{ is on the line } y_0 + t * v^\rho_0, t > 0 \\ \|x - y_0\|^{-2} & \text{all other x,} \end{cases} \tag{1}$$

where x is a point in the scene, y_0 is the AV's initial location, h is the cone of points from which the pedestrian can reach the AV's constant velocity trajectory, v^ρ_0 is the AV's initial velocity, and t is time. The edges of the cone h are easily found by defining the AV's constant velocity v^ρ_0 (AV's initial velocity at timestep $t = -1$) future motion as a line $\hat{y_t} = y_0 + t * v^\rho_0$. The shortest distance from any point x in front of the AV to the AV's future trajectory $\hat{y_t}$ is the distance from the point x to the orthogonal projection $x_\perp$ of the point in the line $\hat{y_t}$. Let the constant velocity AV reach $x_\perp$ at timepoint $\hat{t}$. Then if $\|x - x_\perp\| < \hat{t}\|v^\pi_{max}\|$, where $\|v^\pi_{max}\| = 3ms^{-1}$ is the maximal speed of the pedestrian, then the point $x \in h$.

2.1.2 The semantic segmentation of static objects in S

The RGB and semantic top view of the scene's static objects is referred to as S. The construction of the semantic map and the top view RGB of S follow the procedure of [1]. The semantic labels used are building, fence, static obstacles, pole, road, sidewalk, vegetation, wall and traffic sign/light. The

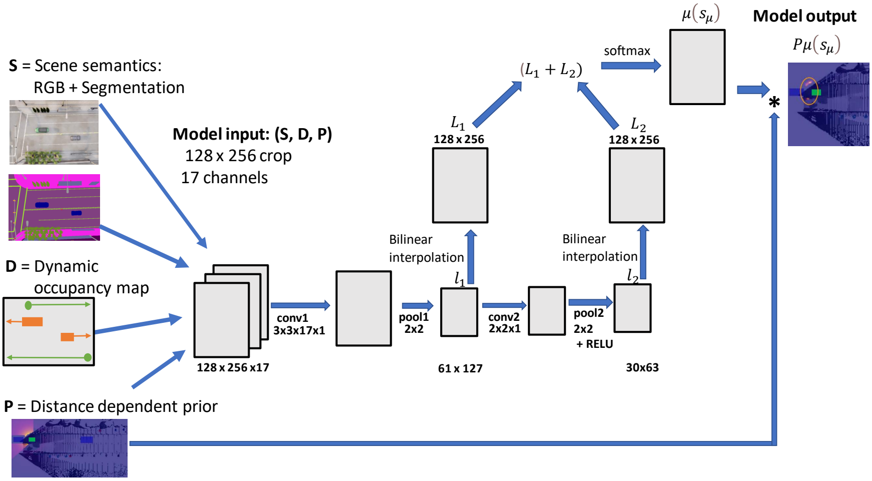


Figure 1: Pedestrian initial spatial distribution $P\mu$ architecture. The model input consists of channels for the top view scene semantic and RGB S, the dynamic occupancy map D, and the prior P that may be replaced by PO to enforce initialization in occluded spaces only. Note that the dynamic occupancy map D of μ and the dynamic occupancy map D_t of π are different. The neural network output is multiplied by the prior P to produce the pedestrian initial spatial distribution.

semantic segmentation is obtained by gathering 2D semantic labeled images from the data gathering stationary AV's perspective together with the depth map of the scene in the same perspective. The 3D points of each pixel can then be reconstructed from the depth map, and a segmentation label and a RGB color can be assigned to each 3D point. This is done for very 50th gathered frame (500 frames in total). Finally mode-voting is applied to obtain the semantic class of a 3D point, and mean to attain the color. The dense dataset is gathered from a moving drone's perspective. Finally the 3D reconstruction is voxellized and projected into the top-view perspective.

2.1.3 The dynamic occupancy map D

The dynamic occupancy map contains the constant velocity estimates of the cars and external pedestrians in separate channels. This provides the model with the most basic car and pedestrian motion estimates. Given that a car's constant velocity estimate at timestep t is the bounding box b_t, the pixels in the bounding box b_t are set to $D(b_t) = 1/t$ if $D(b_t) < 1/t$. That is the earliest occupancy is noted in the inverted dynamic map. The earlier steps in the forecast trajectories are more relevant to μ as they are temporally closer to the pedestrian initialization x_0.

Note that in the pedestrian behaviour model D_t refers to a dynamic occupancy map that is not inverted and that is dependent on the timestep t. D_t contains the AVs, external pedestrians and cars occupancy trajectories up to timestep t, and the constant velocity future trajectory estimates of all pedestrians and cars from timestep t onwards. The map D_t is constructed in the same fashion as D, but D_t also includes occupancy for previous timesteps. Further D_t uses scaled (by constant 0.003 to ensure values lie in range [0,1] for sequence lengths up to 300 timesteps) but not inverted timestamps. The pedestrian observes a local neighborhood of this dynamic map denoted $D_t(x_t)$ which is bigger than the pedestrian.

2.2 Model architecture

The model architecture is visualized in Fig. 1. The proposed μ model has an input of size (128×256×C). This is passed through a (3×3×C×1) convolution followed by a (2×2×1) max-pooling layer. Let the output of this max-pooling layer be referred to as l_1. l_1 is convoluted by a (2×2×1×1) filter, (2×2×1) max-pooling and passed through ReLu activation. Let the output of the Relu activation be called l_2.

Now l_1 and l_2 are bi-linearly interpolated back to the original image resolution of (128×256), let the upsampled layers be denoted by L_1, L_2. The neural network output is then $\mu(s^\mu) = \text{softmax}(L_1 + L_2)$, where $softmax$ is the softmax all of the pixels of the image to ensure that the model output is a distribution. Finally the $\mu(s^\mu)$ is multiplied by the prior P to get the estimated pedestrian initial spatial location distribution $P\mu(s^\mu)$.

3 Reward functions

Here we will provide the details of the reward functions of μ, π, ρ. A number of the reward components are shared between the model components.

3.1 Reward of μ

A number of the reward components are adapted from π. The reward function r_μ consist of collision terms R_v, R_p, R_s that are indicator functions which are activated when the controlled pedestrian's intended next step position $x_t + a_t^\pi$ collides with external vehicles, external pedestrians and static objects respectively. We introduce a new reward term R_a an indicator function that is positive when $x_t + a_t^\pi$ and the AV collide. The reward terms R_v, R_p, R_s are multiplied with negative factors $\lambda_v = -2, \lambda_p = -0.1, \lambda_s = -0.02$ to discourage collisions with external agents and objects, while R_a is multiplied with a positive factor $\lambda_a = 2$.

Further μ is rewarded for initializations that lead to pedestrian motion in areas often traversed by pedestrians. This is done by the reward components R_d and R_k. To allow for per frame updates of the dataset we adapt R_d to be an indicator function which is 1 when $x_t + a_t^\pi$ is intercepting a past external pedestrian trajectory or a constant-velocity predicted pedestrian trajectory (i.e. interception with a non-zero D_t). In a similar fashion we define R_k to reward the pedestrian for being near external pedestrian trajectories. R_k is evaluated as the ratio of non-zero pixels in D_t in the neighborhood $D_t(x_t)$ of the pedestrian x_t. The terms R_d and R_k are multiplied by $\lambda_d = 0.01$ and $\lambda_k = 0.01$.

Further a positive reward is given to μ for steps taken by π towards the goal g^π. Let the indicator function $I_g(s_t, a_t, g^\pi)$ be positive when $x_t + a_t^\pi$ has reached the goal, i.e. $\|x_t + a_t^\pi - g^\pi\| < \epsilon$, where $\epsilon = \sqrt{2}$ pixels. Then $R_g(s_t, a_t, g^\pi) = 1 - \frac{\|x_t + a_t^\pi - g^\pi\|}{\|x_t - g^\pi\|}$ when the agent has not reached the goal location $I_g(s_t, a_t, g^\pi) < 1$. The goal term R_g is multiplied by $\lambda_g = 0.1$.

The reward components R_p, R_s, R_k, R_d, R_g are multiplied together when non-zero. When the pedestrian collides with a vehicle, the AV or reaches a goal location then r_μ is equal to only the reward term R_v, R_a or R_g, and the model receives a 0 reward in the following time-steps. As shown below and in the main paper the initial distribution model's reward is dependent on the full trajectory of the pedestrian and the AV model, and is thus expressed as

$$R_\mu(\tau) = \sum_{t=0}^{T} \gamma^t r_\mu(s_t, a_t, s_{t+1}). \tag{2}$$

Further we can summarize r_μ, as

$$r_\mu(s_t, a_t, s_{t+1}) = \begin{cases} \lambda_a & \text{if } R_a(s_t, a_t, s_{t+1}) \\ \lambda_p & \text{if } R_v(s_t, a_t, s_{t+1}) \\ 0 & \text{if } R_v(s_k, a_k, s_{k+1}) > 0, \text{ for any } k < t \\ 0 & \text{if } R_a(s_k, a_k, s_{k+1}) > 0, \text{ for any } k < t \\ 0 & \text{if } I_g(s_k, a_k, s_{k+1}) > 0, \text{ for any } k < t \\ \Pi(\lambda_p R_p, \lambda_s R_s, \lambda_k R_k, \lambda_d R_d, \lambda_g R_g) & \text{otherwise} \end{cases}, \tag{3}$$

where $\Pi(.)$ is the product of the non-zero inputs, and where in (3) the general lambda followed by the general reward $\lambda_* R_*$ denotes the following function,

$$\lambda_* R_* = \begin{cases} 1 + \lambda_* R_*(s_t, a_t, s_{t+1}) & \text{if } \lambda_* > 0 \\ 1 - \lambda_* R_*(s_t, a_t, s_{t+1}) & \text{if } \lambda_* < 0 \\ 1 & \text{otherwise .} \end{cases} \tag{4}$$

3.2 Reward of π

The reward function of π consist of pedestrian motion encouraging terms R_{ped}, collision penalizing terms R_{coll}, the terms R_g and I_g encouraging movement towards the goal g^π, and a term discouraging unnatural articulated motion R_ϕ. The collision discouraging terms in $R_{coll}(s_t, a_t, s_{t+1}) = \lambda_p R_p(s_t, a_t, s_{t+1}) + \lambda_s R_s((s_t, a_t, s_{t+1}) + \lambda_a^\pi R_a(s_t, a_t, s_{t+1})$, where R_v, R_p, R_s, R_a are described in §3.1, and $\lambda_a^\pi = -\lambda_a$. The reward term R_ϕ penalizes lathe changes in the average yaw ϕ (in degrees) of the joints in the agent's lower body as $R_\phi(x_t, v_t) = \max(\min(\phi - 1.2, 0), 2.0)$. The term R_g follows the definition given in §3.1. The pedestrian agent also receives a large positive reward $\lambda_G = 2$ for reaching its goal location i.e. $I_g(s_t, a_t, s_{t+1}) > 0$. After collision with a car and after reaching a goal the pedestrian agent is considered dead, and thus receives 0 rewards.

The pedestrian motion encouraging term R_{ped} consists of a reward term that promotes motion on pedestrian trajectories R_d^π. The reward R_d^π is an indicator function which is 1 if any pixel in the pedestrian bounding box coincides with D_T. The pedestrian bounding box is centered at the pedestrian's position x_t and is of size 1.2m×1.2m. The second reward term in R_{ped} is R_k^π that rewards the pedestrian for being near external pedestrian trajectories. The pedestrian trajectories in D_T are blurred by an exponential kernel producing a density map D_k. The pedestrian is rewarded $R_k^\pi(s_t, a_t, s_{t+1}) = D_k(x_{t+1})$ is equal to the kernel smoothed valued of the kernel at the pixel x_{t+1}. The pedestrian like motion promoting terms can be gathered as, $R_{ped}(s_t, a_t, s_{t+1}) = \lambda_k R_k^\pi(s_t, a_t, s_{t+1}) + \lambda_d R_d^\pi(s_t, a_t, s_{t+1})$. A small negative reward $R_\phi(s_t, a_t, s_{t+1}) = \max(\min(\phi - 1.2, 0), 2.0)$ is given for excessively large changes in the average lower body joints yaw ϕ of the pedestrian agent's articulated pose. The term R_ϕ is multiplied by $\lambda_\phi = -0.0001$.

Finally the pedestrian reward can be summarized as

$$r^\pi(s_t, a_t, s_{t+1}) = \begin{cases} 0 & \text{if agent is dead} \\ \lambda_v & \text{if } R_v(s_t, a_t, s_{t+1}) > 0 \\ R_{coll} + R_{ped} + \lambda_g R_g + \lambda_G I_g + \lambda_\phi R_\phi & \text{otherwise} \end{cases}, \tag{5}$$

where the reward terms $R_{coll}, R_{ped} R_g, I_g, R_\phi$ are evaluated on (s_t, a_t, s_{t+1}) when input parameters are omitted.

3.3 Reward of ρ

The reward function of the AV ρ contains the collision penalizing terms R_v^ρ, R_p^ρ, R_s^ρ that are analogous to the collision penalizing terms R_v, R_p, R_s. The reward terms $R_v^\rho, R_p^\rho, R_s^\rho$ are indicator functions that are 1 if the learnt AV's planned position $y_t + a_t^\rho$ collides with an external car, any pedestrian or static object in the timestep $t + 1$. The terms can again be gathered $R^\rho_{coll.}(s_t, a_t, s_{t+1}) = \lambda_v^\rho R_v^\rho(s_t, a_t, s_{t+1}) + \lambda_p^\rho R_p^\rho(s_t, a_t, s_{t+1}) + \lambda_s^\rho R_s^\rho(s_t, a_t, s_{t+1})$, where $\lambda_v^\rho = -2, \lambda_p^\rho = -2, \lambda_s^\rho = -2$. The reward function r_ρ contains also the term $R_o(y_t, a_t^\rho)$ that is the ratio of pixels in the AV's bounding box that have the semantic label sidewalk, multiplied by $\lambda_o = -0.1$. Finally we introduce a term to encourage the AV to move. The reward term $R_{dist}(s_t, a_t, s_{t+1}) = \|y_0 - y_t\|/t * v^\rho_{max}$, where v^ρ_{max} =70km/h is the maximal speed of the AV. The reward term R_{dist} is multiplied by $\lambda_{dist} = 0.01$. The AV is considered dead after any collision (i.e. when $\max(R_v^\rho, R_p^\rho, R_s^\rho) > 0$), and thus a reward of 0 is given after any collision. The full reward of the alive AV is,

$$r_\rho(s_t, a_t, s_{t+1}) = R^\rho_{coll.}(s_t, a_t, s_{t+1}) + \lambda_o R_o(y_t, a_t^\rho) + \lambda_{dist} R_{dist}(s_t, a_t, s_{t+1}). \tag{6}$$

4 Simultaneous learning of μ, ρ and π

To gradually improve the AV at collision avoidance the ATS can be used to train the AV. Once the AV improves the ATS can be fitted to the new AV model. This gives rise to the possibility of training the AV and the ATS alternatively or even simultaneously. Here we present algorithms for alternative and simultaneous training of the AV and the ATS model. If the pedestrian model is not dependent on an external dataset then even the pedestrian model π can be trained simultaneously with ATS and AV. In the following experiments in §5.3 we utilize the SPL-goal agent from [1] as the pedestrian model π.

4.1 Policy Gradient Framework for the Problem Described in Section 3.1

We would like to find the parametrized Θ pedestrian initial location distribution μ_Θ that maximizes the objective

$$J_\mu(\Theta) = \mathbb{E}_{x_0\sim\mu_\Theta(.|s^\mu), s^\mu\sim q, a_t^\pi\sim\pi, a_t^\rho\sim\rho, s_t\sim p(.|s_t,a_t)}\left[R_\mu(x_0,\tau)\right], \tag{7}$$

where R_μ is the discounted cumulative reward $R_\mu(x_0,\tau) = \sum_{t=0}^{T-1}\gamma^t r_\mu(s_t, a_t, s_{t+1})$, and τ trajectory of an episode be denoted $\tau = (a_0, s_1, ..., a_{T-1}, s_T)$, and x_0 is the initial pedestrian position location, π and ρ are the behaviour model's of the pedestrian and the AV agent respectively, taking actions a_t^π and a_t^ρ. And $p(s_{t+1}|s_t, a_t)$ is the environment dynamics that predicts the successive state s_{t+1}, where $s_t = (s_t^\pi, s_t^\rho)$ and $a_t = (a_t^\pi, a_t^\rho)$ are vectors containing the states and actions of the pedestrian model π and AV ρ respectively. The traffic scene observations s^μ have a distribution $q(s^\mu)$. Further r_μ is μ's reward function, and t is the current timestep and T is the episode length. Finally s_0 is a function of s^μ, x_0 (see Section 3.1 in main paper) and $x_0 \sim \mu_\Theta(x_0|s^\mu)$. Let the discounted cumulative reward $R_\mu = \sum_{t=0}^{T-1}\gamma^t r_\mu(s_t, a_t, s_{t+1})$ we can express (7) as

$$J_\mu(\Theta) = \int_{s^\mu}\int_{x_0}\int_\tau R_\mu(x_0,\tau)p_\tau(\tau|x_0)\mu_\Theta(x_0|s^\mu)q(s^\mu)d\tau dx_0 ds^\mu, \tag{8}$$

where p_τ is the probability density function of τ given x_0. Then p_τ can be factored as follows,

$$p_\tau(\tau|x_0) = \prod_{t=0}^{T-1}\pi(a_t^\pi|s_t^\pi)\rho(a_t^\rho|s_t^\rho)p(s_{t+1}|s_t, a_t). \tag{9}$$

Now taking a derivative of (9) with respect to the parameters θ we note that only μ depends on θ,

$$\nabla_\Theta J_\mu(\Theta) = \int_{s^\mu}\int_{x_0}\int_\tau \nabla_\Theta\mu_\Theta(x_0|s^\mu)R_\mu(x_0,\tau)p_\tau(\tau|x_0)q(s^\mu)d\tau dx_0 ds^\mu. \tag{10}$$

We can now follow the classical policy gradient method [2] and use $\nabla_\Theta\mu_\Theta = \mu_\Theta\nabla_\Theta\log(\mu_\Theta)$, and rewrite (10) as,

$$\nabla_\Theta J_\mu(\Theta) = \int_{s^\mu}\int_{x_0}\int_\tau \nabla_\Theta\log(\mu_\Theta(x_0|s^\mu))R_\mu(x_0,\tau)p_\tau(\tau|x_0)\mu_\Theta(x_0|s^\mu)q(s^\mu)d\tau dx_0 ds^\mu \tag{11}$$

$$= \mathbb{E}[\log(\mu_\Theta(x_0|s^\mu))R_\mu(x_0,\tau)]. \tag{12}$$

We can evaluate the above expectation with the Markov Chain Monte Carlo method. Given K traffic scenes s_k^μ, with M pedestrian initial locations $x_0^{m,k} \sim \mu(x_0|s_k^\mu)$ in each traffic scene, and N sample trajectories $\tau^{m,k,n} \sim p_\tau(\tau|x_0^{m,k})$ for each pedestrian initialization $x_0^{m,k}$ then the Monte Carlo estimate of 11, is the following

$$\hat{\nabla}_\Theta J_\mu(\Theta) = \frac{1}{NMK}\sum_{m=1}^{M}\sum_{k=1}^{K}\sum_{n=1}^{N} R_\mu(x_0^{m,k}, \tau^{m,k,n})\nabla_\Theta\log(\mu_\Theta(x_0^{m,k}|s_k^\mu)). \tag{13}$$

In the same manner using (9) we can estimate the gradient of π_β with,

$$\hat{\nabla}_\beta J_\pi(\beta) = \frac{1}{NMK}\sum_{m=1}^{M}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\gamma^t r_\pi(s_t^{m,k,n}, a_t^{m,k,n}, s_{t+1}^{m,k,n})\nabla_\beta\log(\pi_\beta(a_t^{\pi,m,k,n}|s_t^{\pi,m,k,n})). \tag{14}$$

And similarly the AV policy's ρ_ξ gradient can be estimated by,

$$\hat{\nabla}_\xi J_\rho(\xi) = \frac{1}{NMK}\sum_{m=1}^{M}\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\gamma^t r_\rho(s_t^{m,k,n}, a_t^{m,k,n}, s_{t+1}^{m,k,n})\nabla_\xi\log(\rho_\xi(a_t^{\rho,m,k,n}|s_t^{\rho,m,k,n})). \tag{15}$$

Finally the three gradient estimates (13),(14),(15) can be estimated from the same sample trajectories, giving rise to Algorithm 1. Alternatively a possibly more stable alternating training scheme could be used as shown in Algorithm 2

Algorithm 1 Learning μ, π and ρ simultaneously

```
Initialize $\Theta_1, \beta_1, \xi_1$ randomly.
Initialize learning rates $\alpha_\Theta, \alpha_\beta, \alpha_\xi$
Set $\mu_1 = \mu(\Theta_1)$, $\pi_1 = \pi(\beta_1)$, $\rho_1 = \rho(\xi_1)$
for $k = 1 \dots K$ iterations do
  Initialize empty set $O = \{\}$
  Sample $s_k^\mu$ from the dataset
  for $m = 1 \dots M$ iterations do
    Sample $x_0^{k,m} \sim \mu_k(x_0|s_k^\mu)$
    for $n = 1 \dots N$ iterations do
      Sample $\tau^{k,m,n} \sim p_\tau(.|x_0^{k,m})$ where $a_t^{\pi,k,m,n} \sim \pi_k$ and $a_t^{\rho,k,m,n} \sim \rho_k$
      Add sample trajectory $\tau^{k,m,n}$ to $O$
    end for
  end for
  Update $\Theta_{k+1} = \Theta_k + \alpha_\Theta \hat{\nabla}_\Theta J_\mu(\Theta)$ using samples in $O$
  Update $\beta_{k+1} = \beta_k + \alpha_\beta \hat{\nabla}_\beta J_\pi(\beta)$ using samples in $O$
  Update $\xi_{k+1} = \xi_k + \alpha_\xi \hat{\nabla}_\xi J_\rho(\xi)$ using samples in $O$
  Update parameters $\mu_{k+1} = \mu(\Theta_{k+1})$, $\pi_{k+1} = \pi(\beta_{k+1})$, $\rho_{k+1} = \rho(\xi_{k+1})$
end for
```

Table 1: The number of epochs the μ of the presented models were trained for

Model	$P\mu$	$OP\mu$	$P\mu$-D	SPL+ϵ	SPL A.	STPN A	CV
	6	9	2	8	9	2	10

5 Experiments

5.1 Hyperparameters in experiments

The Adam optimizer with a learning rate of $\alpha_\mu = 5 \times 10^{-3}$ for μ and $\alpha_\rho = 3 \times 10^{-2}$ for ρ is used in experiments. The weights of μ are initialized randomly, and the weights of ρ are initialized to 1. A discount rate of $\gamma = 0.99$ is used for all of the models. The presented values are of the pedestrian location distribution models μ showed highest validation performance, seen in Table 1.

5.2 Experiments on the Dense CARLA dataset (D)

We introduce the Dense CARLA dataset (D) a smaller more object-dense dataset consisting of 4 different simulations of 5 scenes, gathered from a drone's perspective. The model $P\mu$-D is trained on the dataset and tested on the regular CARLA dataset. The model $P\mu$-D is trained with the reward R_{STPN}, for a fair comparison the model is compared to $P\mu$ trained on the regular dataset with the reward R_{STPN} for 2 epochs. The results are shown in Table 2. The proposed μ is robust to changes in dynamics from training to testing as seen in the small drop in the number of collisions when comparing $P\mu$-D and $OP-\mu$ in Table 2. The model $P\mu$-D trained on the denser dataset leads to fewer collisions than the base model $P\mu$ and has a higher π-entropy than $P\mu$, likely because the π is less controllable by μ in denser traffic. The distribution of $P\mu$-D is visualized in Fig.3 of the main paper on a dense dataset scene.

5.3 Alternative and Simultaneous training

The AV and the initial pedestrian model are trained simultaneously and alternatively where in the latter case the AV is trained for two epochs for every epoch of training μ. Both models are trained for a total of 14 epochs with the reward $r_\mu = R_{STPN}$. We also report the Avg distance- the average distance travelled by the AV. The *Simultaneous-*μ, ρ has collision rate that is not statistically not different from $OP\mu$ in Table 2. But *Alternative-*μ, ρ has almost twice as many collisions as *Simultaneous-*μ, ρ. Further the alternative *Alternative-*ρ has a similar collision rate when tested with $OP\mu$. This suggests that the AV model learnt by alternative training is poor at collision avoidance. The AV model

Algorithm 2 Learning μ, π and ρ alternatively

```
Initialize $\Theta_1, \beta_1, \xi_1$ randomly.
Initialize learning rates $\alpha_\Theta, \alpha_\beta, \alpha_\xi$
Set $\mu_1 = \mu(\Theta_1)$, $\pi_1 = \pi(\beta_1)$, $\rho_1 = \rho(\xi_1)$
for $j = 1 \ldots J$ iterations do
  Set $\mu_{j,1} = \mu_j$, $\pi_{j,1} = \pi_j$, $\rho_{j,1} = \rho_j$
  for $k = 1 \ldots K_\mu$ iterations do
    Initialize empty set $O = \{\}$
    Sample $s^{\mu}_{j,k}$ from the dataset
    for $m = 1 \ldots M$ iterations do
      Sample $x_0^{j,k,m} \sim \mu_{j,k}(x_0|s^{\mu}_{j,k})$
      Sample $N$ trajectories $\tau^{j,k,m,n} \sim p_\tau(.|x_0^{j,k,m})$ s.t. $a_t^{\pi,j,k,m,n} \sim \pi_j$, $a_t^{\rho,j,k,m,n} \sim \rho_j$, and
      add to $O$
    end for
    Update $\Theta_{j,k+1} = \Theta_{j,k} + \alpha_\Theta \hat{\nabla}_\Theta J_\mu(\Theta)$ using samples in $O$
    Update $\mu_{j,k+1} = \mu(\Theta_{j,k+1})$
  end for
  Set $\mu_{j+1} = \mu_{j,K_\mu}$
  for $k = 1 \ldots K_\pi$ iterations do
    Initialize empty set $O = \{\}$
    Sample $s^{\mu}_{j,k}$ from the dataset
    for $m = 1 \ldots M$ iterations do
      Sample $x_0^{j,k,m} \sim \mu_{j+1}(x_0|s^{\mu}_{j,k})$
      Sample $N$ trajectories $\tau^{j,k,m,n} \sim p_\tau(.|x_0^{j,k,m})$ s.t. $a_t^{\pi,j,k,m,n} \sim \pi_{j,k}$, $a_t^{\rho,j,k,m,n} \sim \rho_j$, and
      add to $O$
    end for
    Update $\beta_{j,k+1} = \beta_{j,k} + \alpha_\beta \hat{\nabla}_\beta J_\pi(\beta)$ using samples in $O$
    Update $\pi_{j,k+1} = \pi(\beta_{j,k+1})$
  end for
  Set $\pi_{j+1} = \pi_{j,K_\pi}$
  for $k = 1 \ldots K_\rho$ iterations do
    Initialize empty set $O = \{\}$
    Sample $s^{\mu}_{j,k}$ from the dataset
    for $m = 1 \ldots M$ iterations do
      Sample $x_0^{j,k,m} \sim \mu_{j+1}(x_0|s^{\mu}_{j,k})$
      Sample $N$ trajectories $\tau^{j,k,m,n} \sim p_\tau(.|x_0^{j,k,m})$ s.t. $a_t^{\pi,j,k,m,n} \sim \pi_{j+1}$, $a_t^{\rho,j,k,m,n} \sim \rho_{j,k}$,
      and add to $O$
    end for
    Update $\xi_{j,k+1} = \xi_{j,k} + \alpha_\xi \hat{\nabla}_\xi J_\rho(\xi)$ using samples in $O$
    Update $\rho_{j,k+1} = \rho(\xi_{j,k+1})$
  end for
  Set $\rho_{j+1} = \rho_{j,K_\rho}$
end for
```

Table 2: An ablation studying the effect of the prior during the training of μ shows that the μ is robust to changes in the prior during training as $OP - \mu$ and $P - \mu$ trained with the priors OP and P respectively, have indistinguishable collision rates (stdev is 0.02).

	$OP - \mu$	$P\mu$-D
#. collisions	**0.22**	0.19
Avg distance	7.8	7.7
π-entropy	0.23	0.29

Table 3: Training the π and μ simultaneously *Simultaneous* results in metrics similar to those of separately trained models. This is confirmed by testing the *Alternative-μ*,*Simultaneous-μ* against the *baseline AV*, and the *Alternative-ρ*,*Simultaneous-ρ* against *OPμ*

	Alternative		
	μ, ρ	μ	ρ
#. collisions	0.41($\pm$0.03)	0.22($\pm$0.02)	0.42($\pm$0.02)
Avg distance	5.4($\pm$0.1)	7.8 ($\pm$0.5)	5.3($\pm$0.2)
π-entropy	0.18($\pm$0.01)	0.23($\pm$0.01)	0.16($\pm$0.01)
	Simultaneous		
	μ, ρ	μ	ρ
#. collisions	0.21($\pm$0.02)	0.20($\pm$0.01)	0.25($\pm$0.02)
Avg distance	7.7($\pm$0.1)	7.9 ($\pm$0.6)	7.5($\pm$0.5)
π-entropy	0.23($\pm$0.01)	0.21($\pm$0.01)	0.16($\pm$0.01)

Alternative-ρ travels 2 meters less than the other models in Table 3. Altogether this suggest that the AV model does not benefit from alternative training with μ. Interestingly the *Alternative-μ* has the same collision rate as $OP\mu$, showing that the ATS model can improve even if the AV model is lacking behind. The simultaneously trained μ and ρ are comparable in collisions and entropy to the $OP\mu$ and the *baseline AV* model. Of higher interest is the low entropy that *Alternative-ρ* and *Simultaneous-ρ* have when tested with $OP\mu$. This could imply that even the AV can learn to place itself such that the pedestrian agent acts as predictably as possible. Further $OP\mu$ receives higher collision rates with *Alternative-ρ* and *Simultaneous-ρ* than *Alternative-μ* and *Simultaneous-μ* respectively. We hypothesize that this could be because in simultaneous or alternative training it is harder to balance hyperparameters to both μ and ρ.

6 Extending μ to model the pedestrian goal distribution

In the main paper we present the model μ that models the pedestrian initial distribution. Here we present an extension of the model that also allows for learning the pedestrian behaviour model's goal location g^{π} distribution μ_g. Since μ_g is optimized with the same reward as μ, this should give the pedestrian initial distribution model more opportunities to enforce collisions. The proposed method is to use the same model architecture for μ_g as for μ, but to use a different prior P_g. It should be noted that in initial experiments we tried to model μ and μ_g in the same network, by simply extending μ with an additional fully connected layer outputting the goal distribution. It was quickly noticed that when π's initial position and goal location were undecided the model struggled to learn as most sampled trajectories resulted in no collisions, even when μ was pretrained. It is clear that μ_g should be conditioned on the sample x_0 rather than on the distribution $P\mu$, as the best goal location depends for example on which side of the AV the pedestrian is initialized. To do so the goal prior P_g is conditioned on x_0, y_0, v_0^{ρ}.

The goal prior $P_g(x|x_0, y_0, v_0^{\rho})$ is found by solving a linear system of inequalities. The goal prior is found by solving for the points $x \in g$ for which the constant velocity prediction of the AV and the pedestrian collide within the given time and speed constraints. More specifically, the points $x \in G$

solve for pedestrian speeds s and collision times t the following system of linear inequalities,

$$\begin{cases} |y_0 + v_0^{\rho} t - x_0 + (x - x_0)st| \leq s_x + s_y \\ 0 \leq t \leq T \\ 0 \leq s \leq \|v_{max}^{\pi}\|, \end{cases} \tag{16}$$

where s_x, s_y are the side lengths of the bounding boxes of the pedestrian and AV. For points $x \in G$ a collision is possible (i.e. there exists s, t that fulfill (16)), the goal prior is assigned the reciprocate distance travelled by the pedestrian to the collision. Finally the goal prior is normalized. If ρ has an initial speed of 0 then $P_g = P$. To summarize the goal prior is given by

$$P_g(x|x_0, y_0, v_0^{\rho}) = \begin{cases} P(x) & \text{if } \|v_0^{\rho}\| = 0 \\ \frac{1}{s_{max}(x)t_{min}(x)} & \text{for } x \in G \\ 0 & \text{otherwise,} \end{cases} \tag{17}$$

where $s_{max}(x)$ is the maximal speed s for the point x that fulfills (17) and $t_{min}(x)$ is the minimal time for collision that fulfills (17) for x. When trained on two scenes, and validated on two scenes the presented goal mode μ_g increased the frequency of collisions from 0.1 to 0.7 with 100 epochs on the validation set. Note in this small experiment the SPL-goal model was used without the Human Locomotion Network (i.e. the STPN-goal model), and μ and μ_g were trained with the reward R_{STPN}.

---

7 List of Mathematical Notations

- α_{β}- learning rate of the parameters β.
- α_{μ}- learning rate of μ.
- α_{ρ}- learning rate of ρ.
- α_{Θ}- learning rate of the parameters Θ.
- α_{ξ}- learning rate of the parameters ξ.
- a_t- a vector of actions taken by μ and ρ at timestep t.
- a_t^{π}- the action taken by the pedestrian at timestep t.
- a_t^{ρ}- the action taken by the AV at timestep t.
- B^{μ}- behavioural constraints for pedestrian inital spatial distribution.
- B^{π}- behavioural constraints for the pedestrian's policy.
- B^{ρ}- behavioural constraints for the AV's policy.
- β- parameters of the parametric policy $\pi(\beta)$.
- b- bias in the AV model.
- b_t- a car's bounding box at timestep t.
- C- number of channels in μ_{Θ}.
- c_t- AV's speed sampled from a neural network.
- D- reciprocal temporal mapping of dynamic objects in s^{μ}.
- D_k- heatmap of pedestrians. The exponential kernel blurred heatmap of pedestrians in D_T.
- D_t- temporal mapping of dynamic objects in s_t^{π}.
- d_t- AV's distance to the closest external car.
- d_t^x- pedestrian agent's distance to the closest external car .
- δ_t- AV's intersection with the sidewalk.
- ϵ- maximal distance to the goal, to attain the goal-reaching related reward.
- f_{μ}- additional terms of the loss J_{μ} that are not related to the number of collisions.
- f_{π}- additional terms of the loss J_{π} that are not related to the number of collisions.
- f_{ρ}- additional terms of the loss J_{ρ} that are not related to the number of collisions.
- γ- reward discount rate.
- g- acceleration of gravity.
- G- the set of possible goal locations that can lead to a collision assuming constant velocity motion.
- g^{π}- the pedestrian's goal location.
- h- the cone of initial locations that lead to a collision assuming that the AV moves at a constant velocity, and that the pedestrian has a maximal speed of $\|v_{max}^{\pi}\| = 3ms^{-1}$.
- I- indicator function, indicating a collision.
- I_g- indicator function, indicating the pedestrian has reached its goal.

- J- policy gradient loss function.
- J_μ- initial pedestrian placement's loss.
- J_π- pedestrian behaviour mode's loss.
- J_ρ- AV's loss.
- K- number of traffic scenes in the dataset/ available in simulator.
- k- iterator.
- λ_*- the scaling of a general reward term R_*.
- λ_a- the scaling of the reward term R_a in r_μ.
- λ_a^π- the scaling of the reward term R_a in r_π.
- λ_d- the scaling of the reward term R_d and R_d^π in r_μ and r_π respectively.
- λ_{dist}- the scaling of the reward term R_{dist}.
- λ_G- the scaling of the reward term I_g.
- λ_g- the scaling of the reward term R_g.
- λ_k- the scaling of the reward term R_k and R_k^π in r_μ and r_π respectively.
- λ_o- the scaling of the reward term R_o.
- λ_ϕ- the scaling of the reward term R_ϕ.
- λ_p- the scaling of the reward term R_p.
- λ_p^ρ- the scaling of the reward term R_p^ρ.
- λ_s- the scaling of the reward term R_s.
- λ_s^ρ- the scaling of the reward term R_s^ρ.
- λ_v- the scaling of the reward term R_v.
- λ_v^ρ- the scaling of the reward term R_v^ρ.
- l_1, l_2- convolutional layers of the neural network μ_Θ.
- L_1, L_2- up-sampled layers of the neural network μ_Θ.
- μ- distribution of pedestrian initial locations.
- μ_g- distribution of pedestrian goal locations.
- μ_Θ- the policy gradient neural network modelling the pedestrian initial distribution in the scene.
- M- number of sampled pedestrian initial locations.
- m- iterator.
- N- number of sampled trajectories.
- n- iterator.
- Ω- the set of allowed values for (τ, x_0, s^μ).
- O- scene occlusion map.
- q- the probability distribution of scenes s^μ.
- P- prior.
- P_g- prior of goal location.
- p- world dynamics and noise.
- p_τ- the probability density function of τ.
- π- pedestrian behaviour model.
- ρ- AV's policy.
- $R = \sum_{t=0}^{T} \gamma^t r(s_t, a_t, s_{t+1})$- a vector of cumulative sums of rewards.
- R_*- A general reward term.
- R_a- A reward term that is 1 if the pedestrian agent collides with the AV.
- R_{coll} reward terms that penalize collisions for the pedestrian agent.
- R_{coll}^ρ the AV's reward terms that penalize collisions.
- R_d- a reward term that encourages the pedestrian to reside on non-zero pixels of D.
- R_d^π- a reward term that encourages the pedestrian to reside on non-zero pixels of D_T.
- R_{dist}- a reward term that encourages the AV to move.
- R_g- a reward term that encourages motion towards the goal g^π.
- R_k- a reward term that encourages the pedestrian to reside on a blurred D.
- R_k^π- a reward term that encourages the pedestrian to reside on D_k.
- R_μ- the one step reward function of μ: the cumulative sum of rewards $R_\mu(x_0) = \sum_t \gamma^t r_\mu(x_t, a_t^\pi, y_t, a_t^\rho)$.
- R_o- A reward term that measures the ratio of pixels of the AV overlapping with the sidewalk..
- R_p- A reward term that is 1 if the pedestrian agent collides with another pedestrian.
- R_p^ρ- A reward term that is 1 if the AV collides with a pedestrian.
- R_{ped}- reward terms that encourage motion in areas frequently visited by pedestrians.
- R_ϕ - a reward term that discourages large unnaturally changes in the pedestrian's pelvis.

- R_ρ- The AV's cumulative discounted reward.
- R_{STPN}- the $STPN$ model reward in Table 2 in the main paper.
- R_s- A reward term that is 1 if the pedestrian agent collides with static objects.
- R_s^ρ- A reward term that is 1 if the AV collides with static objects.
- R_v- A reward term that is 1 if the pedestrian agent collides with an external vehicle.
- R_v^ρ- A reward term that is 1 if the AV collides with an external vehicle.
- $r = [r_\mu, r_\pi, r_\rho]$ - the joint reward vector of the pedestrian initial distribution model μ, the pedestrian behaviour model π and the AV model ρ.
- r_μ- the pedestrian initial distributions' reward function (per timestep).
- r_π- the pedestrian behaviour model's reward function.
- r_ρ- the AV's reward function
- S- scene semantics.
- $s^\mu = (S, D, OP)$- the pedestrian initial location model's state.
- s^π- the pedestrian behaviour model's state.
- s^ρ- the AV's state.
- s_{max} - maximal collision speed for the pedestrian to collide with the AV from location x_0.
- s_t- the AV and the pedestrian behaviour model's state at time t.
- s_t^ρ- the AV's state at time t.
- s_t^π- the pedestrian behaviour model's state at time t.
- s_x- the size of the bounding box of the pedestrian agent.
- s_y- the size of the bounding box of the AV.
- σ_ρ- the AV model's standard deviation.
- t - timestep.
- t_{min} - minimal collision time for the pedestrian to collide with the AV from location x_0.
- T - last timestep of an episode.
- Θ-parameters of μ_Θ.
- τ- the pedestrian and AV's state-action history $(a_0, ..., s_T, a_T, s_{T+1})$.
- v_0^π is the pedestrian agent's initial velocity.
- v_0^ρ is the AV's initial velocity.
- v_{max}^π- pedestrian's maximal possible speed.
- v_{max}^ρ- the AV's maximal possible speed.
- x- a point in the scene.
- x_0- initial position of pedestrian.
- x_t- pedestrian's position at timestep t.
- $x_\perp$ - the otrohogonal projection of a point x in the line $\hat{y}_t$. A point that is on the the AV's future trajectory, and is closest to the point x.
- y_t- AV's position at timestep t.
- $\hat{y}_t$ - the constant velocity prediction of the AV's future motion
- w- learn-able weight in the AV's model.

References

[1] M. Priisalu, C. Paduraru, A. Pirinen, and C. Sminchisescu. Semantic synthesis of pedestrian locomotion. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020.

[2] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.

Paper III

Varied Realistic Autonomous Vehicle Collision Scenario Generation

Maria Priisalu[1(✉)], Ciprian Paduraru[2], and Cristian Smichisescu[1,3]

[1] Lund University, Lund, Sweden
{maria.priisalu,cristian.sminchisescu}@math.lu.se
[2] University of Bucharest, Bucharest, Romania
ciprian.paduraru@fmi.unibuc.ro
[3] Google Research, Zürich, Switzerland

Abstract. Recently there has been an increase in the number of available autonomous vehicle (AV) models. To evaluate and compare the safety of the various models the AVs need to be tested in several diverse safety-critical scenarios. We propose the Adversarial Test Case Generator (ATCG) that differently from previous test case generators allows for the generation of realistic collision scenarios with varied AV and pedestrian behaviour models, on varied scenes and with varied traffic density. Given a top-view image and the semantic segmentation of a traffic scene, the ATCG learns to place multiple AVs and goal-reaching pedestrians in the scene such that collisions occur. Pedestrians in previous multi-agent traffic scenario generation works are confined to unrealistic behaviours such as seeking collisions with the AV or ignoring the AV. Although such scenarios with multiple suicidal pedestrians are collision prone it is unlikely in reality that all pedestrians act abnormally. In realistic collision scenarios the generated pedestrians' behaviours must resemble real pedestrians. The ATCG is a team of Reinforcement Learning (RL) agents and can be easily extended with additional RL agents to produce more complex scenes allowing for advanced AVs to be tested.

Keywords: Autonomous Vehicle · AV Testing · Multi Agent Reinforcement Learning

1 Introduction

The extensive work on autonomous vehicles (AV) [1–3] has brought about a need for testing AVs [4] in safety-critical situations (such as collisions and near-collisions) to ensure the safety of all traffic participants in deployment. Safety critical scenarios are not frequent in traffic, therefore data gathering is extremely time-consuming as well as unethical. Testing an AV on a dataset of collision scenarios [5–8] is not sufficient since any safety critical scenario dataset is inherently limited in variability. To alleviate these issues safety-critical scenarios, in particular collisions between AVs and pedestrians, can be generated [9–33]. The majority

R. Gade et al. (Eds.): SCIA 2023, LNCS 13886, pp. 354–372, 2023.
https://doi.org/10.1007/978-3-031-31438-4_24

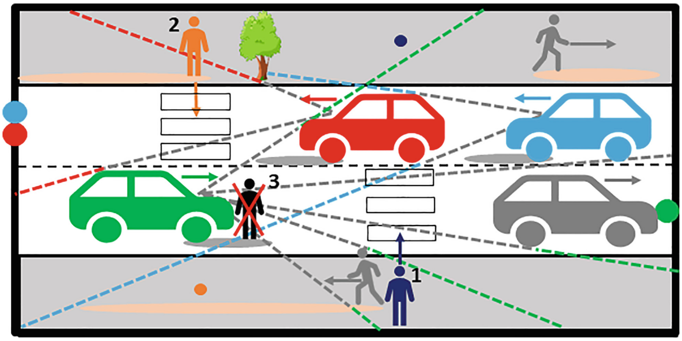


Fig. 1. The ATCG sees the top view projection of a 3D point cloud of the scene with external cars and pedestrians (in gray) and their initial velocities (arrows). The ATCG places out the coloured pedestrians and AVs in the scene with goals (marked as circles in the colour of the AV or pedestrian) and initial velocities (arrows). Dotted lines show areas occluded for the AVs. The dark blue pedestrian #1 is placed out such that it is occluded for the green AV by an external pedestrian. The orange pedestrian #2 is initialized in an area that is often frequented by pedestrians (light orange ground) and occluded for the blue AV. The blue AV cannot see that the red AV is braking for the orange pedestrian #2. Pedestrians (pedestrian #3) cannot be initialized within the braking distance of the AVs (gray area under the AV's wheels). (Color figure online)

of existing test case scenario generators [9–33] make scenario generation low parametric by either assuming heuristic motion models for pedestrians (i.e. constant velocity or adversarial pedestrians) or by making particular confining assumption about the geometry of the traffic scene (assuming a straight street, or a crossing of a particular shape, etc.). As a first step [34] propose a visual single pedestrian and single AV test case generator that generalizes across scenes without confining assumptions about the scene geometry and allows for test case generation with various pedestrian behavior models. We propose the Adversarial Test Case Generator (ATCG) that generalizes [34] to multi-agent scenario generation, ensuring the realism of the traffic scene and all of its participants. The ATCG is a team of Reinforcement Learning (RL) agents that learn where in a scene to place N pedestrians and M AVs to generate collisions. The proposed framework allows for the interchangeability of the pedestrian and the AV models and can generate realistic and diverse collision scenarios for AVs. We are the first to our knowledge to propose a multi-agent AV collision scenario generator that can utilize any goal-driven state-of-the-art pedestrian forecasting model, such as the collision-avoiding, semantically reasoning, pose articulated model [35].

Most AV models are either modular [3], with components for perception, planning and vehicle control, or end-to-end AV models [1,2] that directly map sensor data to vehicle control. We aim to test the full pipeline of modular AVs and end-to-end AV models by generating scenarios in a 3D environment. Modeling scenarios in 3D allows for the simulation or augmentation of sensor data [36–41] as well as the evaluation of the AV's trajectory after actuation of the AVs control.

Traffic is highly dependent on semantics [42,43]. The AVs and pedestrians plan their future trajectory based on the semantic objects around them [44,45], see Fig. 1. To generate collisions the ATCG must be able to predict the future motion of the AVs and pedestrians, therefore the ATCG observes the scene semantics. The ATCG utilizes scene-semantics dependent priors which increase sample efficiency in learning without making assumptions about the geometry of the scene and its participants [11–33].

Collision scenario generation is an optimization problem maximizing the number of collisions over a set of scenario descriptive parameters. The AV can either be treated as a black box [17–19] or white box [46,47]. White box treatment provides strong gradients but limits the interchangeability of the AV model and assumes that the AV and the environment are differentiable. The ATCG models the AV as a black box to allow any AV model to be tested and to avoid assumptions about the AV's sensory inputs and dynamics. In literature the black box optimization methods used to generate collision scenarios are Bayesian Optimization (BO) [17], Genetic Algorithms (GA) [18–23] and RL [30–32,34]. BO [48] suffers from the curse of dimensionality and therefore cannot be applied to visual problems. RL takes gradient steps and is therefore more sample efficient than GA. The ATCG is modeled by visual RL agents. When the AV is an RL agent the collision scenario generation problem becomes a Multi-Agent Reinforcement Learning (MARL) game [49].

The ATCG is composed of the Adversarial Autonomous Vehicle Initial Location Agent (AVILA) which places out AVs in a given scene and the Adversarial Pedestrian Initial Location Agent (APILA) which places out pedestrian agents. The ATCG can be used to augment an existing dataset $\mathcal{D} = (\mathcal{D}_c, \mathcal{D}_p)$, (gray cars and pedestrians in Fig. 1), of pre-recorded car and pedestrian trajectories respectively $\mathcal{D}_c$ and $\mathcal{D}_p$. Utilizing a team of RL agents to generate test scenarios allows us to provide different task-specific reward functions to each agent. Pedestrians and AVs have significantly different behaviour in traffic and this is reflected in the reward functions of the AVILA and the APILA. The AVILA should place AVs such that the AVs behave realistically until a collision occurs. Therefore the AVILA is given a positive reward for collisions and is otherwise rewarded according to the AV's reward function. The APILA's reward is analogous to the AVILA's. The ATCG can be extended with additional RL agents that further constrain the AV, for example by occluding its vision. In difference to previous work [34] we pose the problem of collision-prone test case generation as a teamed MARL game, that is easily extendable to target other risk-based behaviours and allows for realistic-looking scenario generation. We show that natural-looking collision-prone traffic scenarios can be found for multiple collision-avoiding AVs and pedestrians.

1.1 Related Work

AV test scenarios can be generated by adversarially perturbing data, possibly in the latent space [24–29], to produce collisions [33,46,50]. However, the variability of the generated test cases is limited by the dataset's diversity. Virtual

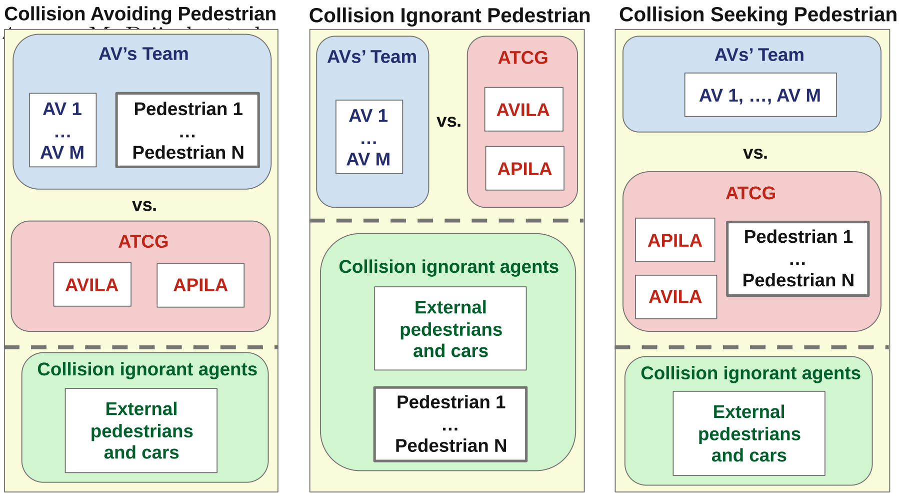

Collision Avoiding Pedestrian
AV's Team
AV 1
...
AV M
Pedestrian 1
...
Pedestrian N
vs.
ATCG
AVILA
APILA
Collision ignorant agents
External pedestrians and cars
Collision Ignorant Pedestrian
AVs' Team
AV 1
...
AV M
vs.
ATCG
AVILA
APILA
Collision ignorant agents
External pedestrians and cars
Pedestrian 1
...
Pedestrian N
Collision Seeking Pedestrian
AVs' Team
AV 1, ..., AV M
vs.
ATCG
APILA
AVILA
Pedestrian 1
...
Pedestrian N
Collision ignorant agents
External pedestrians and cars

Fig. 2. Different teams of the Multi-Agent Game are formed when the pedestrians' behaviour is collision avoiding, collision ignorant or collision seeking.

reality can be used to record pedestrians' poses in near collision scenarios [51–53] producing realistic data, but human-in-the-loop data gathering is not scalable. There are a number of component wise test case generators [54–59] for modular AVs, specifically [55–60] test the visual systems only, and [54] tests the planning and control components of an AV. It is not straightforward how to combine these component-wise test generators to evaluate the full pipeline of the AV from sensor input to control. AV test case generation is still a young research topic with a number of unanswered questions, such as suitable metrics [61], therefore the scenario generator's components (such as the pedestrian behaviour model, dynamics models and sensor-modelling) should be interchangeable until standards are established in the field.

2 The Multi-agent Game

The ATCG is a team of RL agents with the common objective to increase the number of collisions between the AVs and the pedestrians. The ATCG is composed of the APILA with policy μ and the AVILA with policy ν. The ATCG's opposing team, the AVs' team, aims to decrease the number of collisions. The AV's team is composed of the M goal-reaching AVs governed by policy ρ. The N goal-reaching pedestrian agents with policy π can either be in the ATCG if π is collision seeking, on the AVs' team if π is collision avoiding, or on neither of the teams if π is collision ignorant, as seen in Fig. 2. The pedestrian agents and AVs can be any goal-driven behaviour models.

The pedestrian agents and the AVs are given rule-based goal locations. In the future, the ATCG could be extended with an agent that chooses the goal

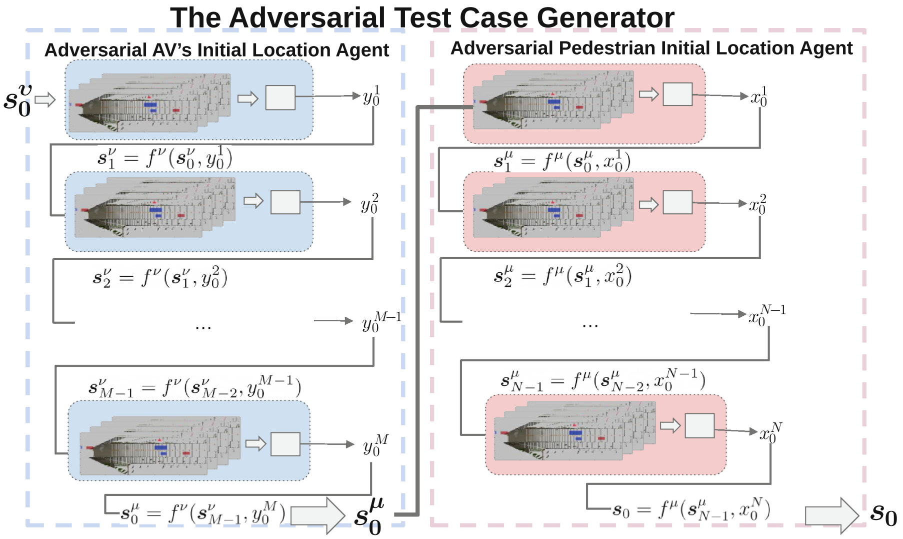

The Adversarial Test Case Generator
Adversarial AV's Initial Location Agent
Adversarial Pedestrian Initial Location Agent
s_0^ν
y_0^1
$s_1^\nu = f^\nu(s_0^\nu, y_0^1)$
y_0^2
$s_2^\nu = f^\nu(s_1^\nu, y_0^2)$
...
y_0^{M-1}
$s_{M-1}^\nu = f^\nu(s_{M-2}^\nu, y_0^{M-1})$
y_0^M
$s_0^\mu = f^\nu(s_{M-1}^\nu, y_0^M)$
s_0^μ
x_0^1
$s_1^\mu = f^\mu(s_0^\mu, x_0^1)$
x_0^2
$s_2^\mu = f^\mu(s_1^\mu, x_0^2)$
...
x_0^{N-1}
$s_{N-1}^\mu = f^\mu(s_{N-2}^\mu, x_0^{N-1})$
x_0^N
$s_0 = f^\mu(s_{N-1}^\mu, x_0^N)$
s_0

Fig. 3. Given the visual scene state s_0^ν AVILA places out M AVs in the scene sequentially at locations $y_0^1 ... y_0^M$. After that s_0^μ is given to APILA, and APILA chooses sequentially the initial locations $x_0^1 ... x_0^N$ of the N pedestrians, producing the initial scene state s_0.

locations of the pedestrian agents with the objective of maximizing collisions. Similarly, an AV's goal location-picking agent could be added to the ATCG. Further, an agent that places out parked cars could be added to the ATCG, with the objective of obstructing the AVs' view of the pedestrians. Or possibly an agent responsible for placing out bus shelters in semantically plausible positions such that the AVs' view of the pedestrians is occluded (increasing the probability that the AV misses the pedestrians, causing a collision). Each agent in the ATCG can have an objective different from that of its teammates (*i.e.*maximizing collisions vs. occlusions). Our proposed ATCG framework is easily extendable by the addition of teammates.

3 The Adversarial Test Case Generator

Given a voxelized scene s_0^ν containing external pedestrians and vehicles $\mathcal{D}$ the AVILA places out M AVs sequentially as shown in Fig. 3. The AVILA is an RL agent that chooses an initial location y_0^m for the m-th AV, $1 \leq m \leq M$ according to its policy $y_0^m \sim \nu(.|s_{m-1}^\nu)$, where s_{m-1}^ν is the scene description containing $m-1$ AVs. The AV with position y_0^m, goal location $g_0^{\rho,m}$ and initial velocity $v_{-1}^{\rho,m}$ chosen as described in Sect. 3.1 is added to s_{m-1}^ν forming s_m^ν. We denote the scene containing all the M AVs as the APILA's initial state s_0^μ. The APILA samples the n-th pedestrian agent's, $1 \leq n \leq N$, initial position x_0^n from its policy $x_0^n \sim \mu(.|s_{n-1}^\mu)$. The pedestrian with the initial position x_0^n, goal

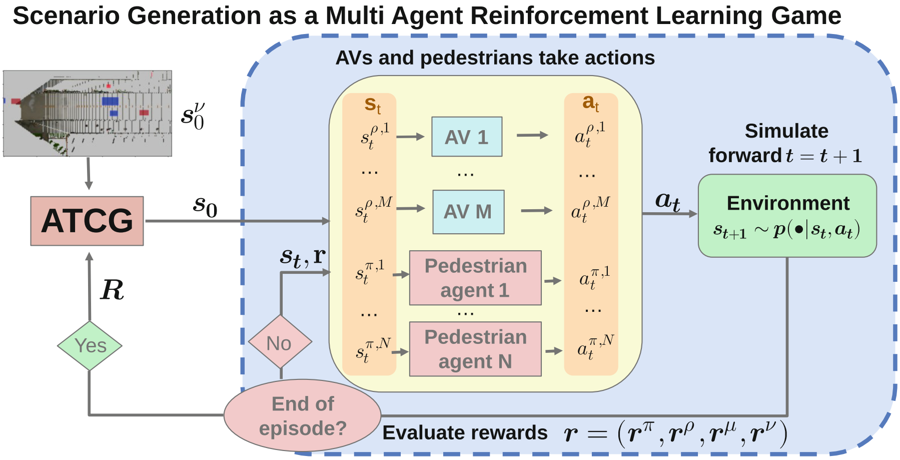


Fig. 4. The order of actions of the MARL game: The ATCG places out pedestrians and AVs in the scene at timestep 0, producing $\boldsymbol{s}_0$. The pedestrian agents and AVs take actions $\boldsymbol{a}_t$ simultaneously conditioned on the state $\boldsymbol{s}_t$. The world dynamics p simulate the scene forward into state $\boldsymbol{s}_{t+1}$. The reward function $\boldsymbol{r}(\boldsymbol{s}_t,\ \boldsymbol{a}_t,\boldsymbol{s}_{t+1})$ is evaluated. If $\boldsymbol{s}_{t+1}$ is a terminal state the reward is provided to the ATCG, otherwise $t = t + 1$.

location $g_0^{\pi,n}$ and velocity $v_{-1}^{\pi,n}$ chosen as described in Sect. 3.2 is added to $\boldsymbol{s}_{n-1}^{\mu}$ forming $\boldsymbol{s}_n^{\mu}$.

We define the state $\boldsymbol{s}_0$ as the state of the scene at timestep $t = 0$ containing all of the AVs and pedestrian agents. The state $\boldsymbol{s}_t$ is a vector of the states of the pedestrians and AVs $\boldsymbol{s}_t = (s_t^{\pi,1}, ..., s_t^{\pi,N}, s_t^{\rho,1}, ..., s_t^{\rho,M})$, where $s_t^{\pi,n}$ is the n-th pedestrian agent's state and $s_t^{\rho,m}$ is the m-th AV's state at timestep t. After $\boldsymbol{s}_0$ is formed, all of the pedestrian agents and the AVs simultaneously choose each an action $a_t^{\pi,n} \sim \pi(.|s_t^{\pi,n})$ and $a_t^{\rho,m} \sim \rho(.|s_t^{\rho,m})$ for $0 \leq t < T$ until the end of the episode, as shown in Fig. 4. The motion of all pedestrians and vehicles is simulated forward by the unknown world dynamics p. That is $\boldsymbol{s}_{t+1} \sim p(.|\boldsymbol{s}_t, \boldsymbol{a}_t)$, where $\boldsymbol{a}_t = (a_t^{\pi,1}, ..., a_t^{\pi,N}, a_t^{\rho,1}, ..., a_t^{\rho,M})$ is a vector of the actions taken by the pedestrian agents and AVs. The episode ends when $t = T - 1$ or when a pedestrian agent collides with a vehicle. The state transitions are evaluated by the pedestrian's $r^\pi(s_t^{\pi,n}, a_t^{\pi,n}, s_{t+1}^{\pi,n})$, AV's $r^\rho(s_t^{\rho,m}, a_t^{\rho,m}, s_{t+1}^{\rho,m})$, the APILA's $r^\mu(s_t^{\pi,n}, a_t^{\pi,n}, s_{t+1}^{\pi,n})$ and the AVILA's $r^\nu(s_t^{\rho,m}, a_t^{\rho,m}, s_{t+1}^{\rho,m})$ reward functions. The rewards of the triplet $(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_{t+1})$ are gathered in a vector $\boldsymbol{r}_t$ of size $2(M + N)$. At the end of the episode the cumulative APILA's reward for the n-th pedestrian $R_n^\mu = \sum_{t=0}^{T-1} \gamma^t r^\mu(s_t^{\pi,n}, a_t^{\pi,n}, s_{t+1}^{\pi,n})$ and the AVILA's reward for the m-th AV $R_m^\nu = \sum_{t=0}^{T-1} \gamma^t r^\nu(s_t^{\rho,m}, a_t^{\rho,m}, s_{t+1}^{\rho,m})$ are given to ATCG. The cumulative rewards for all agents are gathered in a vector $\boldsymbol{R}$ of size $2(M + N)$. Details in Sect. 3.1, Sect. 3.2.

3.1 The Adversarial Autonomous Vehicle Initial Location Agent

The AVILA's policy ν models the AV's initial location distribution and it is the product of its prior ν^p and the parametric policy ν_Θ. The ν^p informs scene semantic and geometric dependent prior knowledge and prevents zero gradients in early learning (due to lack of collisions between the AVs and pedestrian agents). The AVILA's action space is the set of valid initial locations for the AV in the scene. Any location that would make the AV collide in the first timestep is set to zero in the AVILA's prior ν^p. The AVILA takes M actions (i.e. chooses initial locations for M AVs). For $m < M$, the AVILA's state is updated as $\boldsymbol{s}^\nu_m = f^\nu(\boldsymbol{s}^\nu_{m-1}, y^m_0)$ by a deterministic function f^ν, and the state transition's reward is 0. The last state of the AVILA is the last state of the scene $\boldsymbol{s}_T$. The AVILA's trajectory is

$$(\boldsymbol{s}^\nu_0, y^1_0) \rightarrow_{r=0} (\boldsymbol{s}^\nu_1, y^2_0) \rightarrow_{r=0} ...(\boldsymbol{s}^\nu_{M-1}, y^M_0) \rightarrow_{r=\sum_{m=1}^M R^\nu_m} (\boldsymbol{s}_T, .). \tag{1}$$

On the last state transition, AVILA is rewarded for the full trajectories of all the AVs. The AVILA's policy $\nu(.|\boldsymbol{s}^\nu_0) = \nu^p \nu_\theta(.|\boldsymbol{s}^\nu_0)$ is found by solving

$$\max_\Theta \mathbb{E}\left[\sum_{m=1}^{M} c^\nu_H H(\nu^p_{m-1}\nu_\Theta(.|\boldsymbol{s}^\nu_{m-1})) + \sum_{t=0}^{T-1} \gamma^t r^\nu(s^{\rho,m}_t, a^{\rho,m}_t, s^{\rho,m}_{t+1})\right], \tag{2}$$

where entropy H is added to the loss to increase exploration [62], and $c^\nu_H = 0.1$ is a constant. The expectation in (2) is taken over all $\boldsymbol{s}_t, \boldsymbol{a}_t$ for all $0 \leq t \leq T-1$.

AVILA's State. The state $s^\nu_m = (S, D^\nu_m, \nu^p_m)$ contains a top view projection $S \in \mathbb{R}^{(128,256,12)}$ of static objects in the scene labeled with RGB colors and semantic segmentation labels, a dynamic occupancy map $D^\nu_m \in \mathbb{R}^{(128,256,4)}$, and the prior $\nu^p_m \in \mathbb{R}^{(128,256)}$. The D^ν_m contains the future occupancy (predicted with constant velocity) of all external pedestrians and cars in $\mathcal{D}$ and the $(m-1)$ AVs that have been placed out. The first two channels of $\mathcal{D}$ are the reciprocal time of predicted occupancy for all vehicles and pedestrians respectively, drawing AVILA's attention to locations occupied in the near future. The third and fourth channels are indicators of the occupancy of vehicles and pedestrians in timestep $t = 0$. For the first AV the prior $\nu^p_1(z)$ encourages initial locations that lead to long AV trajectories within the scene. The unnormalized $\nu^p_1(z)$ is the distance from a valid location z to the end of the scene (shown as the distance between the AV at y_0 and the black circle g^ρ in Fig. 5) along the expected motion v_z at z (according to $\mathcal{D}_c$). For all other AVs the prior ν^p_m encourages initial locations that occlude the first AV's point of view, by prioritizing locations in the proximity of the first AV and within its field of view. For a position z that is a valid initial location for the m-th AV, where $m > 1$, $\nu^p_m(z)$ is the reciprocal distance to the first AV's location from z. A factor 1/2 reduces $\nu^p_m(z)$ at all points z outside of the first AV's field of view. An AV at a position z is given the point's expected velocity v_z as the initial velocity $v^{\rho,m}_{-1}$ and the edge point of the scene along the direction v_z from y^m_0 as the goal location $g^{\rho,m}_0$, see Fig. 5.

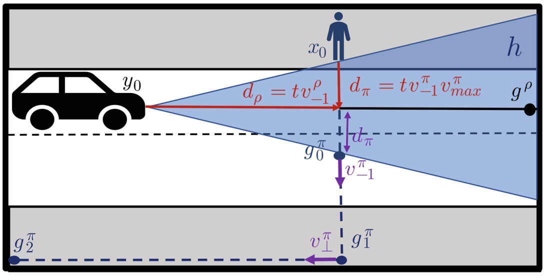


Fig. 5. The cone (in blue) of initial positions h from which the pedestrian can reach the vehicle's initial motion v^ρ_{-1}, such that both the AV at y_0 and the pedestrian at x_0 can reach the same position $x_0 + d_\pi = y_0 + d_\rho$ by time t, with a maximal pedestrian speed of $v^\pi_{max} = 3ms^{-1}$. A black circle is the AV's goal location: the most distant point in the scene on the AV's trajectory. A pedestrian initialized at a position x_0 is given an initial goal g^π_0 that is x_0's reflection in the AV's trajectory. The pedestrian's initial velocity v^π_{-1} is given in the direction of g^π_0. The pedestrian's next goal g^π_1 is the last valid point in the scene along v^π_{-1}. After reaching g^π_1 the pedestrian is given a goal g^π_2 that is in an orthogonal direction $v^\pi_\perp$ to v^π_{-1} (Color figure online)

AVILA's Reward Function. AVILA and the AV have similar reward functions r^ν and r^ρ but with different reward weights $\boldsymbol{\lambda}^\nu \neq \boldsymbol{\lambda}^\rho$. The AVILA's task is to initialize the AV such that the AV collides with pedestrians but otherwise performs as well as possible according to its reward function r^ρ (i.e. desired AV behaviour). The reward $r^\nu = r^\nu_{coll.} + \lambda^\nu_{s.w.} r^\nu_{s.w.} + \lambda^\nu_{dist.} r^\nu_{dist.} + \lambda^\nu_{speed} r^\nu_{speed}$ consist of a collision evaluating reward $r^\nu_{coll.}$, a reward penalizing overlap with sidewalk $r^\nu_{s.w.}$, a reward promoting distance travelled $r^\nu_{dist.}$ and a reward penalizing speeding r^ν_{speed}, where $\lambda^\nu_{s.w.} = -0.1, \lambda^\nu_{dist.} = 0.01, \lambda^\nu_{speed} = -0.1$ are weights. The reward $r^\nu_{coll.} = \lambda^\nu_v r^\nu_v + \lambda^\nu_p r^\nu_p + \lambda^\nu_o r^\nu_o$ consist of indicator functions $r^\nu_v, r^\nu_p, r^\nu_o$ that are 1 if the m-th AV at the position $y^m_t + v^{\rho,m}_t$, (where $\Delta t = 1$) in the frame $t+1$ collides with other vehicles, pedestrians, or any objects in S respectively, weighted by $\lambda^\nu_v = -2, \lambda^\nu_p = 2, \lambda^\nu_o = -2$. The reward $r^\nu_{s.w.}$ is the m-th AV's relative intersection with the sidewalk. The relative distance traveled is rewarded by $r^\nu_{dist} = \|y^m_t + v^{\rho,m}_t - y^m_0\|/(t+1)v^\rho_{max}$, where v^ρ_{max} =70km/h is the AV's maximal speed. Speeding above v^ρ_{ref} =40km/h is penalized by $r^\nu_{speed} = max(\|v^{\rho,m}_t\| - v^\rho_{ref}, 0)$. The AV is considered dead and it obtains a zero reward after a collision.

The Model Architecture. The model ν_Θ is a two-layered convolutional neural network, with input $\boldsymbol{s}^\nu_m$- a multi-dimensional image. The model architecture follows [34] with the addition of semantic channel normalization in $\boldsymbol{s}^\nu_m$ to balance the bias of from heavily represented semantic classes (like road) and give more initial attention to less observed classes (like pole). Each semantic channel of $\boldsymbol{s}^\nu_m$ is normalized to sum to 1, resulting in the normalized $\tilde{\boldsymbol{s}}^\nu_m$, which is passed through the first convolutional layer (conv(3 × 3 × C × 1)→max-pool($2 \times 2 \times 1$)) resulting in the first layer's output l_1. The second convolutional layer (conv($2 \times 2 \times 1$)→max-pool(2×2)→ ReLU), is applied to l_1, producing an output l_2. Bilinear interpolation is used to upsample l_1 and l_2 to the size (128,256) producing L_1 and L_2. Finally $\nu_\Theta(.|\tilde{\boldsymbol{s}}^\nu_m) = \text{softmax}(L_1 + L_2)$. During training REINFORCE [62] with ADAM [63] optimizer is used.

3.2 The Adversarial Pedestrian Initial Location Agent

APILA's policy μ is the product of the pedestrian initial location prior μ^p and the parametric μ_ω (same model architecture as AVILA, see Sect. 3.1) that models the pedestrian initial location distribution. The APILA's action space is the set of valid initial pedestrian locations in the scene, invalid locations have a prior value of 0. The APILA takes N actions (i.e. chooses initial locations for N pedestrians). For $1 \leq n \leq N-1$, APILA's state is updated as $\boldsymbol{s}^\mu_n = f^\mu(\boldsymbol{s}^\mu_{n-1}, x^n_0)$ by a deterministic function f^μ and the state transition's reward is 0. The last state of the APILA is the scene's last state $\boldsymbol{s}_T$ and the final state transition's reward is $\lambda^\mu_\sigma r^\mu_\sigma(\boldsymbol{x}_0) + \sum_{n=1}^N R^\mu_n$, where the additive reward component $r^\mu_\sigma(\boldsymbol{x}_0)$, with weight $\lambda^\mu_\sigma = 0.1$, is the sum of the standard deviations of the pedestrians' initial locations $\boldsymbol{x}_0 = (x^1_0, ..., x^1_N)$ coordinates. APILA's policy $\mu(.|\boldsymbol{s}^\mu_0) = \mu^p \mu_\omega(.|\boldsymbol{s}^\mu_0)$ is found by solving

$$\max_\omega \mathbb{E}\left[\lambda^\mu_\sigma r^\mu_\sigma(\boldsymbol{x}_0) + \sum_{n=1}^{N} c^\mu_H H(\mu^p_{n-1}\mu_\omega(.|\boldsymbol{s}^\mu_{n-1})) + \sum_{t=0}^{T-1} \gamma^t r^\mu(s^{\pi,n}_t, a^{\pi,n}_t, s^{\pi,n}_{t+1})\right], \tag{3}$$

where H is entropy, $c^\mu_H = 0.1$ is a constant, and where the expectation is taken over all $\boldsymbol{s}_t, \boldsymbol{a}_t$ for all $0 \leq t \leq T-1$.

APILA's State. The state $s^\mu_n = (S, D^\mu_n, \mu^p_n)$ where D^μ_n is a dynamic occupancy map, and μ^p_n is the prior. The map D^μ_n is formed analogously to D^μ_n predicting the motion of all AVs, the $(n-1)$ pedestrians that have been placed out, and the external pedestrians and cars in $\mathcal{D}$. The prior is $\mu^p_n = (G(\sigma)^{n-1} * \mu^{TTC}_n)\mu^{\mathcal{D}}$, where μ^{TTC}_n is smoothed by convolution with a 2D Gaussian filter with $\sigma = 15$, and $\mu^{\mathcal{D}}$ is the estimated pedestrian density heatmap. The reciprocal time to collision (TTC [64]) map μ^{TTC}_n is zero in positions that would lead the pedestrian to a collision on the first frame, within the AV's braking distance, and on the AV's constant velocity future trajectory, to avoid trivial collisions. The set of points that can lead to a collision assuming AV's constant motion are denoted h see Fig. 5. For $z \in h, \mu^{TTC}_n(z) = \|z - y^1_0\|^{-1}$ is the reciprocal distance to the first AV. For all other valid points $z' \notin h$ the prior is smaller, $\mu^{TTC}_n(z') = ||z' - y^1_0||^{-2}$. The μ^{TTC}_n has higher values in locations that could lead to a collision quickly. APILA cannot control pedestrians' trajectories beyond the first timestep so it is preferable for the pedestrians to be placed near a possible collision, to increase the likelihood of a collision. The objective (3) promotes all of the pedestrians to be placed near the first AV, but realistically it is unlikely for an AV to be surrounded by pedestrians. Therefore we smooth the prior to encourage

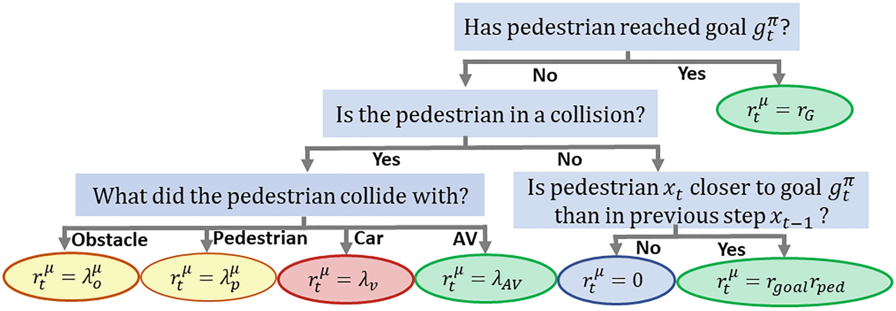

Has pedestrian reached goal g_t^π?
No
Yes
Is the pedestrian in a collision?
$r_t^\mu = r_G$
Yes
No
What did the pedestrian collide with?
Is pedestrian x_t closer to goal g_t^π than in previous step x_{t-1} ?
Obstacle
Pedestrian
Car
AV
No
Yes
$r_t^\mu = \lambda_o^\mu$
$r_t^\mu = \lambda_p^\mu$
$r_t^\mu = \lambda_v$
$r_t^\mu = \lambda_{AV}$
$r_t^\mu = 0$
$r_t^\mu = r_{goal} r_{ped}$

Fig. 6. APILA's reward: Red ovals indicate undesired and lethal pedestrian behaviour (a penalizing reward), green indicates desired pedestrian behaviour (an increasing reward), and yellow undesired (decreasing reward) but not lethal pedestrian behavior. See Fig. 5 (Color figure online)

diversity in the placement of pedestrians while ensuring that collisions occur. The pedestrian density map is

$$\mu^{\mathcal{D}}(z) = \log\left\{\frac{1}{b}\sum_{x\in\mathcal{D}_p}\exp\{-\|z-x\|\}\right\}, \tag{4}$$

where $b = 10^{-4}$ is the bandwidth of the exponential kernel. The $\mu^{\mathcal{D}}$ provides a data-driven prior for locations where pedestrians are more likely to occur.

APILA's Reward Function. APILA's reward shown in Fig. 6 evaluates the n-th pedestrian's behaviour at timestep t, where $\lambda^\mu_{AV} = 2N$, $\lambda^\mu_v = 0$, $\lambda^\mu_p = -0.1$, $\lambda^\mu_o = -0.02$ and $\lambda^\mu_G = 0.001/N$. N is a factor in λ^μ_{AV} because APILA should be motivated to find locations where pedestrians collide with the AV even if this implies that no further reward can be attained from any of the pedestrian agents. Similarly N^{-1} is a factor in λ^μ_G. The reward term $r^\mu_{ped.} = (1+\lambda^\mu_{\mathcal{D}}\mu^{\mathcal{D}})(1+\lambda^\mu_{ped.occ.}r^\mu_{ped.occ.})$, where $r^\mu_{ped.occ.}$ is an indicator function that is 1 if the position $x^n_t + v^{\pi,n}_t$ coincides with an external pedestrian's previous occupancy $\mathcal{D}_p$, and where $\lambda^\mu_{ped.occ.} = 0.01$ and $\lambda^\mu_{\mathcal{D}} = 0.01$. Steps taken towards the goal are rewarded by $r^\mu_{goal} = 1 + \lambda^\mu_g r^\mu_g$, where $r^\mu_g = (l^\pi_{max})^{-1}(|x^n_t + v^{\pi,n}_t - g^{\pi,n}_t| - |x^n_t - g^{\pi,n}_t|)$ and l^π_{max} is the step length of the pedestrian at v^π_{max}, and $\lambda^\mu_g = 0.02/N$. The APILA rewards the pedestrian for taking steps towards its goal, for moving in areas visited by pedestrians, for colliding with AVs, and penalizes all other collisions.

The Pedestrian Model. The competitive pedestrian forecasting model CARLA Semantic Pedestrian Locomotion (SPL)-goal from [35] is used. The SPL is semantically reasoning, collision avoiding and goal-reaching. The SPL is articulated by the Human Locomotion Network (HLN). The SPL enforces human motion like dynamics. In this work, differently from [34,35], the pedestrian agents are not considered dead when reaching a goal but instead are given a new goal, Fig. 5.

The pedestrian's state $s_t^{\pi,n}$ consists of a 5 m × 5 m local crop $S(x_t^n)$ of S, a local crop $D_t^{\pi,n}(x_t^n)$ of the dynamic occupancy map $D_t^{\pi,n}$, the history of the SPL's actions $a_{t-1}^{\pi,n}, ..., a_{t-12}^{\pi,n}$, the previous hidden state of HLN h_{t-1}^n encoding the pedestrian's pose, displacement to the closest vehicle $d_t^{\pi,n}$ from x_t^n, and displacement $(g_t^{\pi,n} - x_t^n)$ to the goal location $g_t^{\pi,n}$. The dynamic occupancy map $D_t^{\pi,n}$ contains the past and predicted future (assuming constant motion) time of occupancy of vehicles and pedestrians. In difference to [34,35] occluded objects and objects outside of the pedestrian's field of view are not observed in $s_t^{\pi,n}$.

The pedestrian reward is additive APILA's reward Fig. 6, with negative weights $\lambda_v^\pi = -2, \lambda_{AV}^\pi = -2$ and the addition of r_ϕ^π to the non-lethal nodes of the bottom row of Fig. 6 to penalize non-smooth motions. The reward $r_\phi^\pi = \lambda_\phi^\pi \min(\max(|\phi_t^n|, 0) - 1.2, 2)$, where $\varDelta_\phi$ is the average yaw of the joints of the lower body of the n-th pedestrian agent and $\lambda_\phi^\pi = -0.0001$.

The AV Model. There is extensive work done on AVs [1–3], and in the future, we plan to extend our work to state-of-the-art AV models. We provide an interface to CARLA [65], available at https://github.com/MariaPriisalu/atcg, to allow for the use of the ATCG in the CARLA simulator. We use a speed-controlling AV model that follows the trajectories of external vehicles $\mathcal{D}_c$. Note that because the pedestrians are not initialized within the braking distance of the AV, the AV can always avoid collisions by standing still. A simple AV model is used to avoid duplicating research of the AV community and to avoid making assumptions about the AV model's sensory inputs. The AV is a policy gradient agent with the state $s_t^{\rho,m}$. The AV's state consists of the distance to the closest pedestrian $d_t^{\rho,m}$, the distance to the closest car $\delta_t^{\rho,m}$ and the ratio of intersection between the AV and the sidewalk $f_t^{\rho,m}$. The AV chooses its action as $a_t^{\rho,m} \sim \mathcal{N}(\text{sigmoid}(w^\rho s_t^{\rho,m} + b^\rho, \sigma_\rho)$, where w^ρ, b^ρ are learnt with REINFORCE [62]. The AV moves in the direction $v_t^{\rho,m}$ that is the direction of expected motion for the vehicle's current location $y_t^{\rho,m}$. The AV's speed $v_{max}^\rho a_t^{\rho,m}$ is the product of the maximal AV's speed v_{max}^ρ and the AV's action $a_t^{\rho,m}$. The AV has perfect dynamics. The AV model has the same reward function as AVILA but with weights $\lambda_{s.w.}^\rho = -0.1, \lambda_{dist}^\rho = 0.01, \lambda_{speed}^\rho = -0.1, \lambda_v^\rho = -1, \lambda_p^\rho = -2, \lambda_o^\rho = -2$, σ_ρ=0.1.

4 Experiments

All models are trained on the CARLA dataset [35] with the addition of the dense dataset from [34] to the training set. The dataset contains 3D reconstructions (with semantic and RGB labels) of scenes constructed from images gathered onboard a car. The training set consists of 102 scenes and the validation set of 53 scenes of Town 1. The test set consists of 150 scenes of Town 2. The scenes are of size 51.2 m × 25.6 m, with a voxel size $(20\,\text{cm})^3$. When nothing else is stated the *AV base model* from [34] is used to model AVs' policy ρ in evaluations. During the evaluation, the pedestrian and the AV models act according to the mode of their policies, but the ATCG samples actions from its policy. All models are trained in Tensorflow, with a learning rate of 0.1 for the ATCG and the pedestrian models, and a learning rate of 0.017 for the AV. A discounting factor $\gamma = 0.99$ is used, and an entropy weight of $c_H^{\mu} = 0.1, c_H^{\nu} = 0.1$ for the ATCG. When training only the ATCG a trajectory length of $T = 100$ is used, and a trajectory length of 30 is used when the AV is trained alternatively with the ATCG. During training a batch size of $\frac{30}{N}$ episodes is used (where N is the number of pedestrian agents), and during validation each scene is evaluated for 5 episodes. In testing each scene is evaluated for 10 episodes. The testing is performed with three different random seeds and the average and standard deviation of the results are reported. The measures reported are

- R_{π}^{+} collision free - the average discounted cumulative reward of pedestrians that do not take part in a collision with a vehicle.
- R_{ρ}^{+} collision free - the average discounted cumulative reward of AVs that do not take part in a collision with a pedestrian.
- $\#$ of collisions - the average number of collisions per episode.
- average $\mu^{\mathcal{D}}$ - average pedestrian density value (4) of the pedestrians' trajectories.

We report R_{μ} the average APILA's discounted cumulative reward and R_{ν} the average AVILA's discounted cumulative reward to show the ATCG's performance with different pedestrian behaviours.

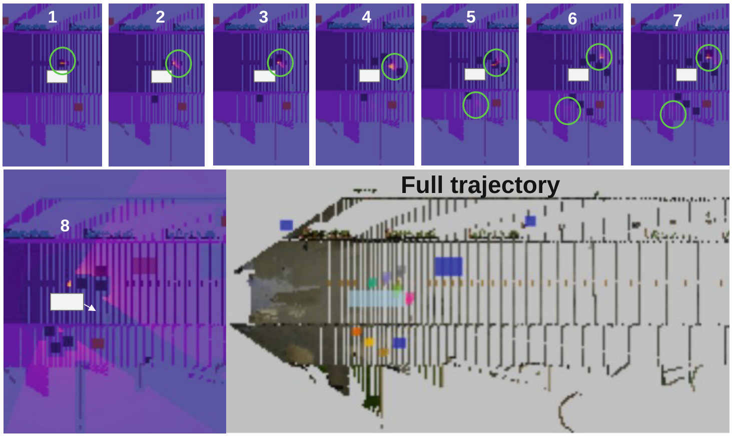


Fig. 7. The ATS's policy and sample initialization. *Full trajectory, bottom right image* The ATS [34] places the 8 pedestrians close to the AV in light blue (traveling to the right), resulting in an unnatural collision scene. *1–8:* The initial pedestrian location distribution μ (peaks of the distribution are shown within green circles) of 8 pedestrians (black boxes) is very peaked (lighter pink color indicates higher probability) close to the white AV. (Color figure online)

We begin by evaluating the APILA without AVILA. AVs are placed randomly on valid initial locations. In the following 1 AV model and 8 pedestrian agents are utilized. A randomly acting AV is used during training and the *base AV model* during testing. APILA outperforms sampling from the ATS [34] and from APILA's prior μ^p as seen in Table 1 *left* by producing as many collisions as the other methods but resulting in a higher pedestrian reward for pedestrians that do not engage in collisions. The APILA's initialized pedestrian motion is more often in areas often visited by pedestrians, i.e. higher average $\mu^{\mathcal{D}}$, compared with the ATS [34] and by the prior. The ATCG places pedestrians more naturally in the scene than ATS, as seen in Fig. 7 and Fig. 8. Ablations showing the differences between APILA and ATS [34] can be found in the supplement (available at https://github.com/MariaPriisalu/atcg and on arXiv).

A fully trained AV makes it harder to find collisions, and thus makes the training of APILA harder. The addition of AVILA allows the ATCG model to place the fully trained AV in the scene relaxing the optimization problem and allowing ATCG to be trained without balancing between the training of the AV

Table 1. *Left:* APILA gives more natural initial locations to pedestrians that are not actively colliding (higher R_π^+ and average $\mu^{\mathcal{D}}$) with the AV, than ATS or the prior. *Right:* ATCG places AVs and pedestrians more naturally than the prior as seen by the higher rewards of the AVs R_ρ^+ and pedestrians R_π^+ that did not collide.

Metric	**Prior** μ^p	**ATS**[34]	**APILA**
R_π^+	3.7(±1.4)	3.0(±0.9)	**4.9**(±0.3)
# of coll.	0.13(±0.06)	**0.4**(±0.2)	**0.27**(±0.06)
average $\mu^{\mathcal{D}}$	0.37(±0.09)	0.34(±0.1)	**0.56**(±0.06)

Metric	**Prior**	**ATCG**
R_π^+	1(±2)	**1.8**(±0.5)
R_ρ^+	−3.1(±0.5)	−**2.6**(±0.1)
# coll.	**0.2**(±0.2)	**0.3**(±0.1)

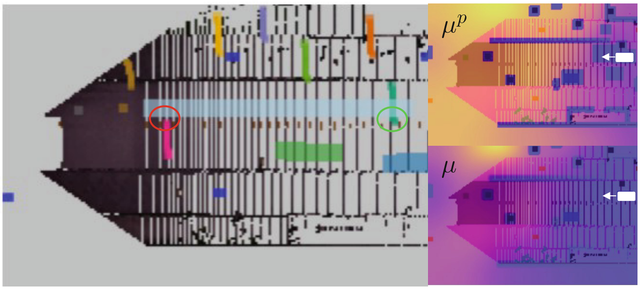


Fig. 8. APILA's policy and sample initialization. APILA's policy μ in *right bottom* is less dispersed than the prior μ^p in *right top*, but still varied enough to place pedestrians in varied positions, mostly on the sidewalk. A collision (red circle) between the light blue AV's trajectory and a pink pedestrian's trajectory is seen as well. The light blue AV traveling to the left and the green pedestrian avoid a collision (green circle). The ATCG places pedestrians more naturally in the scene than the ATS in Fig. 7 where all pedestrians crowd the AV. (Color figure online)

Table 2. The ATCG model is able to find collision scenarios with varied pedestrian behaviour models when the AV and the ATCG are trained alternatively.

Metric	Constant Velocity	Distracted SPL	SPL	Collision Avoiding	Collision Seeking
R_μ APILA's reward↑	0.5(±0.2)	**1.2**(±0.8)	**0.76**(±0.09)	**0.60**(±0.07)	**0.6**(±0.2)
R_ν AVILA's reward↑	−0.3(±0.6)	**0.7**(±0.8)	**0.9**(±0.2)	0.2(±0.4)	−0.1(±0.8)
# collisions↑	**0.8**(±0.3)	**0.9**(±0.3)	**1.07**(±0.06)	0.93(±0.06)	**0.9**(±0.3)

and APILA. The ATCG places AVs and pedestrians that are not involved in a collision more naturally in the scene as indicated by the higher non-collision rewards R^+_ρ, R^+_π compared to sampling from the prior, as seen in Table 1 *right*.

The ATCG is trained alternatively with the AV with the following pedestrian models (all articulated by the Human Locomotion Network (HLN) [35]),

- *Constant Velocity (CV)* - a pedestrian moving at a constant velocity towards its goal location with a speed drawn from the distribution $\mathcal{N}(1.23\,\mathrm{ms}^{-1}, 0.3)$ [66].
- *Semantic Pedestrian Locomotion (SPL)* - semantically reasoning and collision avoiding pedestrian model [35].
- *Distracted SPL+ϵ* - a pedestrian does not notice the closest AV with a probability of 0.3 and continues to be distracted for $k \sim Poisson(2)$ timesteps.
- *Collision Avoiding (CA)* - the SPL model that is trained further with the reward r_π. This pedestrian learns to avoid collisions with the AV (see Fig. 2).
- *Collision Seeking (CS)* - the SPL model that is trained further the reward r_π but with $\lambda_{AV} = 2$. This pedestrian learns to collide with the AV (see Fig. 2).

Independently of the pedestrian model utilized the ATCG can generate collisions as seen in Table 2. The ATCG with *CV* pedestrian model produces the lowest collision rate because the AV can easily learn the pedestrian's motion and avoid collisions. The ATCG with the distracted *SPL+ϵ* and the *SPL* produce a higher reward for AVILA than with the collision seeking (*CS*) or collision avoiding (*CA*) pedestrians. This is likely because both *CS* and *CA* are trained together with the AV and ATCG resulting in a less stable optimization problem for the ATCG (the collision-avoiding and the collision-seeking pedestrian RL games in Fig. 2). The ATCG has a high APILA's reward R_μ even with the *CA* pedestrian because the *CA* is predictable for the ATCG. The *CA* is initialized with the *SPL* model and trained further with *SPL*'s objective and therefore *CA* has a lower entropy than the *SPL*. As expected, the *CA* has fewer collisions than the *SPL* model. During training, it was noticed that training the AV with the AVILA made the AV act more cautious of collisions with other vehicles. The effect of varying the pedestrian model on the ATCG with a constant AV model, and on APILA are found in the supplement.

5 Conclusion and Future Work

By reformulating AV collision scenario generation as a teamed RL problem it is possible to train semantically reasoning RL models that place out AVs and pedestrian agents in a scene. The visually reasoning ATCG generalizes across scenes. The model does not make strict assumptions about the scene, but instead reasons based on visual input. Any prior knowledge is given through the semantically motivated priors. The ATCG being separated into different agents allows for the agents to solve the separate tasks of finding the natural but collision-prone initial distributions of AVs and pedestrians. In a multi-agent scenario the natural distribution of different agents is particularly important because the most collision-prone initial locations are less likely with the increasing number of traffic agents. The proposed ATCG balances the search for collision-prone scenarios with the search for realistic scenarios. When the AV is an RL agent then the ATCG and the AV are playing a MARL game. The ATCG can generate collisions independently if the pedestrian agent is on the ATCG's team, the AV's team, or neither team, see Fig. 2. The ATCG can be used to generate natural-looking collision scenarios for varied AV and pedestrian models, with varied traffic density for varied scenes.

The ATCG can be easily extended by adding further agents that are responsible for placing out other constraining factors for the AV (for example placing out parked cars that occlude the AV's vision). This can be done easily by introducing a scene semantics dependent prior and a reward that balances the object's natural occurrence in scenes with obstructiveness for the AV. This is a natural way of ensuring that the AV and the pedestrians have enough constraints such that collisions occur even as the AV and the pedestrian models improve in collision avoidance. Extending the ATCG agent with additional RL agents and evaluating state-of-the-art AV models is our future work.

Acknowledgements. This work was supported by the European Research Council Consolidator grant SEED, CNCS-UEFISCDI PCCF-2016-0180, and the Swedish Foundation for Strategic Research (SSF) Smart Systems Program.

References

1. Zhu, Z., Zhao, H.: A survey of deep RL and IL for autonomous driving policy learning. IEEE Trans. Intell. Transp. Syst. **23**, 14043–14065 (2021)
2. Ye, F., Zhang, S., Wang, P., Chan, C.-Y.: A survey of deep reinforcement learning algorithms for motion planning and control of autonomous vehicles. In: Proceedings of the 2021 IEEE Intelligent Vehicles Symposium, pp. 1073–1080 (2021)
3. Paden, B., Čáp, M., Yong, S.Z., Yershov, D., Frazzoli, E.: A survey of motion planning and control techniques for self-driving urban vehicles. IEEE Trans. Intell. Veh. **1**, 33–55 (2016)
4. Kang, Y., Yin, H., Berger, C.: Test your self-driving algorithm: an overview of publicly available driving datasets and virtual testing environments. IEEE Trans. Intell. Transp. Syst. **4**, 171–185 (2019)
5. Suzuki, T., Aoki, Y., Kataoka, H.: Pedestrian near-miss analysis on vehicle-mounted driving recorders. In: Proceedings of Fifteenth International Conference on Machine Vision Applications, pp. 416–419. IEEE (2017)
6. Yao, Y., Xu, M., Wang, Y., Crandall, D.J., Atkins, E.M.: Unsupervised traffic accident detection in first-person videos. In: Proceedings of 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 273–280. IEEE (2019)

7. Aliakbarian, M.S., Saleh, F.S., Salzmann, M., Fernando, B., Petersson, L., Andersson, L.: VIENA^2: a driving anticipation dataset. In: Jawahar, C.V., Li, H., Mori, G., Schindler, K. (eds.) ACCV 2018. LNCS, vol. 11361, pp. 449–466. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20887-5_28
8. Chandra, R., et al.: METEOR: a massive dense & heterogeneous behavior dataset for autonomous driving. CoRR abs/2109.07648. arXiv: 2109.07648 (2021)
9. Ding, W., et al.: A survey on safety-critical driving scenario generation - a methodological perspective. CoRR abs/2202.02215. arXiv: 2202.02215 (2022)
10. Abdessalem, R.B., Panichella, A., Nejati, S., Briand, L.C., Stifter, T.: Testing autonomous cars for feature interaction failures using many-objective search. In: Huchard, M., Kästner, C., Fraser, G. (eds.) Proceedings of the 33rd IEEE/ACM International Conference on Automated Software Engineering, pp. 143–154. ACM (2018)
11. Li, Y., Tao, J., Wotawa, F.: Ontology-based test generation for automated and autonomous driving functions. Inf. Softw. Technol. **117**, 106200 (2020)
12. Abdessalem, R.B., Nejati, S., Briand, L.C., Stifter, T.: Testing advanced driver assistance systems using multi-objective search and neural networks. In: Lo, D., Apel, S., Khurshid, S. (eds.) Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering, pp. 63–74. ACM (2016)
13. Zhong, Z., Kaiser, G., Ray, B.: Neural network guided evolutionary fuzzing for finding traffic violations of autonomous vehicles. IEEE Trans. Softw. Eng. (2022)
14. Bussler, A., Hartjen, L., Philipp, R., Schuldt, F.: Application of evolutionary algorithms and criticality metrics for the verification and validation of automated driving systems at urban intersections. In: Proceedings of the 2020 IEEE Intelligent Vehicles Symposium, pp. 128–135 (2020)
15. Almanee, S., Wu, X., Huai, Y., Chen, Q.A., Garcia, J.: scenoRITA: generating less-redundant, safety-critical and motion sickness-inducing scenarios for autonomous vehicles. CoRR abs/2112.09725. arXiv: 2112.09725 (2021)
16. Ding, W., Chen, B., Li, B., Eun, K.J., Zhao, D.: Multimodal safety-critical scenarios generation for decision-making algorithms evaluation. IEEE Rob. Autom. Lett. **6**, 1551–1558 (2021)
17. Parashar, P., Cosgun, A., Nakhaei, A., Fujimura, K.: Modeling preemptive behaviors for uncommon hazardous situations from demonstrations. CoRR abs/1806.00143. arXiv: 1806.00143 (2018)
18. Demetriou, A., Alfsvåg, H., Rahrovani, S., Chehreghani, M.: A deep learning framework for generation and analysis of driving scenario trajectories. CoRR abs/2007.14524. arXiv: 2007.14524 (2020)
19. Nishiyama, D., et al.: Discovering avoidable planner failures of autonomous vehicles using counterfactual analysis in behaviorally diverse simulation. In: Proceedings of 23rd IEEE International Conference on Intelligent Transportation Systems, pp. 1–8. IEEE (2020)
20. Zhong, Z., et al.: A survey on scenario-based testing for automated driving systems in high-fidelity simulation. CoRR abs/2112.00964. arXiv: 2112.00964 (2021)
21. Ding, W., Xu, M., Zhao, D.: CMTS: a conditional multiple trajectory synthesizer for generating safety-critical driving scenarios. In: Proceedings of 2020 IEEE International Conference on Robotics and Automation, pp. 4314–4321. IEEE (2020)
22. Sun, X., Zhang, Y., Zhou, W.: Building narrative scenarios for human-autonomous vehicle interaction research in simulators. In: Cassenti, D.N., Scataglini, S., Rajulu, S.L., Wright, J.L. (eds.) AHFE 2020. AISC, vol. 1206, pp. 150–156. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-51064-0_20

23. Karunakaran, D., Worrall, S., Nebot, E.: Efficient falsification approach for autonomous vehicle validation using a parameter optimisation technique based on reinforcement learning (2020). arXiv: 2011.07699
24. Karunakaran, D., Worrall, S., Nebot, E.M.: Efficient statistical validation with edge cases to evaluate highly automated vehicles. In: Proceedings of the 23rd IEEE International Conference on Intelligent Transportation Systems, pp. 1–8. IEEE (2020)
25. Ding, W., Chen, B., Xu, M., Zhao, D.: Learning to collide: an adaptive safety-critical scenarios generating method. In: Proceedings of 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2243–2250. IEEE (2020)
26. Wang, J., et al.: AdvSim: generating safety-critical scenarios for self-driving vehicles. In: Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9909–9918 (2021)
27. Hamdi, A., Mueller, M., Ghanem, B.: SADA: semantic adversarial diagnostic attacks for autonomous applications. In: Proceedings of Thirty-Fourth Conference on Artificial Intelligence, pp. 10901–10908. AAAI Press (2020)
28. Gupta, P., Coleman, D., Siegel, J.: Towards safer self-driving through great PAIN (Physically Adversarial Intelligent Networks). CoRR abs/2003.10662. arXiv: 2003.10662 (2020)
29. Wen, M., Park, J., Cho, K.: A scenario generation pipeline for autonomous vehicle simulators. HCIS **10**, 24 (2020)
30. Koren, M., Kochenderfer, M.J.: Efficient autonomy validation in simulation with adaptive stress testing. In: Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference, pp. 4178–4183. IEEE (2019)
31. Muktadir, G.M., Whitehead, J.: Adversarial jaywalker modeling for simulation-based testing of autonomous vehicle systems. In: Proceedings of the 2022 IEEE Intelligent Vehicles Symposium, pp. 1697–1702. IEEE (2022)
32. Ding, W., Lin, H., Li, B., Zhao, D.: CausalAF: causal autoregressive flow for safety-critical driving scenario generation. In: Proceedings of the 2022 Conference on Robot Learning. PMLR (2022, to appear)
33. Abeysirigoonawardena, Y., Shkurti, F., Dudek, G.: Generating adversarial driving scenarios in high-fidelity simulators. In: Proceedings of the 2019 IEEE International Conference on Robotics and Automation, pp. 8271–8277. IEEE (2019)
34. Priisalu, M., Pirinen, A., Paduraru, C., Sminchisescu, C.: Generating scenarios with diverse pedestrian behaviors for autonomous vehicle testing. In: Proceedings of the 2021 Conference on Robot Learning. PMLR, vol. 164, pp. 1247–1258 (2021)
35. Priisalu, M., Paduraru, C., Pirinen, A., Sminchisescu, C.: Semantic synthesis of pedestrian locomotion. In: Ishikawa, H., Liu, C.-L., Pajdla, T., Shi, J. (eds.) ACCV 2020. LNCS, vol. 12623, pp. 470–487. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-69532-3_29
36. Yang, Z., et al.: SurfelGAN: synthesizing realistic sensor data for autonomous driving. In: Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11115–11124. Computer Vision Foundation/IEEE (2020)
37. Li, W., et al.: AADS: Augmented autonomous driving simulation using data-driven algorithms. Sci. Rob. **4**, eaaw0863 (2019)
38. Saadatnejad, S., Li, S., Mordan, T., Alahi, A.: A shared representation for photorealistic driving simulators. IEEE Trans. Intell. Transp. Syst. **23**, 13835–13845 (2022)

39. Lee, S., et al.: Visuomotor understanding for representation learning of driving scenes. In: Proceedings of the 30th British Machine Vision Conference, p. 262. BMVA Press (2019)
40. Wu, Y., Gao, R., Park, J., Chen, Q.: Future video synthesis with object motion prediction. In: Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5538–5547. Computer Vision Foundation/IEEE (2020)
41. Li, Z., et al.: READ: large-scale neural scene rendering for autonomous driving. CoRR abs/2205.05509. arXiv: 2205.05509 (2022)
42. Makansi, O., Çiçek, Ö., Buchicchio, K., Brox, T.: Multimodal future localization and emergence prediction for objects in egocentric view with a reachability prior. In: Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4353–4362. IEEE (2020)
43. Sun, J., Averbuch-Elor, H., Wang, Q., Snavely, N.: Hidden footprints: learning contextual walkability from 3D human trails. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12363, pp. 192–207. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58523-5_12
44. Fang, J., et al.: Behavioral intention prediction in driving scenes: a survey. CoRR abs/2211.00385. arXiv: 2211.00385 (2022)
45. Rasouli, A., Tsotsos, J.K.: Autonomous vehicles that interact with pedestrians: a survey of theory and practice. IEEE Trans. Intell. Transp. Syst. **21**, 900–918 (2019)
46. Hanselmann, N., Renz, K., Chitta, K., Bhattacharyya, A., Geiger, A.: KING: generating safety-critical driving scenarios for robust imitation via kinematics gradients. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Proceedings of the 2022 European Conference on Computer Vision, vol. 13698, pp. 335–352. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19839-7_20
47. Wan, Z., et al.: Too afraid to drive: systematic discovery of semantic DoS vulnerability in autonomous driving planning under physical-world attacks. In: The Proceedings of the 29th Annual Network and Distributed System Security Symposium. The Internet Society (2022)
48. Mockus, J.: On Bayes methods for seeking an extremum. Avtomatika i Vychislitelnaja Technika **3**, 53–62 (1972)
49. Dai, Q., Xu, X., Guo, W., Huang, S., Filev, D.: Towards a systematic computational framework for modeling multi-agent decision-making at micro level for smart vehicles in a smart world. Robot. Auton. Syst. **144**, 103859 (2021)
50. Rempe, D., Philion, J., Guibas, L.J., Fidler, S., Litany, O.: Generating useful accident-prone driving scenarios via a learned traffic prior. In: Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 17305–17315 (2022)
51. Serrano, S.M., Llorca, D.F., Daza, I.G., Sotelo, M.Á.: Insertion of real agents behaviors in CARLA autonomous driving simulator. In: da Silva, H.P., Vanderdonckt, J., Holzinger, A., Constantine, L.L. (eds.) Proceedings of the 6th International Conference on Computer-Human Interaction Research and Applications, pp. 23–31. SCITEPRESS (2022)
52. Vasquez, R., Farooq, B.: Multi-objective autonomous braking system using naturalistic dataset. In: Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference, pp. 4348–4353 (2019)
53. Schmitt, P., et al.: nuReality: a VR environment for research of pedestrian and autonomous vehicle interactions. CoRR abs/2201.04742. arXiv: 2201.04742 (2022)

54. Li, C., Cheng, C., Sun, T., Chen, Y., Yan, R.: ComOpT: combination and optimization for testing autonomous driving systems. In: Proceedings of the 2022 IEEE International Conference on Robotics and Automation, pp. 7738–7744. IEEE (2022)
55. Boloor, A., et al.: Attacking vision-based perception in end-to-end autonomous driving models. J. Syst. Architect. **110**, 101766 (2020)
56. Machiraju, H., Balasubramanian, V.N.: A little fog for a large turn. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision 2020, pp. 2902–2911 (2020)
57. Cao, Y., et al.: Adversarial sensor attack on LiDAR-based perception in autonomous driving. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp. 2267–2281. Association for Computing Machinery (2019)
58. Fursa, I., et al.: Worsening perception: real-time degradation of autonomous vehicle perception performance for simulation of adverse weather conditions. SAE Int. J. Connected Autom. Veh. **5**, 87–100 (2022)
59. Kim, E., et al.: Querying labelled data with scenario programs for sim-to-real validation. In: Proceedings of the 2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems, pp. 34–45 (2022)
60. Spooner, J., Palade, V., Cheah, M., Kanarachos, S., Daneshkhah, A.: Generation of pedestrian crossing scenarios using Ped-Cross Generative Adversarial Network. Appl. Sci. **11**, 471 (2021)
61. Dagdanov, R., Eksen, F., Durmus, H., Yurdakul, F., Ure, N.K.: DeFIX: detecting and fixing failure scenarios with reinforcement learning in imitation learning based autonomous driving. In: Proceedings of the 25th IEEE International Conference on Intelligent Transportation Systems, pp. 4215–4220. IEEE (2022)
62. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn. **8**, 229–256 (1992)
63. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) Proceedings of the 3rd International Conference on Learning Representations (2015)
64. Hydén, C.: Traffic conflicts technique: state-of-the-art. Traffic Saf. Work Video Process. **37**, 3–14 (1996)
65. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: an open urban driving simulator. In: Proceedings of the 2017 Conference on Robot Learning. PMLR, vol. 78, pp. 1–16 (2017)
66. Chandra, S., Bharti, A.K.: Speed distribution curves for pedestrians during walking and crossing. Procedia Soc. Behav. Sci. **104**, 660–667 (2013)

Supplementary Material of Varied Realistic Autonomous Vehicle Collision Scenario Generation

Maria Priisalu[1], Ciprian Paduraru[2], and Cristian Smichisescu[1,3]

[1] Lund University, Sweden
{maria.priisalu,cristian.sminchisescu}@math.lu.se
[2] University Of Bucharest, Romania ciprian.paduraru@fmi.unibuc.ro
[3] Google Research

1 Additional experimental results

1.1 Qualitative results of the ATCG model

In Fig. 1 it can be seen that ATCG produces natural-looking and varied initial locations for AVs and pedestrians (all external agents in dark blue color). Some of the pedestrians (in green circles) manage to avoid collisions with the AVs and others do not (red circles).

The policy of the ATCG agents can be seen in Fig. 2. Note that the ATCG's policy is more dispersed than that of ATS as seen in Figure 7 of the main paper. This is due to the Gaussian smoothing in the prior of APILA, introducing the estimated pedestrian density $\mu^{\mathcal{D}}$ as factor in APILA's prior μ^{p} and introducing the sum of the standard deviations of the initial location coordinates $r_{\sigma}(\boldsymbol{x}_0)$ in the total reward of APILA. See section §1.2 below.

The Table 2 in the main paper the AV and the ATCG have been trained alternatively, here in Table 1 we only train the ATCG and keep the AV model constant. The ATCG is able to find collision scenarios for the *base AV* model independently of pedestrian behavior model. The *Collision seeking* and *Collision avoiding* pedestrians result in the highest rewards for APILA and AVILA.

1.2 Ablation study of APILA

The APILA is adapted from ATS [1] by introducing smoothing μ_n^{TTC} with 2D Gaussian filter (G) on the ATS's prior μ_n^{TTC}, by including the estimated pedestrian density $\mu^{\mathcal{D}}$ in the prior μ^{p} and by the introduction of the reward term $r_{\sigma}(\boldsymbol{x}_0)$. The effect of these ablations on APILA can be seen in Table 2. Note that ATS [1] only places out pedestrians, and AVILA is novel. The policy of the ablation in Table 2 without G,$\mu^{\mathcal{D}}$ and $r_{\sigma}(\boldsymbol{x}_0)$ can be seen in Figure 7 of the main paper. The visual inspection of Figure 7 of the main paper shows that without smoothing pedestrians crowd the AV. To avoid placing all pedestrians near the AV smoothing was introduced, and the resulting pedestrians' initializations are shown in Fig. 1 and Fig. 2 and Figure 8 of the main paper. Smoothing (G) decreases APILAs collision rate but increases the pedestrians' average pedestrian density value (average $\mu^{\mathcal{D}}$), as seen in Table 2. After smoothing more pedestrians get

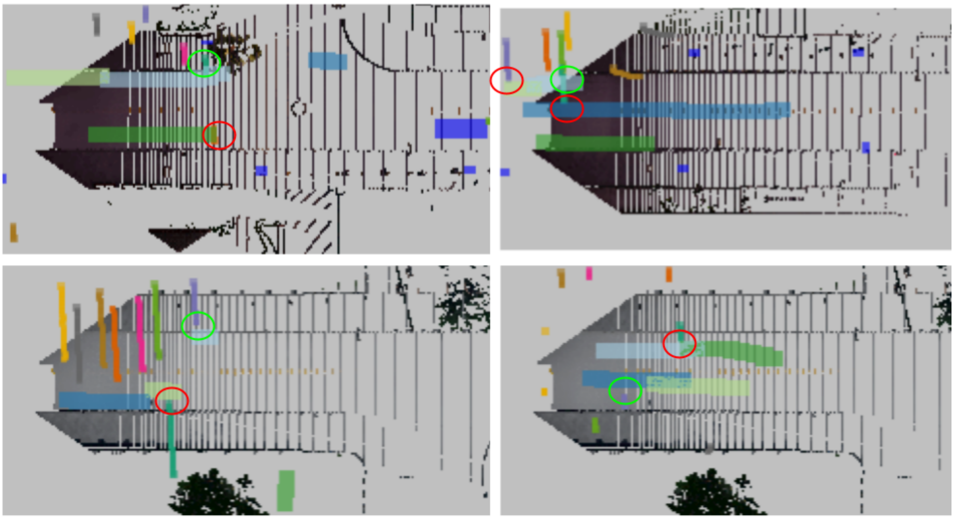


Fig. 1. Four sample scenarios showing multiple cases where the AV and pedestrians manage to avoid collisions (shown with green circles) and cases where they collide (shown with red circles). The top lane of cars is moving to the left and the bottom lane to the right.

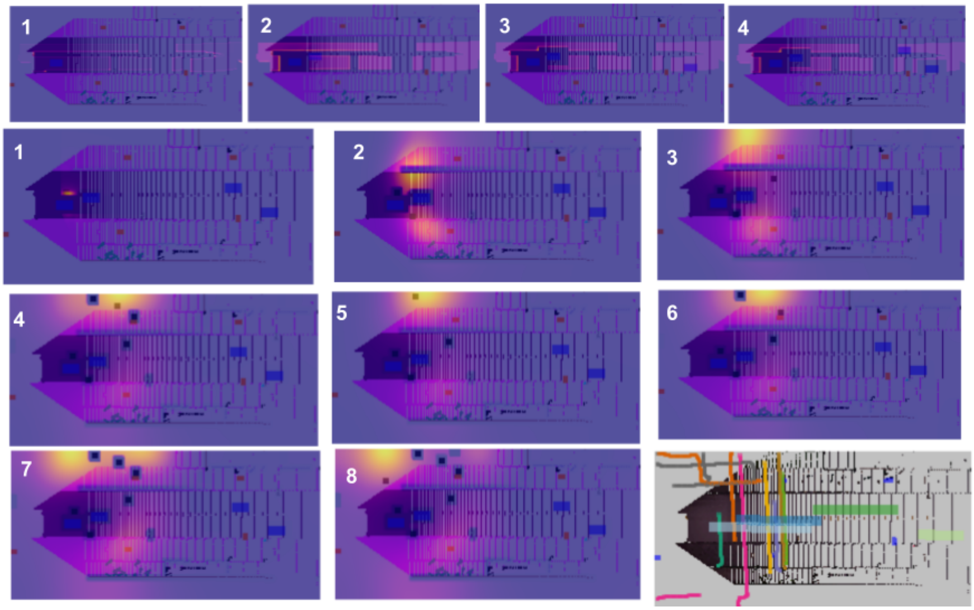


Fig. 2. A sample scenario depicting no collision but varied and realistic initialization of pedestrians and AVs by ATCG. The first row shows AVILA's policy for placing out 4 AVs, second to fourth row show APILA's policy for placing out the first to 8th pedestrian. The trajectories of all of the agents are shown in the fourth row third column.

Table 1. Performance of ATCG trained with various pedestrian models and with the *base AV* model.

Metric	**Constant velocity**	**Distracted SPL**	**SPL**	**Collision avoiding**	**Collision seeking**
R_μ APILA's	0.6(±0.3)	0.8(±0.5)	0.5(±0.4)	**1.0**(±0.6)	**1.0**(±0.1)
R_ν AVILA's	-7.6(±0.6)	-9(±2)	**-7.0**(±0.6)	**-6.7**(±0.9)	**-8**(±3)
# collisions	**1.0**(±0.2)	**0.7**(±0.3)	**1**(±0.2)	**0.87**(±0.06)	**0.9**(±0.4)

Table 2. Ablations of APILA.

$\mu^{\mathcal{D}}$	$r_\sigma(\boldsymbol{x}_0)$	G	↑ R^+_π collision free	↑# collisions	↑ average $\mu^{\mathcal{D}}$
			1.5(± 0.3)	**0.5**(± 0.1)	0.32(± 0.2)
✓			3.2(± 0.4)	**0.5**(± 0.1)	0.25(± 0.08)
	✓		2.8(± 1.6)	**0.4**(± 0.2)	0.3(± 0.1)
		✓	**6.3**(± 0.9)	0.1(± 0.1)	**0.51**(± 0.07)
	✓	✓	4.0(± 0.9)	0.3(± 0.1)	**0.52**(± 0.08)
✓	✓	✓	**4.9**(± 0.3)	0.27(± 0.06)	**0.56**(± 0.06)

initialized on the sidewalk and further away from the AV where more pedestrians can naturally be found. The pedestrian density map $\mu^{\mathcal{D}}$ inclusion provides APILA prior knowledge of typical pedestrian distributions in the given scene. The reward $r_\sigma(\boldsymbol{x}_0)$ encourages variation among the initial locations of the pedestrians. Both the inclusion of $r_\sigma(\boldsymbol{x}_0)$ and $\mu^{\mathcal{D}}$ increase the collision rate and decrease the tendency of pedestrian agents to stay near areas often visited by pedestrians. The decrease in collisions brought about by smoothing is countered by the inclusion of $\mu^{\mathcal{D}}$ and $r_\sigma(\boldsymbol{x}_0)$. This is likely because both $\mu^{\mathcal{D}}$ and $r_\sigma(\boldsymbol{x}_0)$ encourage variation among the initial locations of the N pedestrians, leading the model to find more scenarios that lead to collisions. We propose to include all of the model components $\mu^{\mathcal{D}}$,G and $r_\sigma(\boldsymbol{x}_0)$ to balance between likely and collision-prone scenarios, but the choice may vary due to the application.

The effect of varying the pedestrian behavior policy on APILA can be seen in Table 3, utilizing a goal-driven collision avoiding pedestrian such as SPL results in a comparable number of collisions with using a constant velocity pedestrian model. We also report the metric $H(\pi^-)$ collision - the pedestrian behaviour policy's entropy during its collision course. The constant velocity pedestrian results in a lower average μ^D than the other pedestrian behavior models, as the other models are trained to tend to stay in areas frequented by pedestrians. Utilizing randomly distracted pedestrians [1] does not lead to an increase in collisions, likely because the pedestrian's distracted behaviour is unstructured and thus not learnable for APILA, as also confirmed in [1]. The collision seeking pedestrian attains a higher entropy during collisions and is thus more unpredictable in near-collision scenarios causing the collision frequency to drop, while the collision avoiding pedestrian's entropy is low making the pedestrian's motion easier to predict even

Table 3. The constant velocity pedestrian model results in a decreased number of collisions. APILA can generate collisions independently of the pedestrian behaviour policy. The SPL model produces the most pedestrian-like behaviour.

Metric	**Constant velocity**	**Distracted SPL**	**SPL**	**Collision avoiding**	**Collision seeking**
R_π^+ collision free	1.8(±0.1)	3.1(±1.5)	**4.9**(±0.3)	2.5(±0.6)	3.7(±0.8)
# collisions	**0.2**(±0.1)	**0.4**(±0.2)	**0.27**(±0.06)	**0.3**(±0.1)	0.13(±0.06)
average $\mu^{\mathcal{D}}(x_t)$	0.48(±0.08)	0.45(±0.04)	**0.56**(±0.06)	0.31(±0.01)	0.5(±0.2)
$H[\pi]^-$ collision	0	0.08(±0.02)	**0.05**(±0.03)	**0.04**(±0.01)	0.1(±0.1)

if the pedestrian attempts to avoid collisions. Most notably APILA can generate collisions independently of the pedestrian behaviour policy.

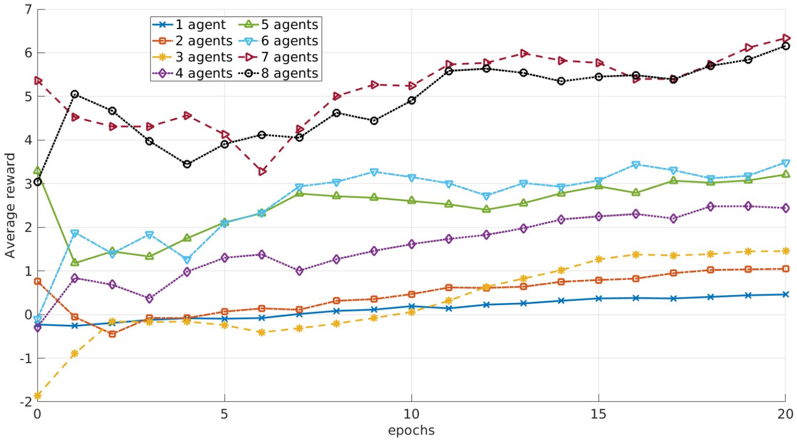


Fig. 3. APILAs model trained with 7 and 8 pedestrians start learning with a higher reward and continue to increase in reward with an increasing number of epochs.

As expected increasing the number of pedestrians implies a higher collision frequency giving a higher reward value, as seen in Fig. 3.

1.3 Ablations of AVILA

In Fig. 4 the average number of collisions between AVs and pedestrians is shown. It can be seen that a larger number of AVs results in general in a larger number of collisions. Three AVs appear to be optimal for the given dataset. It could be that beyond this the AVs have trouble avoiding collisions with one another when trying to avoid pedestrians.

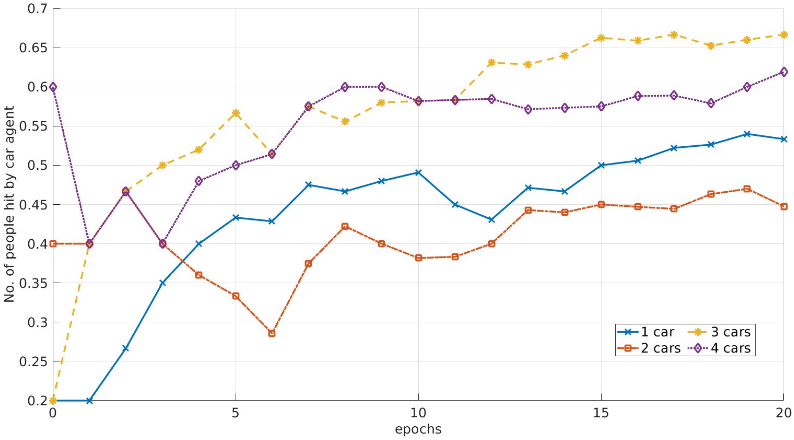


Fig. 4. Three AVs produce the largest amount of collisions between the AVs and pedestrians on the validation set.

References

1. Priisalu, M., Pirinen, A., Paduraru, C. & Sminchisescu, C. *Generating Scenarios with Diverse PedestrianBehaviors for Autonomous Vehicle Testing* in *PMLR: Proceedings of CoRL 2021* (Nov. 2021).

Paper IV

Paper IV is not included in the electronic copy of this thesis, but can be found in the printed version.

Paper v

Semantic and Articulated Pedestrian Sensing Onboard a Moving Vehicle

Maria Priisalu[1]

Lund University, Sweden
maria.priisalu@math.lu.se

Abstract. It is difficult to perform 3D reconstruction from on-vehicle gathered video due to the large forward motion of the vehicle. Even object detection and human sensing models perform significantly worse on onboard videos when compared to standard benchmarks because objects often appear far away from the camera compared to the standard object detection benchmarks, image quality is often decreased by motion blur and occlusions occur often. This has led to the popularisation of traffic data-specific benchmarks. Recently Light Detection And Ranging (LiDAR) sensors have become popular to directly estimate depths without the need to perform 3D reconstructions. However, LiDAR-based methods still lack in articulated human detection at a distance when compared to image-based methods. We hypothesize that benchmarks targeted at articulated human sensing from LiDAR data could bring about increased research in human sensing and prediction in traffic and could lead to improved traffic safety for pedestrians.

Keywords: Pedestrian Detection · Autonomous Vehicles

1 Introduction

Autonomous vehicle (AV) research is gaining momentum [1–4] in modeling vehicle-to-vehicle interactions, but pedestrian-vehicle motion planning models [5–46] could be improved by articulated human motion modelling. Pedestrians in difference to vehicles provide strong visual cues of their intent, as well as current and future motion through their articulated pose [47–49]. Human motion is predictable up to one second with around one centimeter average per joint error when observing articulated motion [50]. The motion information present in the pedestrian pose is unused in most AV motion planning models [5–46], as well as in AV model testing. Progress in articulated pedestrian modeling is slowed down by the lack of data due to the difficulty in recovering articulated pedestrian poses in real traffic scenarios. The importance of preserving the relationship between pedestrian motion and scene semantics on pedestrian motion perception is shown in Fig. 1. The lack of data has lead to the development of AV scene understanding models [5–46] that are oblivious to pedestrian poses and other visual cues (such as facial expressions etc), thus simply omitting available motion cues. Further AV testing is not yet utilizing realistic articulated pedestrian models and instead tests AV's interactions with heuristic pedestrian motions [51–77].

Since AV's are not evaluated in interactions with real humans at scale the possible safety issues in pedestrian detection, tracking and forecasting are relatively unknown.

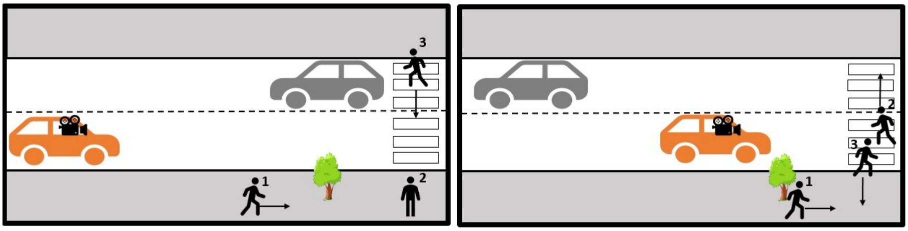


Fig. 1. By semantically modeling articulated pedestrians an AV in orange in the left figure can foresee that pedestrian 1 will continue moving in the same direction eventually being occluded by the tree (see right figure), that pedestrian 2 may choose to cross when standing next to a crosswalk (see right figure), and that the third pedestrian will continue to cross once visible. Modeling articulated pedestrians will also ease the AV to differentiate between the second and third pedestrians as their paths cross (see figure on right), as a sudden change in direction is unlikely on a crossroad and given the pedestrians' articulated pose.

We argue that articulated semantically grounded pedestrian sensing and modeling is currently an underdeveloped research field due to a lack of Ground Truth (GT) data. Supervised articulated human sensing models [78–89] are often evaluated on clean benchmarks [90–95] where humans are and clearly and often fully visible, close to the camera and captured in good lightning conditions. This leads to methods that fail at a distance as well as in the presence of motion blur or poor lightning and occlusions. Unsupervised [94, 96–104] and weakly supervised [105–115] training have become popular to overcome the lack of difficult and varied GT data. These models could however be improved with combined temporal and traffic-centered semantic modeling to obtain human 3D pose tracking at scale from a moving vehicle.

A ground truth dataset of articulated human motion in 3D would allow one to evaluate the discrepancy between the true and estimated scale and depth, robustness to occlusions, and motion blur in human pose detection and forecasting. In parallel to this work, an approximated dataset of articulated humans in the wild has been released [116], but the dataset still exhibits humans that are close to the camera in the presence of little camera motion when compared to images from traffic and lacks annotations in the presence of large occlusions. Even though [116] is a step in the right direction it does not express the full complexity of the problem of articulated pedestrian motion estimation from onboard vehicles.

Existing monocular absolute scale depth estimators generalize poorly on previously unseen scenes [117, 118]. The same may be expected of the partially supervised and unsupervised 3D human sensing models [94, 96–115], and this is likely to also affect the estimated limb lengths of the pedestrian. Correctly estimated limb lengths however allow for a precise estimation of the pedestrian's travelling speed. Note that a moving camera requires a robust and temporally smooth pedestrian sensor and motion model to deal with possible image blur, occlusions and to avoid confusion between the motion of the pedestrian and the camera. Robust and complete pedestrian motion sensing and prediction may directly reduce the number of lethal collisions with AVs.

Pedestrian trajectory forecasting is hard because pedestrians appear to move stochastically when compared to the more regular motion of cars, in particular when pedestrians are modelled by their bounding (3d) boxes [5–46]. In general pedestrian motion prediction is hard as the goal of the pedestrians and the reason for a particular speed is unknown even if articulated motion is available. But pedestrians plan their motion in the scene depending on the geometry of the semantics surrounding them; for example, pedestrians may cross the road to avoid staying on a pavement that is very shallow and is next to a densely trafficked road [48, 49]. Further, pedestrian dynamics depend on the particular pedestrian's physique [50]. A complete pedestrian forecasting model should therefore be semantically aware as well as articulated. Currently, to our knowledge only [119] present an articulated semantically reasoning pedestrian forecasting model. A key difficulty in training articulated and semantically reasoning pedestrian models lies in the lack of data as mentioned before, but also in the lack of varied data. Pedestrians act often monotonely in traffic [120–122] as complex behaviors occur seldom in traffic, and existing datasets often do not express the full variability in human dynamics and appearances. To be utilized on-board in real time further research is necessary into robust real-time articulated semantically reasoning pedestrian motion models.

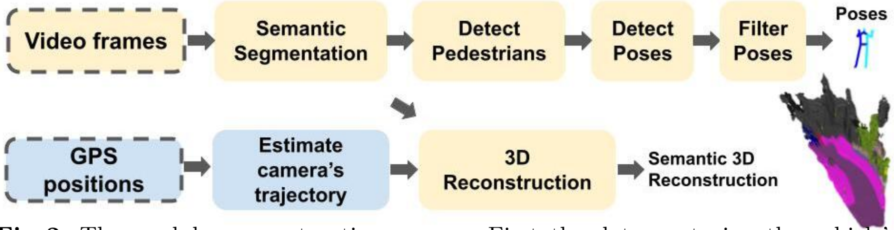


Fig. 2. The modular reconstruction process: First the data capturing the vehicle's trajectory is estimated from GPS coordinates or accelerometer data, this is then used to initialize camera matrices in the 3D reconstruction of the scene. The frames of the binocular video are semantically segmented, semantic segmentation is used to remove moving objects (vehicles and people), and the background objects are 3D reconstructed. The semantic segmentation (or images) is then used to find the bounding boxes (BBox) of pedestrians. Then pose estimation is performed followed by filtering to disallow physically unplausible poses. Note that from semantic segmentation 3D BBox of cars can be estimated.

Autonomous vehicles typically have a number of sensors that all together generate large amounts of data (possibly up to the order of Tb per minute), so the data must be filtered for salient objects. By filtering the data we stand at the risk of possibly missing something important like a partially occluded pedestrian. Therefore how to best represent a traffic scene for autonomous driving is still an open research topic [123–127]. Within motion planning High Definition (HD) maps containing scene details in a compact representation [128], and Bird's Eye View (BEV) images that is to say top view image of the scene, are common because they allow for 2D vision models to easily be utilized on traffic data [124]. Both HD and BEV are compressed scene representations that do not in general allow for sensor data augmentations. In this work we opt for a semantically labeled 3D reconstruction with articulated pedestrians because this allows for detailed modeling of pedestrians, the evaluation of physical distances between objects, and data augmentation for a number of sensors (such as camera and LiDAR). This is an uncompressed scene representation that allows for data augmentation and testing but it is difficult to recover from only onboard binocular video, as will be detailed. Human sensing is performed on images [129–141] because this is a more mature research field than human sensing from other sensor data such as LiDAR-scans [142–146].

Recovering a semantic 3D model of the scene with articulated pedestrians can be done modularly as shown in Fig. 2 by estimating the recording device's motion, semantically segmenting the scene, and 3D reconstructing the scene. Adding articulated pedestrians to the 3D scene reconstruction requires detecting the pedestrians in the scene, estimating the pose of the pedestrians in 2D, estimating the pose of the pedestrians in 3D, and filtering any physically unrealistic poses. A number of estimation errors can occur along the way, making such data gathering hard. We hypothesize that articulated human sensing, tracking and prediction could be improved by combining the three tasks, as is done for vehicles in [146–151]. After the development of the presented results pose tracking has been posed as the problem of tracking the pose of one or more pedestrians [152–154].

Note that even though human motion can be captured with a Motion Capture (MoCap) system, or recently even from selected images [116], it is not trivial to set up large-scale experiments to gather traffic datasets that contain a large variety of possible scene geometries, semantics, and GT poses. This is because MoCap data gathering requires intervening with the scene, and existing human pose sensing methods from images cannot yet capture the poses of all humans in the images [116]. Further, most MoCap methods cannot be utilized accurately outdoors with large occlusions. More research is needed in human motion capture in traffic. Markerless human pose detection results often look impressive [155], but don't often present any results for humans who are far away in the presence of motion blur, which is the case in traffic data. Human detection at a distance in the presence of motion blur is still challenging, let alone human pose detection. Other sensors can be used to remedy motion blur and aid in human detection [156] and articulated human sensing, for example [157] perform an initial step to utilize LiDAR and images to detect distant humans in real traffic data.

2 Scene reconstruction

We use the Cityscapes dataset [158] that consists of binocular video sequences, with a length of 30 frames at 17 frames per second, gathered from calibrated

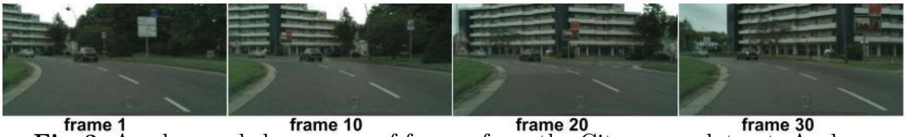


Fig. 3. A sub-sampled sequence of frames from the Cityscapes dataset, Aachen.

cameras placed on the front screen of a vehicle. Sample images are shown in Fig. 3. The data gathering vehicle's position can be estimated from the provided GPS coordinates or accelerometer data. Disparity maps are provided for each frame, and a GT semantic segmentation is provided of the leftmost camera's image at the 20th frame. The images contain some blur because they are captured from behind the windscreen as the vehicle moves. Image blur, the fast camera motion in the forward direction (most 3D reconstruction methods are fragile to this) and independently moving objects make 3D reconstruction of the sequences hard. The inherent difficulties in 3D reconstructing onboard videos have led to the increased popularity of LiDAR for depth estimation.

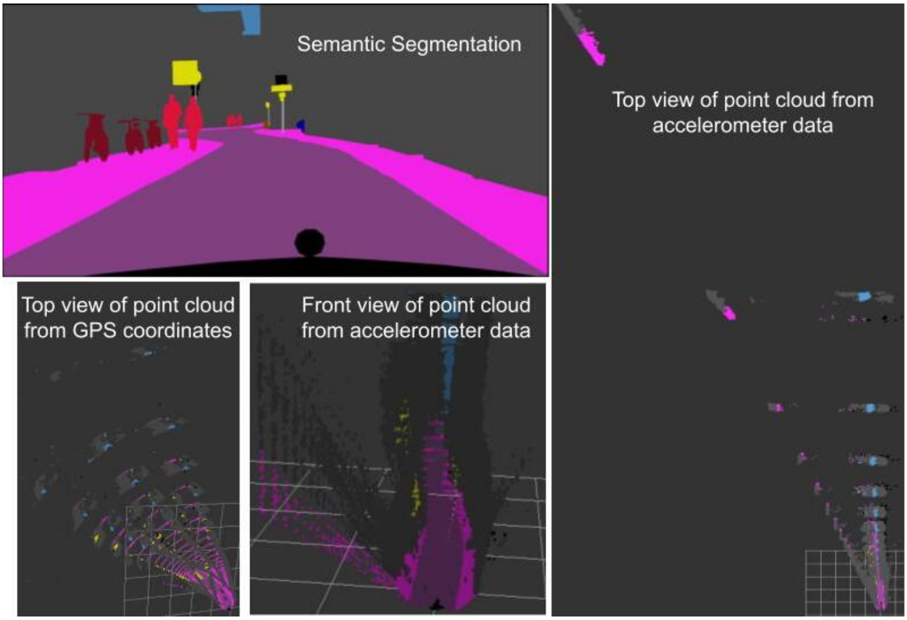

Semantic Segmentation
Top view of point cloud from accelerometer data
Top view of point cloud from GPS coordinates
Front view of point cloud from accelerometer data

Fig. 4. Visualizations of 3D pointclouds from using the vehicle's GPS coordinates or accelerometer readings to estimate camera position with per-frame disparity maps. The GPS is noisier than the accelerometer resulting in a noisier pointcloud.

2.1 Initial Camera Positions

Assuming that the cameras cannot move within the rig or the car we can estimate the cameras' motion as the vehicle's motion. The vehicle's motion can be estimated from the Global Positioning System (GPS) or the accelerometer data. The GPS coordinates contain jumps as seen in Fig. 4 where the cameras' estimated position and each frame's disparity map are used to create pointclouds for each frame that are then aggregated. It can be seen that in the GPS-based vehicle trajectory, the vehicle's rotation oscillated from frame to frame causing the 3d point clouds of different frames to diverge, while the accelerometer data results in a smoother pointcloud. This suggests that the accelerometer-based vehicle trajectory is a better initialization for the camera matrices in a 3D reconstruction system.

2.2 3D Reconstruction

Multiple 3D reconstruction methods were tested, but only COLMAP [159] converged on a large number of the available sequences. It should be noted that all libraries were tested on the same three sequences, all containing some moving pedestrians and vehicles and strong forward motion as this is typical for the Cityscapes dataset. The following libraries were tested with the following results: *Open Structure for Motion Library* (OpenSFM) [160]- A Structure from Motion(SFM) system, that is an incremental 3d reconstruction system. Fails to reconstruct the Cityscapes scenes likely because the change in camera rotation is too small between frames. Bundler [161] is also a SFM system. Finds <10 matches, and fails again likely because the images are blurry and the rotational difference between the initial camera views is too small. *OpenCV Structure from motion library* [162] - A SFM library that uses DAISY features [163]. Result of 30 frames - finds relatively few points without a clear structure. See Fig. 5. *VisualSFM* [164] a paralellized SFM pipeline with Bundler. Only a thresholded number of large-scale features are matched across images. This unfortunately fails possibly because of image blur or the lack of distinct large-scale structures in the images. The method is unable to find enough SIFT feature points likely because the images are blurry and finds no verified matches between two stereo images. Finally, VisualSFM cannot handle forward motion, not finding a good initial pair of images with enough matches. *ORBSLAM* [165]- ORB-feature [166] (a fast feature descriptor combining gradient and binary features) based Simultaneous Localization And Mapping (SLAM) system. Finds too few keypoints, likely due to blur and depth threshold. Results in a too sparse reconstruction. *COLMAP* [167, 168]- an incremental SFM and Muti-view stereo(MVS) system. Extracts SIFT [169] features that are exhaustively matched (other matching methods are also available) across all images. Converges for 150 scenes on the training and validation set and 150 scenes on the test set. See Fig. 6. For further details on the different systems see Table 1 and the Appendix.

A number of the reconstruction methods fail to find reliable matches across images, likely because of the motion blur and poor quality of the images as the

Table 1. Overview of the algorithms used by the different SFM and SLAM libraries. See the Appendix for more details.

Method	Image features	Matching algorithm	First view selection	Method for selecting additional views	Bundle Adjustment
OpenSFM	HAFP+HOG	Exhaustive Fast approx. NN	First frames >30% outliers	largest overlap with pointcloud	
Bundler	SIFT	Exhaustive approx NN	large difference in rotation	largest overlap with pointcloud	SPA
OpenCv	DAISY	Exhaustive NN			inexact Newton
VisualSFM	SIFT GPU	Preemptive mathcing	thresholded no. of large features	largest overlap with pointcloud	Multicore BA
ORBSLAM	ORB	Stereo matching closer than 40b	first frames	next frame	Levenberg Marquardt
COLMAP	SIFT	Exhaustive NN	Algorithm of Beder & Steffen	high inlier ratio	PCG

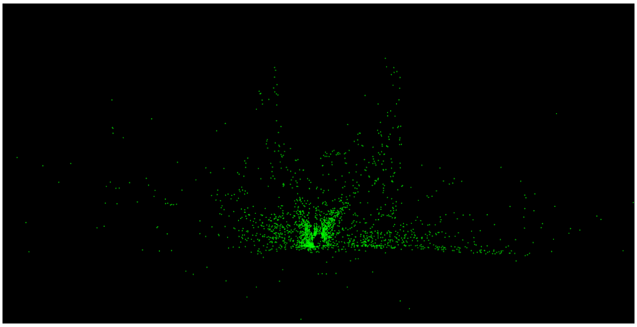


Fig. 5. 3D pointcloud reconstructed by the OpenCV library. There are too few 3D points in the pointcloud to detect what has been reconstructed.

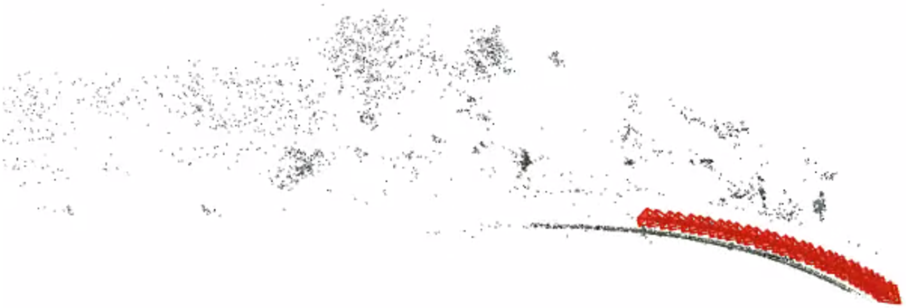


Fig. 6. COLMAP's sparse 3D reconstruction of the scene depicted in Fig. 7 and Fig. 4. The red rectangular pyramids depict the different camera positions, showing correctly that the vehicle traveled on a curved road.

cameras are mounted behind the windscreen of the vehicle. Secondly, the majority of visual 3D reconstruction methods fail at reconstructing in the presence of large forward motion of the camera in particular at fast speeds (i.e. the speed of a car) in the presence of a large number of objects at a large distance to the camera. COLMAP [167, 168] differs mostly from the other methods by the fact that it is modeled for camera views with at times large overlaps; by its outlier robust triangulation, probabilistic new view selection and iteratively applied final Bundle adjustment (BA) alternated with filtering and triangulation. It should, however, be noted that COLMAP is not applicable in real-time, (recently a real-time adaption [159] has become available), and it could not reconstruct all of the Cityscapes scenes (3475 in the training set and 1525 in the test set). The majority of 3D reconstruction methods are not well-fitted to reconstruct images captured from a moving vehicle. This has led to the increased popularity of LiDAR sensors as they can measure directly the distance to objects which is particularly useful in the presence of moving objects (pedestrians, cars, bikers etc.) when only a few images may be available of the object at a particular location.

2.3 Filtering out Non-stationary Objects in the 3D Reconstructions

Moving objects such as cars and pedestrians need to be removed in SFM, to this end the Gated Recurrent Flow Propagation (GRFP) net [170] that is a video segmentation network that utilizes optical flow to stabilize semantic segmentation in video data. The GRFP is used to segment the frames of the Cityscapes sequences.

COLMAP is adapted by adding semantic segmentation as an additional channel (in addition to the 3 RGB channels) describing points during SFM. The camera matrices are initialized based on the accelerometer data. A subsampled sequence of frames from the left camera can be seen in Fig. 3 and the semantic segmentation of the last frame in the left and the right images can be seen in Fig. 7. The resulting sparse reconstruction in Fig. 6 and dense reconstruction can be seen in Fig. 8. Semantic segmentation of the reconstructed 2D points is then transferred to the 3D pointcloud as detailed in the supplementary material of [171]. Note that moving objects are filtered out only during sparse reconstruction, the dense reconstruction is instead filtered for objects with dynamic object labels during voxelisation.

A number of reconstructions are shown in Fig. 9. Some reconstructions correctly recover the structure of the road such as Tübingen, Ulm and Weimar, also in Fig. 10. In general, the reconstruction deteriorates further away from the camera. This can be seen in the reconstruction of Tübingen in Fig. 9 where some of the road (in purple) is misaligned with the rest of the reconstruction and is tilted downwards. This is expected as objects further away from the camera are harder to recognize and estimate the distance to. The reconstructions elongate objects as can be seen in the reconstruction of Tübingen Ulm and Weimar in Fig. 9. COLMAP is however not always successful, when the frames change in viewpoints is small the found reconstruction ends up being flat like in Bremen in Fig. 9 or almost flat like seen in the top view of Darmstadt in Fig. 9.

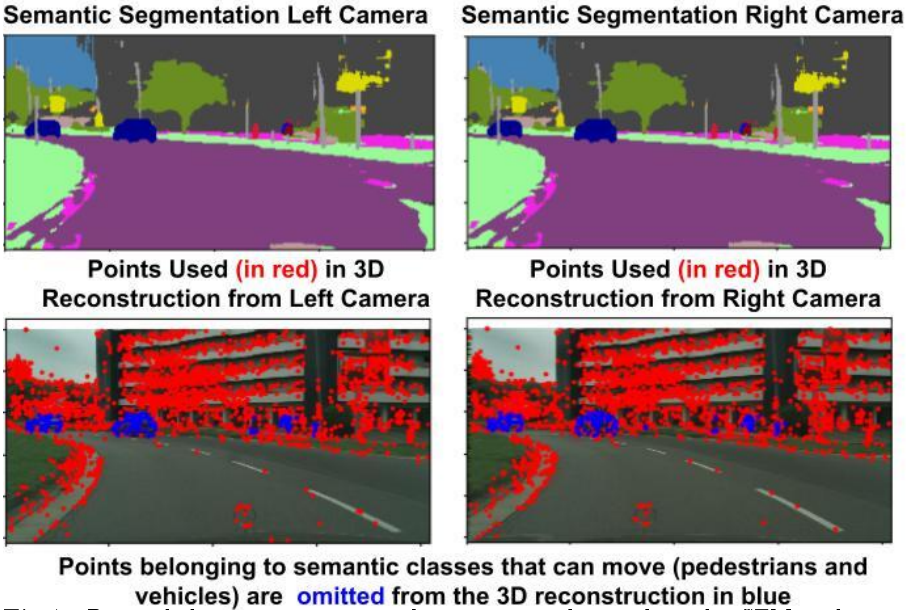

Semantic Segmentation Left Camera
Semantic Segmentation Right Camera
Points Used (in red) in 3D Reconstruction from Left Camera
Points Used (in red) in 3D Reconstruction from Right Camera
Points belonging to semantic classes that can move (pedestrians and vehicles) are omitted from the 3D reconstruction in blue

Fig. 7. Points belonging to moving objects cannot be used in the SFM and must filtered out. In the top row, the semantic segmentation of the left and the right camera images are shown for one frame, and in the bottom row, the points used in the sparse reconstruction of COLMAP are shown. In red points that are included in the SFM are shown. In blue points that are omitted in the SFM (as they belong to semantic classes of pedestrians and vehicles) are shown.

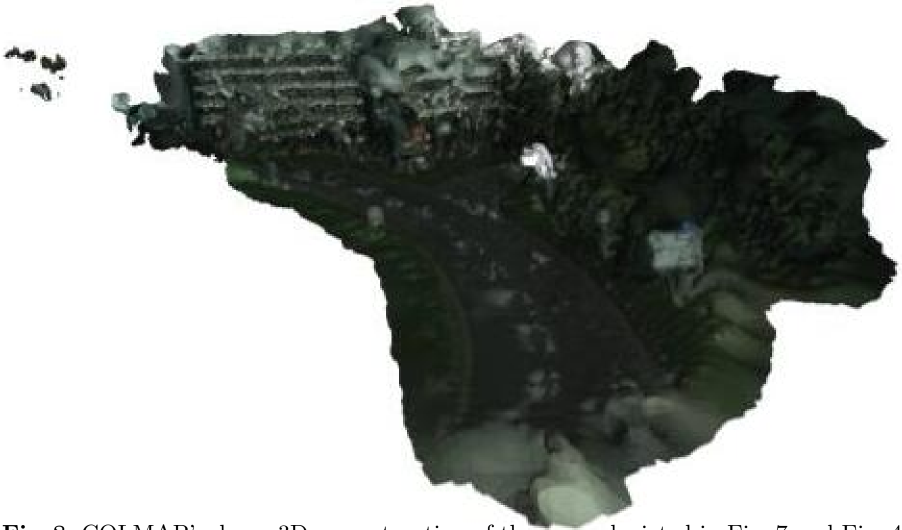


Fig. 8. COLMAP's dense 3D reconstruction of the scene depicted in Fig. 7 and Fig. 4. The dense reconstruction is noisy but the scene is recognizeable.

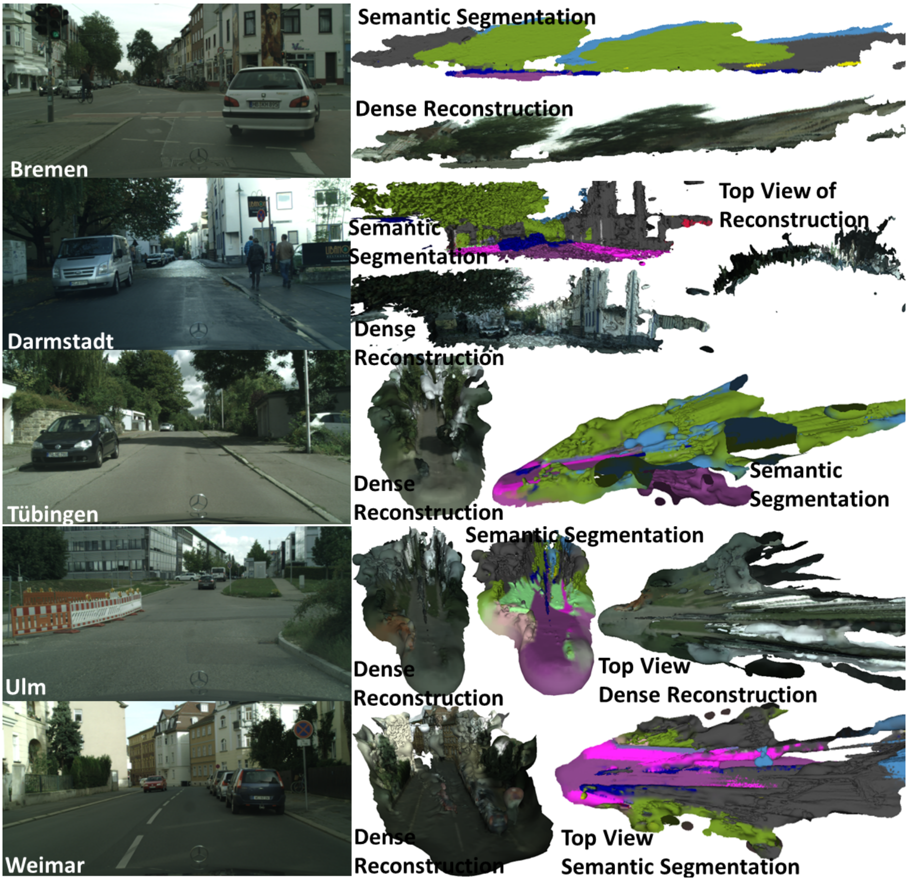


Fig. 9. Dense reconstructions. *First row:* The Bremen sequence's first frame (to the left) and a flat reconstruction (to the right) labeled with semantic segmentation labels (top) and RGB (bottom).*Second row:* The Darmstadt's reconstruction appears fine from the front view (middle) but is flat and curved when viewed from the top (left).*Third row:* Tübingen results in a correct reconstruction of the street close to the camera (middle), but an incorrect estimation of the street topology due to uphill view (right).*Fourth row:* Ulm is reconstructed correctly with a patch of grass separating the road and the sidewalk as seen in front (middle) and top view(to the right).*Fifth row:* Correctly reconstructed street shape as seen in front (middle) and top view (to the right).

Table 2. Frequency of the semantic classes on the CARLA dataset and accuracy of the *Pointnet++* for the different semantic classes. The semantic classes are in the order of decreasing frequency. Objects of class wall obtains the lowest accuracy.

Class	**vegetation**	**building**	**road**	**sidewalk**	**fence**	**static**	**wall**	**pole**	**sign**
Frequency %	37.62	18.50	17.87	17.55	3.00	2.81	1.71	0.83	0.09
Accuracy	0.93	0.86	0.88	0.67	0.84	0.51	**0.50**	0.88	0.71

To directly label a pointcloud experiments were conducted with the popular pontcloud segmentation network *Pointnet++* [172]. The Cityscapes has no GT segmented pointclouds, so a model that was finetuned on CARLA [173] generated pointclouds was tested but resulted in confused labels. Finetuning of Pointnet++ [172] on the synthetic CARLA dataset(from [171]) resulted in a low mean average class accuracy of 0.62, with per class results shown in Table 2. The classes that occur seldom get low accuracy, so objects such as traffic sign get almost always incorrectly labelled. It is also worth noting that strangely enough points belonging to walls are correctly marked only in half of the occurences. In general the results suggest that Pointnet++ results are not on bar with labelling 3D reconstructions according to projections of 2D semantic maps. It is possible that more recent methods [174–181] could improve the results.

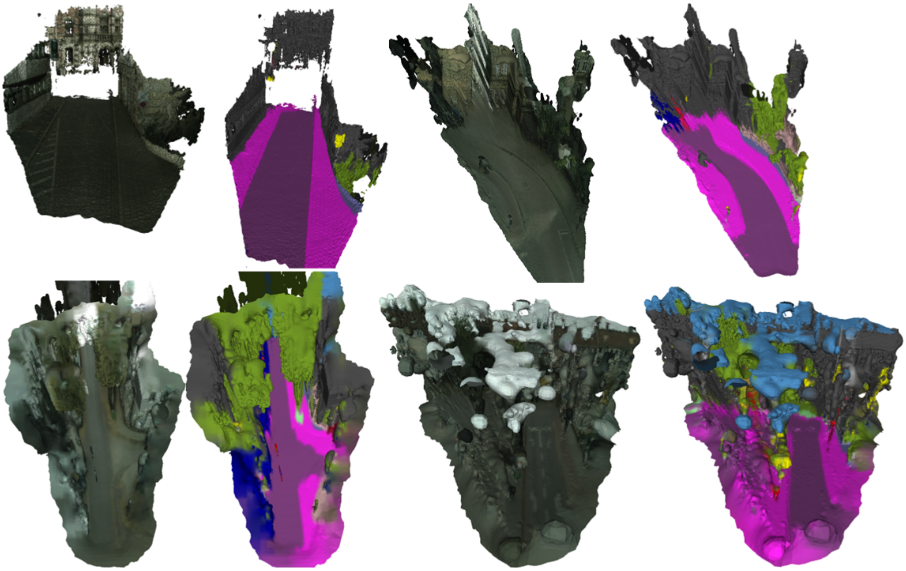


Fig. 10. Additional dense reconstructions from Bremen showing that noise levels vary but the street shape is often successfully reconstructed.

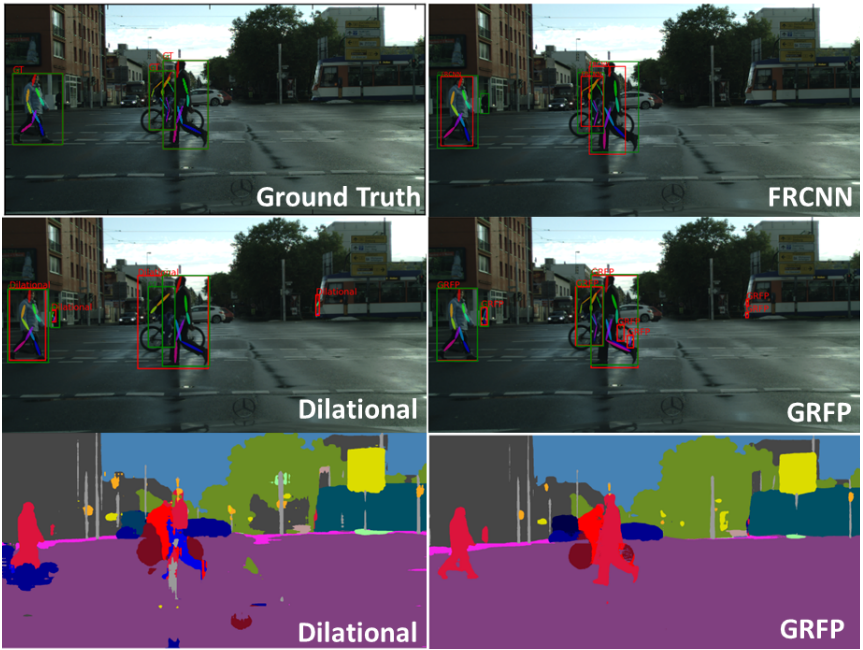


Fig. 11. Segmentation, BBoxes and 2D joint position estimates of *OpenPose* with *Dilational*, *GRFP* and *FRCNN*. *Dilational* net and *GRFP* manage to separate different pedestrians who are visually close by but also introduce false positives. *GRFP* produces cleaner BBoxes than *Dilational*.

3 Pedestrian sensing

Detecting humans is hard because they are relatively small in traffic images, they vary in physique and visual qualities depending on the human's pose and clothing, and they change their positions from frame to frame. The fact that most popular object detectors are biased to detect close-up objects centered in an image makes them ill-fitted to traffic data because in traffic humans appear often a distance from the camera. We compared object detection, segmentation, and human pose detecting network's ability to detect pedestrians on the Cityscapes dataset [158] by comparing the detected pedestrian's BBox overlap with BBoxes generated from GT segmentations. The tested methods are

- *DilationalNet-10* [182]- A popular semantic segmentation network with dilated convolutions for larger receptive field.
- *The Gated Recurrent Flow Propagation(GRFP) [170]* - A temporally smoothed video segmentation network showing temporally smooth result on the Cityscapes dataset [158].
- *Faster-RCNN(FRCNN)* [183] - A popular object detection network with high throughput and good performance on benhcmarks.
- *OpenPose* [82, 83] - A popular multi human 2D pose estimating network, that has a runtime that scales well with increasing number of visible humans.

Table 3. The number of true positive, false negative and false positive BBoxes on the training and validation sets of Cityscapes for the 20th frame. The two strongest contenders for pedestrian detection are the *GRFP* and the *FRCNN*. The *GRFP* produces the largest number and area of true positives, and the *FRCNN* produces the smallest number and area of false positives and negatives.

Model	**True positives**	**Average TP area**	**False positives**	**Average FP area**	**False Negatives**	**Average FP area**
Dilational	2887	0.699	7,516	0.008	16,664	0.016
GRFP	**3588**	**0.707**	14,317	0.01	**15,980**	0.015
FRCNN	2952	0.706	578	**0.001**	16,522	**0.009**
OpenPose	165	0.682	**343**	0.003	18,997	0.017

In Table 3 it can be seen that *FRCNN* produced the smallest false positive(FP) and false negative(FN) average BBox area, but has the second highest true positive(TP) Intersection over Union (IoU) area. Because the areas of the BBoxes vary we present both the FP, FN, and TP counts and areas (normalized with respect to the total GT BBox areas), to observe how many individuals are detected versus how much of the visual area is covered by the pedestrians. *FRCNN* is accurate in detecting large BBoxes, and it detects on average larger BBoxes than the GRFP as seen in Fig. 12 *Left*. *GRFP* on the other hand is better at capturing distant pedestrians but also produces a large amount of FPs. Based on this *FRCNN* is the most suitable pedestrian detector as it is the most accurate in detecting pedestrians close to the vehicle, these pedestrians have the highest risk of being run over if undetected.

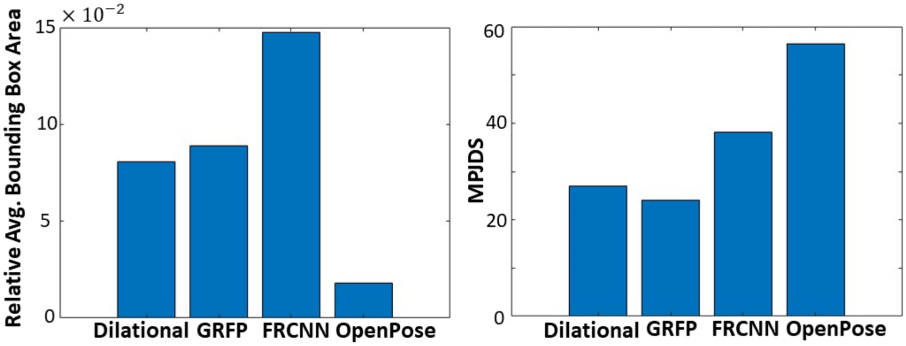


Fig. 12. *Left:* The average relative BBox areas of the different pedestrian detection methods. FRCNN detects on average the largest BBoxes and OpenPose the smallest. *Right:* The average distance from estimated joint positions to human mask (from GT segmentation). GRFP's human BBoxes result in the lowest distance from estimated joint positions to human mask and OpenPose in the largest.

OpenPose is used to estimate the articulated human 2D pose on the BBoxes found by *Dilational*, *GRFP* and *FRCNN* and on the whole image. We introduce the Mean Per Joint Distance to Segmentation(MPJDS) metric which is the average distance from an estimated 2D joint position to a pedestrian or biker segmentation mask. The MPJDS is an approximate measure of how accurately *OpenPose* can estimate the pose of a pedestrian present in the BBoxes found by the different models, results are shown in Fig. 12 *Right*. *GRFP* results is the smallest error likely because *GRFP* detects smaller BBoxes than *FRCNN* resulting in smaller absolute errors. *OpenPose* applied on the whole image detects pedestrians that appear to be far away from the camera, but fails to estimate their pose, resulting in large joint errors for small BBoxes. Eventhough *OpenPose* has presented impressive results it fails to detect multiple pedestrians in traffic scenarios without a separate pedestrian detector.

Table 4. The FRCNN detects fewer pedestrians and bikers than Dilational but results in a lower MPJDS, suggesting that FRCNN detects pedestrians that are clearer.

Model	MPJDS	MPJDS norm.	Number of pedestrians	Number of bikers	Crossover pedestrians	Crossover bikers
GT	32.50	0.99	**1,803**	**17,415**	**1.0**	**1.0**
Dilational	27.17	0.87	850	4,572	0.94	0.91
FRCNN	**10.74**	**0.38**	392	2,888	0.86	0.85

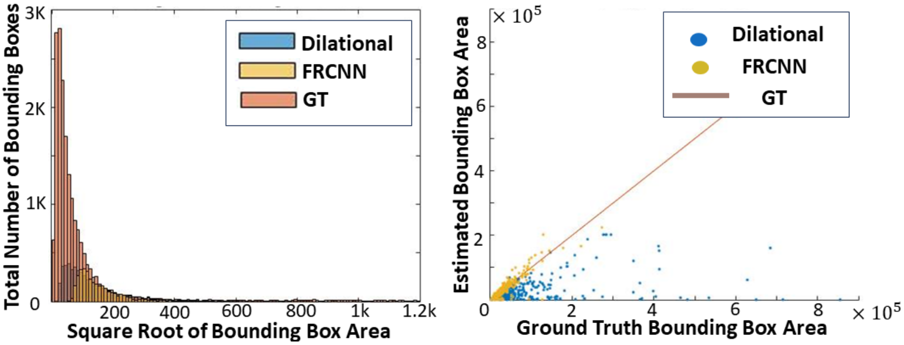


Fig. 13. *Left: FRCNN* detects only large BBoxes. *Dilational* can detect smaller BBoxes, but many GT bbes are undetected by both methods. *Right:* The BBoxes found by *FRCNN* are in general representative of the GT BBox sizes, while *Dilational* underestimates BBox sizes.

To study the accuracy of *OpenPose* on BBoxes that truly contain a pedestrian we filter out the BBoxes that have at least 50% cross-over with the GT BBoxes, results are shown in Table 4. By *cross-over* is meant the percentage that the GT BBox intercepts the estimated BBox with. If an estimated BBox intercepts with a number of GT BBoxes then only the highest cross-over is recorded. The MPJDS of *OpenPose* applied on the GT BBoxes in Table 4 is much larger than that of the other two methods because the GT contains pedestrians who are hard to spot in the images (distant or occluded as seen in sizes in Fig. 14). These pedestrians go unnoticed by the *Dilational* and *FRCNN*. Even though *FRCNN* has a lower crossover percentage than the *Dilational* it obtains the lowest MPJDS suggesting that *FRCNN* detects the most clearly visible pedestrians. In Table 4 it can be seen that *FRCNN* detects fewer pedestrians and bikers *Dilational*, but results in a much lower MPJDS. The MPJDS of *FRCNN* is high even though the crossover is lower than for the other models. This is likely because *FRCNN* finds pedestrians that are closer to the camera and thus clearer, omitting smaller pedestrians that are captured by *Dilational* as seen in Fig. 13*Left*.

The *FRCNN* correctly estimates the GT BBox sizes as seen in Fig. 13 *Right*, but *Dilational* underestimates BBox sizes showing the pedestrians only partially and therefore has a higher MPJDS than *FRCNN* even for large GT BBoxes as seen in Fig. 14. *Dilational* net can detect smaller pedestrians because it has been trained on the Cityscapes dataset in difference to *FRCNN*. It is possible that *FRCNN* cannot detect small pedestrians because it has been trained with larger anchor sizes than the visible pedestrians. An example showing close up occluded pedestrians comparing the *GRFP*, *Dilational* net, *FRCNN* and GT can be seen in Fig. 11.

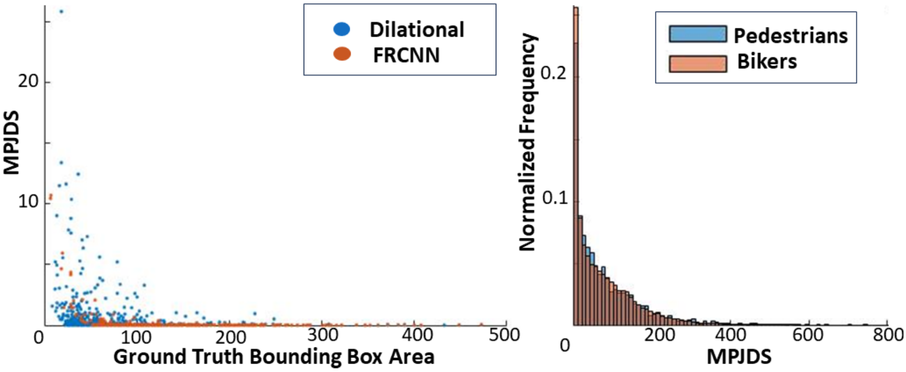


Fig. 14. *Left:* The MPJDS is plotted against the BBox area for the BBoxes found by the different methods. *FRCNN* finds larger BBoxes and results in lower MPJDS for these larger BBoxes than for the BBoxes found by *Dilational* net, suggesting that FRCNN finds easier to detect pedestrians. *Right:* Histogram of MPJDS distribution of the *FRCNN* detections shows that most errors are small. There appear to be no outliers with large MPJDS error.

The *Dilational* net has trouble differentiating between the labels: “pedestrian”, “biker”, and “bike” as seen in Fig. 11. Therefore *Dilational* net BBoxes are fitted with skeletons after allowing biker and pedestrian labels to be interpreted as the same label. Also, bike labels are allowed to be interpreted as human if they are in connection to rider or pedestrian labels. In Fig. 11 it can also be

Table 5. By introducing smallest size constraints on the BBoxes the number of false positives can be reduced significantly.

Model	GT	GT Filtered	Dilational	GRFP
True Positives	**15,934**	8,986	3,316	3,643
False Positives	0	0	**235**	1,038

seen that only a small change in the placement of the BBox around a pedestrian results in variations in the estimate of the pose, showing that *OpenPose* is not robust to errors in pedestrian BBox placement.

Further on crowded images the *FRCNN* has superior performance because it can separate between pedestrians as seen in Fig. 22, and *GRFP* is superior in distant pedestrian detection as seen in Fig. 23. In the crowded scene the pose estimator gets confused with BBoxes because for some pedestrians only a single body part is visible, and it is hard for the pose estimator to detect that the body part is not just a blurry image of a human. Videos of sample pose estimations can be found at `https://youtu.be/qpxpdtHbbGA` where it can be seen that the pose estimations are not temporally smooth for any of the proposed methods. To avoid false detections of the *Dilational* net and *GRFP* we remove any BBoxes that are smaller than 7 pixels in width and 25 pixels in height. This results in a decrease in the number of false positives for *GRFP* and the *Dilational* as seen in Table 5.

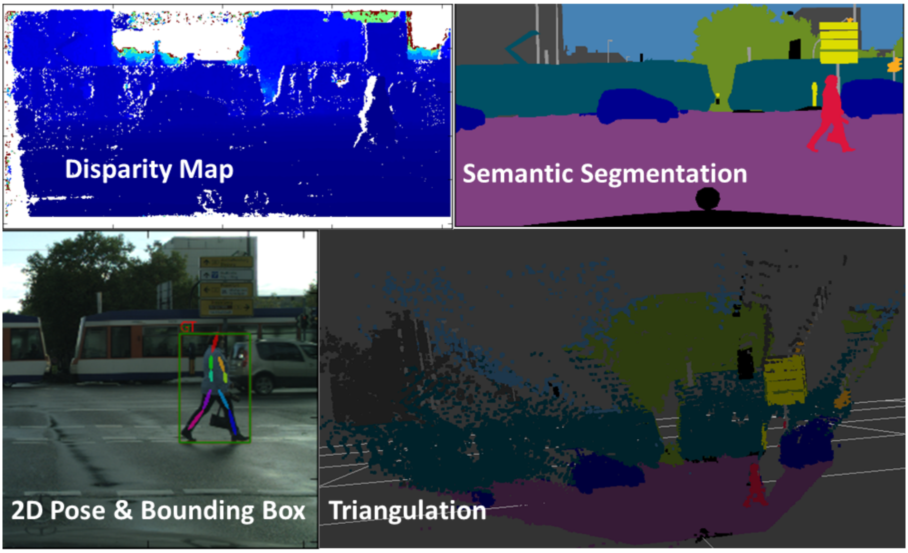


Fig. 15. A scene's semantic segmentation, disparity map, triangulation of the frame from the disparity map and the triangulated human pose.

3.1 Reconstructing Pedestrians

Triangulation can be used to reconstruct the human 3D poses from the 2D poses found with the dataset's disparity maps. This however results in a noisy pose estimate as seen in Fig. 15. Stereo triangulation results in a noisy 3D reconstruction. The triangulated 2D joint positions can therefore receive incorrect depth resulting in implausible 3D poses. Often a body joint receives the depth of the background resulting in an elongated limb, as seen in Fig. 16. To correct such errors we apply a threshold to limb lengths, proportioned according to the hip length or back-bone length of the pedestrian. This is not robust because the hip and backbone length are estimated according to a standard skeleton from Human3.6M [90]. The limb length can be estimated according to an average skeleton relative to the height of a person. The height of the person can be roughly approximated from the bounding box height, with the downside that BBox height is pose-dependent.

The corrected skeleton may still suggest a physically implausible pose. To correct this the nearest neighbour plausible pose is found from Human3.6M [90]. To find an outlier robust estimate of the nearest neighbor a thresholded loss is applied. Procrusets analysis is used to find the optimal alignment between the skeletons. The final corrected pose with scaling according to hip or backbone are shown in Fig. 16.

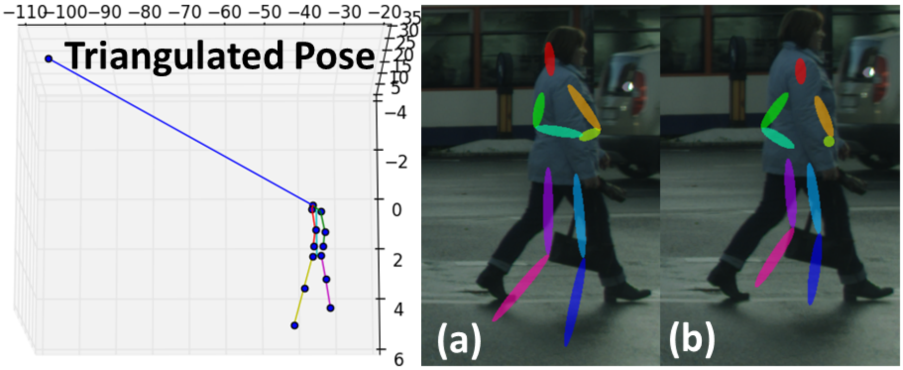


Fig. 16. *To the Left:* Incorrectly triangulated head position, full image in Fig. 15. All axis are in meters. Procrustes corrected skeletons with (a) scaled according to backbone length and (b) scaled according to hipbone length. Neither of the scalings give the desired result.

It is clear that the scaling and rotation of the resulting 3D pose are imperfect. When triangulating the pose for each frame jitter can be expected between frames due to noise. Therefore a monocular single-person 3D pose estimator *Deep Multitask Architecture for Fully Automatic 2D and 3D Human Sensing* (DMHS) [184] is tested as well.

The *DMHS* is applied to GT and *FRCNN* see Fig. 17 *left* and *right* respectively. At times *FRCNN* provides a too small BBox, by increasing the boundary

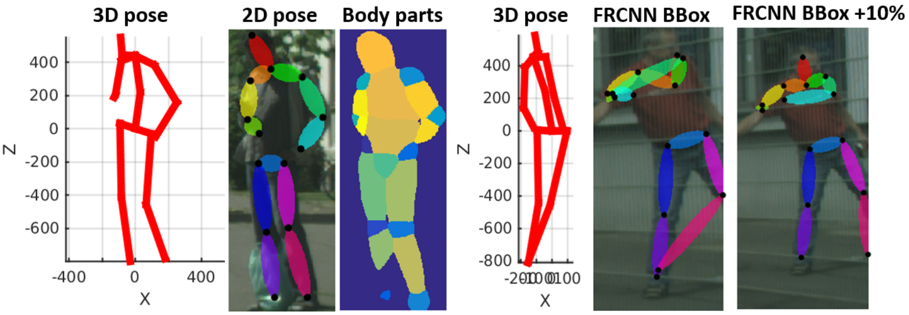


Fig. 17. To the *Left DMHS* estimates the pose of a pedestrian decently correctly when the pedestrian is clearly visible. The 3D pose (scale in cm), 2D pose and Body part segmentation are shown respectively. To the *right* BBox enlargement improves the pose estimation when some limbs are not visible.

(with 10%) the results improve, see Fig. 17 *right*. The pose detector fails when multiple people are present in the BB, or when the pedestrians are poorly visible. Eventhough *FRCNN* detects close-up pedestrians, still very few BBoxes are clearly visible and thus few obtain accurate pose estimations.

3.2 Reconstructing Vehicles

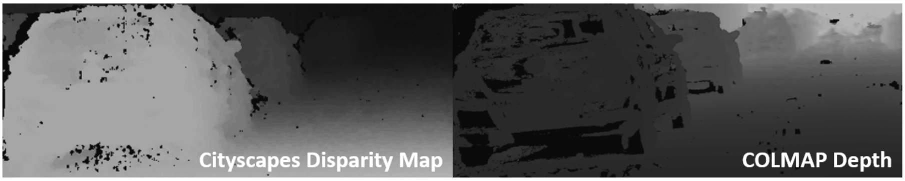


Fig. 18. To the *left* Cityscapes disparity map and *right* COLMAP depth estimate of a *FRCNN* BBOx of a car clearly contain multiple cars.

The *FRCNN* has trouble separating multiple instances of cars when cars are parked in a row as seen in Fig. 18 due to the large visual overlap. During triangulation for simplicity we model cars found by the detection model by fixed sized 3D BBoxes. As a result during 3D triangulation multiple vehicles that are visible in one 2D BBox get replaced by one 3D BBox with an average disparity for all ofthe cars visible in the 2D bbox. This results in an incorrect 3D reconstruction of the scene. To improve this Path Aggregation Network for Instance Segmentation (PANNET) [185] an *FRCNN* architecture based instance segmentation network is utilized instead. Sample segmentation showing the correct instance segmentation of *PANNET* is shown in Fig. 3.2.

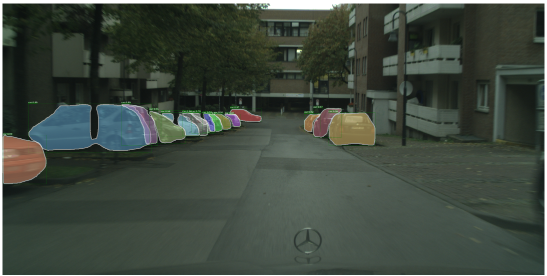


Fig. 19. PANNET correctly separates different parked cars even in the presence of occlusions.

4 Developments in the field

The presented results were developed from 2016 to 2018. In parallel to our work [186, 187] noted that detecting objects at a distance is hard, and in particular human detection at a distance or in the presence of occlusions has gained popularity [188–195]. Further it has been noted by [196] that a number of human detection models do not generalize across datasets. Optimal alignment of BBoxes has also been studied in [197]. The run-time accuracy trade-off of object detection methods aimed to be utilized on AVs is studied in [198].

More compact representations of scenes are often utilized in AV planning containing either rasterized graphs with local context [199] or BEV representations [124]. This is suitable as planning must occur fast, but we still believe that articulated human motion ought to be included in the representation. The advantage of utilizing 3D pointclouds and images is that the 3D reconstructed scenes can easily be utilized to train AVs on augmented data [119, 171, 200].

Human and object detection in traffic from alternative sensors like Radar [201], LiDAR [143, 202–206], event cameras [207] have been studied as a way to boost object and human detection performance. Methods to improve low-quality image data by reducing motion blur [208], increasing image quality in low light or low-resolution images [209] performing detection on RAW images [210], or object in-painting to recover from occlusions [211] could possibly greatly improve human sensing in real traffic data.

Further LiDAR sensors have gained popularity to avoid the difficult task of estimating the depths of moving objects. Unfortunately sensing of articulated humans in LiDAR [212] has not yet caught up with the methods developed to sense humans in videos. There exist methods that combine LiDAR and RGB fusion [37, 145, 213–215] for pedestrian detection and trajectory forecasting, the same could be done for human pose forecasting. Human pose estimation has developed greatly with more models that fit meshes to human bodies to densely estimate human pose and semantic mask [184, 212, 216–218], methods that reason about the physics of the estimated pose have been developed [219–221], as well as methods that utilize temporal constraints [222–224]. Still, the majority of articulated human sensing methods are developed on visuals where humans are centered in the images [184, 216, 217, 219–221], leaving a gap to traffic data where most humans are relatively far away. Human pose estimation and forecasting are more frequently being combined with segmentation [225–227], tracking [225, 228, 229] gait recognition [230] and camera pose estimation [231, 232]. Human motion can be very informative in traffic and pedestrian behavior modeling can even be used to detect vehicles in blind spots in traffic [233]. To maximally utilize the available information in human motion, methods that are robust to variations in the human physique and behavior need to be developed, but this is hard due to the relative lack of data.

Vehicle orientation and shape estimation techniques are amongst others in images [234], in LiDAR [235]. A method that jointly performs semantic segmentation and 3D reconstructions could benefit both tasks [236].

Improved depth estimation of pedestrians and vehicles through a LiDAR data-like data representation Pseudo-LiDAR is studied in [237, 238] Finally, 3D reconstruction methods have developed greatly with more learning integrated into the 3D reconstruction pipeline [239–243], from learned monocular depthmaps [244], to learned 3D reconstruction features [245] to 3d matching [246] and the visually pleasing NERF-based methods [247, 248]. Semantic segmentation, tracking, and object detection methods are also becoming less supervised utilizing learned matching and language model based labels [144, 249–256]. Combining different visual tasks like object detection semantics segmentation, tracking with 3D modeling has seen success in [144, 257]. This is quite natural because as seen in Fig. 18 the two tasks are closely intertwined and information sharing may help in both directions.

Traffic datasets that are focused on pedestrians have become more abundant [258–269], but there exist only datasets with estimated articulated labels for pedestrians [157]. Even though progress has been made on marker-less human motion capture [116] the methods need to be made robust for multiple humans at a distance and in the presence of occlusions. In parallel to our work, a study [270] on occlusion rates in pedestrian bounding boxes on the Cityscapes dataset was performed. We note that [270] may be treated as complementary to the work presented here that focuses on the task of 3D human pose reconstruction rather than just bounding box occlusions.

5 Conclusion

None of the discussed methods of 3D reconstructing human pose are robust enough to be utilized to forecast human motion for assisted driving. This is because there is a gap in performance for human sensing methods between the datasets used in standard benchmarks and the performance on real traffic data, suggesting that benchmarks of human motion sensing are not representative of utilization in traffic. Instead, traffic-based articulated 3D human sensing benchmarks should be developed. Available 3D human pose datasets in the wild [116] still lack in distant pedestrians under poor lighting conditions, or provide only approximate human poses [157]. To make articulated human sensing robust temporal smoothness, consistent use of an individual's estimated limb lengths, foreseeing typical motions given the human's environment, and understanding of the physical constraints of the human body should be solved simultaneously as the problems share information. So far a number of methods have solved some of these subproblems, but a unifying method is still to be developed. As a result to the lack of a robust articulated humans sensing method a large number of existing autonomous vehicle planning models [5–30, 37, 38, 42–46, 199] treat pedestrians by their bounding boxes, thus omitting the motion cues available in human pose and therefore ignoring available future motion cues. If robust and complete articulated human sensing methods are developed, then complete human forecasting methods may be developed and utilized in the planning stages of AVs.

References

1. Paden, B., Čáp, M., Yong, S. Z., Yershov, D. & Frazzoli, E. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles* **1,** 33–55 (2016).
2. Zhu, Z. & Zhao, H. A survey of deep RL and IL for autonomous driving policy learning. *IEEE Transactions on Intelligent Transportation Systems* **23,** 14043–14065 (2021).
3. Ye, F., Zhang, S., Wang, P. & Chan, C.-Y. A Survey of Deep Reinforcement Learning Algorithms for Motion Planning and Control of Autonomous Vehicles. *IEEE Transactions on intelligent transportation systems.* (2021).
4. Wei, L., Li, Z., Gong, J., Gong, C. & Li, J. *Autonomous Driving Strategies at Intersections: Scenarios, State-of-the-Art, and Future Outlooks* in *24th IEEE International Intelligent Transportation Systems Conference, ITSC 2021, Indianapolis, IN, USA, September 19-22, 2021* (IEEE, 2021), 44–51.
5. Bae, I. & Jeon, H.-G. *A Set of Control Points Conditioned Pedestrian Trajectory Prediction* in *Proceedings of the AAAI Conference on Artificial Intelligence* **37** (2023), 6155–6165.
6. Li, J. *et al.* EvolveHypergraph: Group-Aware Dynamic Relational Reasoning for Trajectory Prediction. *arXiv preprint arXiv:2208.05470* (2022).
7. Chen, Y., Liu, C., Mei, X., Shi, B. E. & Liu, M. *HGCN-GJS: Hierarchical Graph Convolutional Network with Groupwise Joint Sampling for Trajectory Prediction* in *2022 IEEE/RSJ IROS* (2022), 13400–13405.
8. Djuric, N. *et al. Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving* in *Proceedings of the IEEE/CVF WACV* (2020), 2095–2104.
9. Liang, M. *et al. Pnpnet: End-to-end perception and prediction with tracking in the loop* in *Proceedings of the IEEE/CVF Conference on CVPR* (2020), 11553–11562.
10. Luo, Y., Cai, P., Lee, Y. & Hsu, D. Gamma: A general agent motion model for autonomous driving. *IEEE Robotics and Automation Letters* **7,** 3499–3506 (2022).

11. Ma, Y. *et al. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents* in *Proceedings of the AAAI Conference on artificial intelligence* **33** (2019), 6120–6127.
12. Zhu, Y. *et al.* Robust trajectory forecasting for multiple intelligent agents in dynamic scene. *arXiv preprint arXiv:2005.13133* (2020).
13. Sriram, N., Liu, B., Pittaluga, F. & Chandraker, M. *Smart: Simultaneous multi-agent recurrent trajectory prediction* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16* (2020), 463–479.
14. Zhao, T. *et al. Multi-agent tensor fusion for contextual trajectory prediction* in *Proceedings of the IEEE/CVF Conference on CVPR* (2019), 12126–12134.
15. Luo, K. *et al.* Safety-oriented pedestrian motion and scene occupancy forecasting. *arXiv preprint arXiv:2101.02385* (2021).
16. Shen, M., Habibi, G. & How, J. P. *Transferable pedestrian motion prediction models at intersections* in *2018 IEEE/RSJ IROS* (2018), 4547–4553.
17. Fang, L., Jiang, Q., Shi, J. & Zhou, B. *Tpnet: Trajectory proposal network for motion prediction* in *Proceedings of the IEEE/CVF Conference on CVPR* (2020), 6797–6806.
18. Park, S. H. *et al. Diverse and admissible trajectory forecasting through multimodal context understanding* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16* (2020), 282–298.
19. Van der Heiden, T., Shankar-Nagaraja, N., Weiß, C. & Gavves, E. Safe-Critic: Collision-Aware Trajectory Prediction.
20. Yang, T., Nan, Z., Zhang, H., Chen, S. & Zheng, N. *Traffic agent trajectory prediction using social convolution and attention mechanism* in *2020 IEEE IV* (2020), 278–283.
21. Cheng, H., Liao, W., Yang, M. Y., Rosenhahn, B. & Sester, M. Amenet: Attentive maps encoder network for trajectory prediction. *ISPRS Journal of Photogrammetry and Remote Sensing* **172,** 253–266 (2021).
22. Giuliari, F., Hasan, I., Cristani, M. & Galasso, F. *Transformer networks for trajectory forecasting* in *2020 25th ICPR* (2021), 10335–10342.
23. Anderson, C., Vasudevan, R. & Johnson-Roberson, M. Off the Beaten Sidewalk: Pedestrian Prediction in Shared Spaces for Autonomous Vehicles. *IEEE Robotics and Automation Letters* **5,** 6892–6899 (2020).
24. Salzmann, T., Ivanovic, B., Chakravarty, P. & Pavone, M. *Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data* in *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XVIII* **12363** (Springer, 2020), 683–700.
25. Hamandi, M., D'Arcy, M. & Fazli, P. *DeepMoTIon: Learning to Navigate Like Humans* in *28th IEEE RO-MAN 2019, New Delhi, India, October 14-18, 2019* (IEEE, 2019), 1–7.

26. Yao, X., Zhang, J. & Oh, J. Following Social Groups: Socially Compliant Autonomous Navigation in Dense Crowds. *arXiv preprint arXiv:1911.-12063* (2019).
27. Li, C., Meng, Y., Chan, S. H. & Chen, Y.-T. *Learning 3d-aware egocentric spatial-temporal interaction via graph convolutional networks* in *2020 IEEE ICRA* (2020), 8418–8424.
28. Chen, Y., Liu, C., Shi, B. E. & Liu, M. Robot Navigation in Crowds by Graph Convolutional Networks With Attention Learned From Human Gaze. *IEEE Robotics and Automation Letters* **5,** 2754–2761 (2020).
29. Ivanovic, B., Leung, K., Schmerling, E. & Pavone, M. Multimodal Deep Generative Models for Trajectory Prediction: A Conditional Variational Autoencoder Approach. *IEEE Robotics and Automation Letters* **6,** 295–302 (2021).
30. Girgis, R. *et al. Latent Variable Sequential Set Transformers for Joint Multi-Agent Motion Prediction* in *The Tenth ICLR, ICLR 2022, Virtual Event, April 25-29, 2022* (OpenReview.net, 2022).
31. Tang, B. *et al.* Collaborative uncertainty benefits multi-agent multi-modal trajectory forecasting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).
32. Huang, X. *et al.* DiversityGAN: Diversity-aware vehicle motion prediction via latent semantic sampling. *IEEE Robotics and Automation Letters* **5,** 5089–5096 (2020).
33. Jiang, B. *et al.* Perceive, interact, predict: Learning dynamic and static clues for end-to-end motion prediction. *arXiv preprint arXiv:2212.02181* (2022).
34. Tang, C. & Salakhutdinov, R. R. Multiple futures prediction. *NeuRIPS* **32** (2019).
35. Manish, Dohare, U. & Kumar, S. *A Survey of Vehicle Trajectory Prediction Based on Deep Learning Models* in *Proceedings of Third International Conference on Sustainable Expert Systems: ICSES 2022* (2023), 649–664.
36. Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B. & Moutarde, F. *THOMAS: Trajectory Heatmap Output with learned Multi-Agent Sampling* in *ICLR* (2022).
37. Li, L. L. *et al. End-to-end contextual perception and prediction with interaction transformer* in *2020 IEEE/RSJ IROS* (2020), 5784–5791.
38. Rhinehart, N., McAllister, R., Kitani, K. & Levine, S. *Precog: Prediction conditioned on goals in visual multi-agent settings* in *Proceedings of the IEEE/CVF ICCV* (2019), 2821–2830.
39. Zeng, W. *et al. DSDNet: Deep Structured Self-driving Network* in *Computer Vision – ECCV 2020* (Springer International Publishing, Cham, 2020), 156–172.
40. Casas, S., Gulino, C., Liao, R. & Urtasun, R. *Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data* in *2020 IEEE ICRA* (2020), 9491–9497.

41. Messaoud, K., Deo, N., Trivedi, M. M. & Nashashibi, F. *Trajectory prediction for autonomous driving based on multi-head attention with joint agent-map representation* in *2021 IEEE IV* (2021), 165–170.
42. Yao, Y., Atkins, E., Johnson-Roberson, M., Vasudevan, R. & Du, X. Bitrap: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation. *IEEE Robotics and Automation Letters* **6,** 1463–1470 (2021).
43. Huang, Z., Hasan, A., Shin, K., Li, R. & Driggs-Campbell, K. Long-term pedestrian trajectory prediction using mutable intention filter and warp LSTM. *IEEE Robotics and Automation Letters* **6,** 542–549 (2020).
44. Deo, N. & Trivedi, M. M. Trajectory forecasts in unknown environments conditioned on grid-based plans. *arXiv preprint arXiv:2001.00735* (2020).
45. Zhao, H. *et al. Tnt: Target-driven trajectory prediction* in *CoRL* (2021), 895–904.
46. Mangalam, K. *et al. It is not the journey but the destination: Endpoint conditioned trajectory prediction* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16* (2020), 759–776.
47. Camara, F. *et al.* Pedestrian models for autonomous driving Part I: low-level models, from sensing to tracking. *IEEE Transactions on Intelligent Transportation Systems* **22,** 6131–6151 (2020).
48. Camara, F. *et al.* Pedestrian models for autonomous driving part ii: high-level models of human behavior. *IEEE Transactions on Intelligent Transportation Systems* **22,** 5453–5472 (2020).
49. Rasouli, A. & Tsotsos, J. K. Autonomous vehicles that interact with pedestrians: A survey of theory and practice. *IEEE Transactions on Intelligent Transportation Systems* **21,** 900–918 (2019).
50. Lyu, K., Chen, H., Liu, Z., Zhang, B. & Wang, R. 3D human motion prediction: A survey. *Neurocomputing* **489,** 345–365 (2022).
51. Zhang, X. *et al.* Finding critical scenarios for automated driving systems: A systematic mapping study. *IEEE Transactions on Software Engineering* **49,** 991–1026 (2022).
52. Ding, W., Lin, H., Li, B. & Zhao, D. *CausalAF: Causal Autoregressive Flow for Safety-Critical Driving Scenario Generation* in *To appear in Proceedings of the 2022 CoRL* (PMLR, 2022).
53. Abeysirigoonawardena, Y., Shkurti, F. & Dudek, G. *Generating Adversarial Driving Scenarios in High-Fidelity Simulators* in *In Proceedings of the 2019 IEEE ICRA* (IEEE, 2019), 8271–8277.
54. Abdessalem, R. B., Panichella, A., Nejati, S., Briand, L. C. & Stifter, T. *Testing autonomous cars for feature interaction failures using many-objective search* in *In Proceedings of the 33rd IEEE/ACM International Conference on Automated Software Engineering* (ACM, 2018), 143–154.
55. Li, Y., Tao, J. & Wotawa, F. Ontology-based test generation for automated and autonomous driving functions. *Information and Software Technology* **117** (2020).

56. Abdessalem, R. B., Nejati, S., Briand, L. C. & Stifter, T. *Testing advanced driver assistance systems using multi-objective search and neural networks* in *In Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering* (ACM, 2016), 63–74.
57. Zhong, Z., Kaiser, G. & Ray, B. Neural Network Guided Evolutionary Fuzzing for Finding Traffic Violations of Autonomous Vehicles. *IEEE Transactions on Software Engineering* (2022).
58. Bussler, A., Hartjen, L., Philipp, R. & Schuldt, F. *Application of evolutionary algorithms and criticality metrics for the verification and validation of automated driving systems at urban intersections* in *In Proceedings of the 2020 IEEE IV* (2020), 128–135.
59. Almanee, S., Wu, X., Huai, Y., Chen, Q. A. & Garcia, J. scenoRITA: Generating Less-Redundant, Safety-Critical and Motion Sickness-Inducing Scenarios for Autonomous Vehicles. *arXiv preprint arXiv:2112.09725* (2021).
60. Ding, W., Chen, B., Li, B., Eun, K. J. & Zhao, D. Multimodal Safety-Critical Scenarios Generation for Decision-Making Algorithms Evaluation. *IEEE Robotics and Automation Letters* **6,** 1551–1558 (2021).
61. Parashar, P., Cosgun, A., Nakhaei, A. & Fujimura, K. Modeling Preemptive Behaviors for Uncommon Hazardous Situations From Demonstrations. *arXiv preprint arXiv:1806.00143* (2018).
62. Zhong, Z. *et al.* A survey on scenario-based testing for automated driving systems in high-fidelity simulation. *arXiv preprint arXiv:2112.00964* (2021).
63. Demetriou, A., Alfsvåg, H., Rahrovani, S. & Chehreghani, M. A Deep Learning Framework for Generation and Analysis of Driving Scenario Trajectories. *arXiv preprint arXiv:2007.14524* (2020).
64. Nishiyama, D. *et al. Discovering Avoidable Planner Failures of Autonomous Vehicles using Counterfactual Analysis in Behaviorally Diverse Simulation* in *In Proceedings of 23rd IEEE International Conference on Intelligent Transportation Systems* (IEEE, 2020), 1–8.
65. Ding, W., Xu, M. & Zhao, D. *CMTS: A Conditional Multiple Trajectory Synthesizer for Generating Safety-Critical Driving Scenarios* in *In Proceedings of 2020 IEEE ICRA* (IEEE, 2020), 4314–4321.
66. Sun, X., Zhang, Y. & Zhou, W. *Building Narrative Scenarios for Human-Autonomous Vehicle Interaction Research in Simulators* in *Advances in Simulation and Digital Human Modeling - Proceedings of the 2020 Virtual Conferences on Human Factors and Simulation, and Digital Human Modeling and Applied Optimization* **1206** (Springer, 2020), 150–156.
67. Karunakaran, D., Worrall, S. & Nebot, E. Efficient falsification approach for autonomous vehicle validation using a parameter optimisation technique based on reinforcement learning. *arXiv preprint arXiv:2011.07699* (2020).
68. Karunakaran, D., Worrall, S. & Nebot, E. M. *Efficient statistical validation with edge cases to evaluate Highly Automated Vehicles* in *In Proceed-*

ings of the 23rd IEEE International Conference on Intelligent Transportation Systems (IEEE, 2020), 1–8.

69. Ding, W., Chen, B., Xu, M. & Zhao, D. *Learning to Collide: An Adaptive Safety-Critical Scenarios Generating Method* in *In Proceedings of 2020 IEEE/RSJ IROS* (IEEE, 2020), 2243–2250.
70. Wang, J. *et al.* AdvSim: Generating Safety-Critical Scenarios for Self-Driving Vehicles. *In Proceedings of the 2021 IEEE/CVF Conference on CVPR,* 9909–9918 (2021).
71. Ding, W. *et al.* A Survey on Safety-Critical Driving Scenario Generation - A Methodological Perspective. *arXiv preprint arXiv:2202.02215* (2022).
72. Zhong, Z. *et al.* A Survey on Scenario-Based Testing for Automated Driving Systems in High-Fidelity Simulation. *arXiv preprint arXiv:2112.00964* (2021).
73. Hamdi, A., Mueller, M. & Ghanem, B. *SADA: Semantic Adversarial Diagnostic Attacks for Autonomous Applications* in *In Proceedings of Thirty-Fourth Conference on Artificial Intelligence* (AAAI Press, 2020), 10901–10908.
74. Gupta, P., Coleman, D. & Siegel, J. Towards Safer Self-Driving Through Great PAIN (Physically Adversarial Intelligent Networks). *arXiv preprint arXiv:2003.10662* (2020).
75. Wen, M., Park, J. & Cho, K. A scenario generation pipeline for autonomous vehicle simulators. *Human-centric Computing and Information Sciences* **10,** 24 (Dec. 2020).
76. Koren, M. & Kochenderfer, M. J. *Efficient Autonomy Validation in Simulation with Adaptive Stress Testing* in *In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference* (IEEE, 2019), 4178–4183.
77. Muktadir, G. M. & Whitehead, J. *Adversarial jaywalker modeling for simulation-based testing of Autonomous Vehicle Systems* in *In Proceedings of the 2022 IEEE IV* (IEEE, 2022), 1697–1702.
78. Wei, S.-E., Ramakrishna, V., Kanade, T. & Sheikh, Y. *Convolutional pose machines* in *Proceedings of the IEEE Conference on CVPR* (2016), 4724–4732.
79. Lin, J. & Lee, G. H. *Trajectory Space Factorization for Deep Video-Based 3D Human Pose Estimation* in *30th BMVC 2019, BMVC 2019, Cardiff, UK, September 9-12, 2019* (BMVA Press, 2019), 101.
80. Zanfir, A., Marinoiu, E. & Sminchisescu, C. *Monocular 3D Pose and Shape Estimation of Multiple People in Natural Scenes - The Importance of Multiple Scene Constraints* in *Proceedings of the IEEE Conference on CVPR* (June 2018).
81. Tripathi, S. *et al. 3D Human Pose Estimation via Intuitive Physics* in *Proceedings of the IEEE/CVF Conference on CVPR* (June 2023), 4713–4725.
82. Cao, Z., Simon, T., Wei, S.-E. & Sheikh, Y. *Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields* in *CVPR* (2017).

83. Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S. & Sheikh, Y. A. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).
84. Zhang, J., Tu, Z., Yang, J., Chen, Y. & Yuan, J. *Mixste: Seq2seq mixed spatio-temporal encoder for 3d human pose estimation in video* in *Proceedings of the IEEE/CVF Conference on CVPR* (2022), 13232–13242.
85. Cheng, Y., Yang, B., Wang, B. & Tan, R. T. *3d human pose estimation using spatio-temporal networks with explicit occlusion training* in *Proceedings of the AAAI Conference on Artificial Intelligence* **34** (2020), 10631–10638.
86. Shan, W. *et al. P-stmo: Pre-trained spatial temporal many-to-one model for 3d human pose estimation* in *ECCV* (2022), 461–478.
87. Chen, T. *et al.* Anatomy-aware 3d human pose estimation with bone-based pose decomposition. *IEEE Transactions on Circuits and Systems for Video Technology* **32,** 198–209 (2021).
88. Zheng, C. *et al. 3d human pose estimation with spatial and temporal transformers* in *Proceedings of the IEEE/CVF ICCV* (2021), 11656–11665.
89. Liu, R. *et al. Attention Mechanism Exploits Temporal Contexts: Real-Time 3D Human Pose Reconstruction* in *2020 IEEE/CVF Conference on CVPR, CVPR 2020, Seattle, WA, USA, June 13-19, 2020* (Computer Vision Foundation / IEEE, 2020), 5063–5072.
90. Ionescu, C., Papava, D., Olaru, V. & Sminchisescu, C. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **36,** 1325–1339 (2013).
91. Sigal, L., Balan, A. O. & Black, M. J. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision* **87,** 4–27 (2010).
92. Mehta, D. *et al. Monocular 3D Human Pose Estimation In The Wild Using Improved CNN Supervision* in *2017 Fifth International Conference on 3D Vision (3DV)* (2017).
93. Joo, H. *et al. Panoptic Studio: A Massively Multiview System for Social Motion Capture* in *2015 IEEE ICCV, ICCV 2015, Santiago, Chile, December 7-13, 2015* (IEEE Computer Society, 2015), 3334–3342.
94. Zanfir, M. *et al. THUNDR: Transformer-based 3D HUmaN Reconstruction with Markers* in *2021 IEEE/CVF ICCV, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021* (IEEE, 2021), 12951–12960.
95. Huang, C. P. *et al. Capturing and Inferring Dense Full-Body Human-Scene Contact* in *IEEE/CVF Conference on CVPR, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022* (IEEE, 2022), 13264–13275.
96. Wandt, B., Little, J. J. & Rhodin, H. *Elepose: Unsupervised 3d human pose estimation by predicting camera elevation and learning normalizing flows on 2d poses* in *Proceedings of the IEEE/CVF Conference on CVPR* (2022), 6635–6645.

97. Hu, X. & Ahuja, N. *Unsupervised 3d pose estimation for hierarchical dance video recognition* in *Proceedings of the IEEE/CVF ICCV* (2021), 11015–11024.
98. Huang, B., Shu, Y., Ju, J. & Wang, Y. Occluded Human Body Capture with Self-Supervised Spatial-Temporal Motion Prior. *arXiv preprint arXiv:2207.05375* (2022).
99. Xu, C., Chen, S., Li, M. & Zhang, Y. *Invariant Teacher and Equivariant Student for Unsupervised 3D Human Pose Estimation* in *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021* (AAAI Press, 2021), 3013–3021.
100. Kundu, J. N. *et al. Non-local Latent Relation Distillation for Self-Adaptive 3D Human Pose Estimation* in *NeurIPS 2021, December 6-14, 2021, virtual* (2021), 158–171.
101. Deng, Y., Sun, C., Zhu, J. & Sun, Y. *SVMAC: Unsupervised 3D Human Pose Estimation from a Single Image with Single-view-multi-angle Consistency* in *International Conference on 3D Vision, 3DV 2021, London, United Kingdom,December 1-3, 2021* (IEEE, 2021), 474–483.
102. Kundu, J. N. *et al. Self-Supervised 3D Human Pose Estimation via Part Guided Novel Image Synthesis* in *2020 IEEE/CVF Conference on CVPR, CVPR 2020, Seattle, WA, USA, June 13-19, 2020* (Computer Vision Foundation / IEEE, 2020), 6151–6161.
103. Kundu, J. N. *et al. Uncertainty-Aware Adaptation for Self-Supervised 3D Human Pose Estimation* in *IEEE/CVF Conference on CVPR, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022* (IEEE, 2022), 20416–20427.
104. Zanfir, A. *et al. Neural Descent for Visual 3D Human Pose and Shape* in *Proceedings of the IEEE/CVF Conference on CVPR* (June 2021), 14484–14493.
105. Zhu, W. *et al.* Motionbert: Unified pretraining for human motion analysis. *arXiv preprint arXiv:2210.06551* (2022).
106. Gong, K., Zhang, J. & Feng, J. *PoseAug: A Differentiable Pose Augmentation Framework for 3D Human Pose Estimation* in *IEEE Conference on CVPR, CVPR 2021, virtual, June 19-25, 2021* (Computer Vision Foundation / IEEE, 2021), 8575–8584.
107. Roy, S. K., Citraro, L., Honari, S. & Fua, P. *On Triangulation as a Form of Self-Supervision for 3D Human Pose Estimation* in *International Conference on 3D Vision, 3DV 2022, Prague, Czech Republic, September 12-16, 2022* (IEEE, 2022), 1–10.
108. Shan, W. *et al.* Diffusion-Based 3D Human Pose Estimation with Multi-Hypothesis Aggregation. *arXiv preprint arXiv:2303.11579* (2023).
109. Zhou, K., Han, X., Jiang, N., Jia, K. & Lu, J. *Hemlets pose: Learning part-centric heatmap triplets for accurate 3d human pose estimation* in *Proceedings of the IEEE/CVF ICCV* (2019), 2344–2353.
110. Li, Z., Liu, J., Zhang, Z., Xu, S. & Yan, Y. *CLIFF: Carrying Location Information in Full Frames into Human Pose and Shape Estimation* in

Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part V **13665** (Springer, 2022), 590–606.

111. Popa, A.-I., Zanfir, M. & Sminchisescu, C. *Deep Multitask Architecture for Integrated 2D and 3D Human Sensing* in *Proceedings of the IEEE Conference on CVPR* (July 2017).
112. Fieraru, M. *et al. REMIPS: Physically Consistent 3D Reconstruction of Multiple Interacting People under Weak Supervision* in *NeuRIPS* **34** (Curran Associates, Inc., 2021), 19385–19397.
113. Zanfir, A. *et al. Weakly Supervised 3D Human Pose and Shape Reconstruction with Normalizing Flows* in *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VI* **12351** (Springer, 2020), 465–481.
114. Gholami, M., Wandt, B., Rhodin, H., Ward, R. & Wang, Z. J. *AdaptPose: Cross-Dataset Adaptation for 3D Human Pose Estimation by Learnable Motion Generation* in *IEEE/CVF Conference on CVPR, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022* (IEEE, 2022), 13065–13075.
115. Usman, B., Tagliasacchi, A., Saenko, K. & Sud, A. *MetaPose: Fast 3D Pose from Multiple Views without 3D Supervision* in *IEEE/CVF Conference on CVPR, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022* (IEEE, 2022), 6749–6760.
116. Von Marcard, T., Henschel, R., Black, M. J., Rosenhahn, B. & Pons-Moll, G. *Recovering accurate 3d human pose in the wild using imus and a moving camera* in *Proceedings of the ECCV* (2018), 601–617.
117. Ming, Y., Meng, X., Fan, C. & Yu, H. Deep learning for monocular depth estimation: A review. *Neurocomputing* **438,** 14–33 (2021).
118. Roussel, T., Van Eycken, L. & Tuytelaars, T. *Monocular depth estimation in new environments with absolute scale* in *2019 IEEE/RSJ IROS* (2019), 1735–1741.
119. Priisalu, M., Pirinen, A., Paduraru, C. & Sminchisescu, C. *Generating Scenarios with Diverse Pedestrian Behaviors for Autonomous Vehicle Testing* in *PMLR: Proceedings of CoRL 2021* (Nov. 2021).
120. Hug, R., Becker, S., Hübner, W. & Arens, M. Quantifying the Complexity of Standard Benchmarking Datasets for Long-Term Human Trajectory Prediction. *IEEE Access* **9,** 77693–77704 (2021).
121. Saadatnejad, S. *et al.* Are socially-aware trajectory prediction models really socially-aware? *Transportation Research Part C: Emerging Technologies* **141,** 103705 (2022).
122. Schöller, C., Aravantinos, V., Lay, F. & Knoll, A. C. What the Constant Velocity Model Can Teach Us About Pedestrian Motion Prediction. *IEEE Robotics and Automation Letters* **5,** 1696–1703 (2020).
123. Guo, J., Kurup, U. & Shah, M. Is it Safe to Drive? An Overview of Factors, Challenges, and Datasets for Driveability Assessment in Autonomous Driving. *arXiv preprint arXiv:1811.11277* (2018).

124. Li, H. *et al.* Delving into the Devils of Bird's-eye-view Perception: A Review, Evaluation and Recipe. *arXiv preprint arXiv:2209.05324* (2022).

125. Singh, A. & Bankiti, V. Surround-View Vision-based 3D Detection for Autonomous Driving: A Survey. *arXiv preprint arXiv:2302.06650* (2023).

126. Casas, S. *et al. Implicit Latent Variable Model for Scene-Consistent Motion Forecasting* in *ECCV* (2020).

127. Xiong, Y., Ma, W.-C., Wang, J. & Urtasun, R. *Learning Compact Representations for LiDAR Completion and Generation* in *Proceedings of the IEEE/CVF Conference on CVPR* (2023).

128. Ma, W.-C. *et al.* Exploiting Sparse Semantic HD Maps for Self-Driving Vehicle Localization. *2019 IEEE/RSJ IROS,* 5304–5311 (2019).

129. Belkada, Y., Bertoni, L., Caristan, R., Mordan, T. & Alahi, A. Do Pedestrians Pay Attention? Eye Contact Detection in the Wild. *arXiv preprint arXiv:2112.04212* (2021).

130. Adeli, V., Adeli, E., Reid, I., Niebles, J. C. & Rezatofighi, H. Socially and contextually aware human motion and pose forecasting. *IEEE Robotics and Automation Letters* **5,** 6033–6040 (2020).

131. Rempe, D. *et al. Contact and human dynamics from monocular video* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16* (2020), 71–87.

132. Mı́nguez, R. Q., Alonso, I. P., Fernández-Llorca, D. & Sotelo, M. A. Pedestrian path, pose, and intention prediction through gaussian process dynamical models and pedestrian activity recognition. *IEEE Transactions on Intelligent Transportation Systems* **20,** 1803–1814 (2018).

133. Cao, Z. *et al. Long-term human motion prediction with scene context* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16* (2020), 387–404.

134. Rasouli, A., Rohani, M. & Luo, J. *Bifold and semantic reasoning for pedestrian behavior prediction* in *Proceedings of the IEEE/CVF ICCV* (2021), 15600–15610.

135. Agrawal, P. & Brahma, P. P. Single shot multitask pedestrian detection and behavior prediction. *arXiv preprint arXiv:2101.02232* (2021).

136. Huynh, M. & Alaghband, G. AOL: Adaptive Online Learning for Human Trajectory Prediction in Dynamic Video Scenes, 262 (2020).

137. Piccoli, F. *et al. Fussi-net: Fusion of spatio-temporal skeletons for intention prediction network* in *2020 54th Asilomar Conference on Signals, Systems, and Computers* (2020), 68–72.

138. Ranga, A. *et al.* VRUNet: Multi-Task Learning Model for Intent Prediction of Vulnerable Road Users. *Electronic Imaging* **32,** 1–10 (2020).

139. Lorenzo, J. *et al. Rnn-based pedestrian crossing prediction using activity and pose-related features* in *2020 IEEE IV* (2020), 1801–1806.

140. Mangalam, K., Adeli, E., Lee, K.-H., Gaidon, A. & Niebles, J. C. *Disentangling human dynamics for pedestrian locomotion forecasting with noisy supervision* in *Proceedings of the IEEE/CVF WACV* (2020), 2784–2793.

141. Kim, U.-H., Ka, D., Yeo, H. & Kim, J.-H. A real-time vision framework for pedestrian behavior recognition and intention prediction at intersections using 3d pose estimation. *arXiv preprint arXiv:2009.10868* (2020).
142. Zhao, J., Li, Y., Xu, H. & Liu, H. Probabilistic prediction of pedestrian crossing intention using roadside LiDAR data. *IEEE Access* **7,** 93781–93790 (2019).
143. Zhang, Z. *et al. Stinet: Spatio-temporal-interactive network for pedestrian detection and trajectory prediction* in *Proceedings of the IEEE/CVF Conference on CVPR* (2020), 11346–11355.
144. Harley, A. W., Lakshmikanth, S. K., Schydlo, P. & Fragkiadaki, K. *Tracking emerges by looking around static scenes, with neural 3d mapping* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16* (2020), 598–614.
145. Zanfir, A. *et al. HUM3DIL: Semi-supervised Multi-modal 3D Human Pose Estimation for Autonomous Driving* in *CoRL* (2022).
146. Shah, M. *et al. LiRaNet: End-to-End Trajectory Prediction using Spatio-Temporal Radar Fusion* in *CoRL* (2020).
147. Agro, B., Sykora, Q., Casas, S. & Urtasun, R. *Implicit Occupancy Flow Fields for Perception and Prediction in Self-Driving* in *Proceedings of the IEEE/CVF Conference on CVPR* (2023).
148. Li, L. L. *et al.* End-to-end Contextual Perception and Prediction with Interaction Transformer. *2020 IEEE/RSJ IROS,* 5784–5791 (2020).
149. Sadat, A. *et al. Perceive, Predict, and Plan: Safe Motion Planning Through Interpretable Semantic Representations* in *ECCV* (2020).
150. Liang, M. *et al.* PnPNet: End-to-End Perception and Prediction With Tracking in the Loop. *2020 IEEE/CVF Conference on CVPR,* 11550–11559 (2020).
151. Sadat, A. *et al.* Jointly Learnable Behavior and Trajectory Planning for Self-Driving Vehicles. *2019 IEEE/RSJ IROS,* 3949–3956 (2019).
152. Xu, L., Xu, R. & Jin, S. *HiEve ACM MM Grand Challenge 2020: Pose Tracking in Crowded Scenes* in *Proceedings of the 28th ACM International Conference on Multimedia* (Association for Computing Machinery, Seattle, WA, USA, 2020), 4689–4693.
153. Chang, S. *et al. Towards Accurate Human Pose Estimation in Videos of Crowded Scenes* in *Proceedings of the 28th ACM International Conference on Multimedia* (Association for Computing Machinery, Seattle, WA, USA, 2020), 4630–4634.
154. Yuan, L. *et al. A Simple Baseline for Pose Tracking in Videos of Crowed Scenes* in *Proceedings of the 28th ACM International Conference on Multimedia* (Association for Computing Machinery, Seattle, WA, USA, 2020), 4684–4688.
155. Saini, N., Huang, C.-H. P., Black, M. J. & Ahmad, A. SmartMocap: Joint Estimation of Human and Camera Motion Using Uncalibrated RGB Cameras. *IEEE Robotics and Automation Letters* (2023).

156. Zhang, L. *et al. Towards Unsupervised Object Detection From LiDAR Point Clouds* in *Proceedings of the IEEE/CVF Conference on CVPR* (June 2023), 9317–9328.
157. Windbacher, F., Hödlmoser, M. & Gelautz, M. *Single-Stage 3D Pose Estimation of Vulnerable Road Users Using Pseudo-Labels* in *Image Analysis* (Springer Nature Switzerland, Cham, 2023), 401–417.
158. Cordts, M. *et al. The Cityscapes Dataset for Semantic Urban Scene Understanding* in *Proc. of the IEEE Conference on CVPR* (2016).
159. Morelli, L. *et al.* COLMAP-SLAM: A FRAMEWORK FOR VISUAL ODOMETRY. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **48,** 317–324 (2023).
160. Mapillary. *Mapillary/opensfm: Open source structure-from-motion pipeline* `https://github.com/mapillary/OpenSfM`.
161. Snavely, N., Seitz, S. M. & Szeliski, R. in *ACM siggraph 2006 papers* 835–846 (2006).
162. OnpenCV. *OpenCV 3.1.0 Structure from Motion Library* `http://docs.opencv.org/3.1.0/de/d7c/tutorial_table_of_content_sfm.html`.
163. Tola, E., Lepetit, V. & Fua, P. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32,** 815–830 (2009).
164. Wu, C. *Towards linear-time incremental structure from motion* in *2013 International Conference on 3D Vision-3DV 2013* (2013), 127–134.
165. Mur-Artal, R. & Tardós, J. D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics* **33,** 1255–1262 (2017).
166. Rublee, E., Rabaud, V., Konolige, K. & Bradski, G. *ORB: An efficient alternative to SIFT or SURF* in *2011 ICCV* (2011), 2564–2571.
167. Schönberger, J. L. & Frahm, J.-M. *Structure-from-Motion Revisited* in *Conference on CVPR* (2016).
168. Schönberger, J. L., Zheng, E., Pollefeys, M. & Frahm, J.-M. *Pixelwise View Selection for Unstructured Multi-View Stereo* in *ECCV* (2016).
169. Lowe, G. Sift-the scale invariant feature transform. *Int. J* **2,** 2 (2004).
170. Nilsson, D. & Sminchisescu, C. *Semantic video segmentation by gated recurrent flow propagation* in *Proceedings of the IEEE Conference on CVPR* (2018), 6819–6828.
171. Priisalu, M., Paduraru, C., Pirinen, A. & Sminchisescu, C. *Semantic Synthesis of Pedestrian Locomotion* in *In Proceedings of the 2020 ACCV* **12623** (2020), 470–487.
172. Qi, C. R., Yi, L., Su, H. & Guibas, L. J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeuRIPS* **30** (2017).
173. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A. & Koltun, V. *CARLA: An Open Urban Driving Simulator* in *In Proceedings of the 2017 CoRL* **78** (PMLR, 2017), 1–16.
174. Schult, J. *et al. Mask3D: Mask Transformer for 3D Semantic Instance Segmentation* in *2023 IEEE ICRA* (2023), 8216–8223.

175. Ngo, T. D., Hua, B.-S. & Nguyen, K. *ISBNet: a 3D Point Cloud Instance Segmentation Network with Instance-aware Sampling and Box-aware Dynamic Convolution* in *Proceedings of the IEEE/CVF Conference on CVPR* (2023), 13550–13559.

176. Sun, J., Qing, C., Tan, J. & Xu, X. *Superpoint transformer for 3d scene instance segmentation* in *Proceedings of the AAAI Conference on Artificial Intelligence* **37** (2023), 2393–2401.

177. Chen, R. *et al. CLIP2Scene: Towards Label-efficient 3D Scene Understanding by CLIP* in *Proceedings of the IEEE/CVF Conference on CVPR* (2023), 7020–7030.

178. Cheng, R., Razani, R., Taghavi, E., Li, E. & Liu, B. *2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network* in *Proceedings of the IEEE/CVF Conference on CVPR* (2021), 12547–12556.

179. Hou, Y., Zhu, X., Ma, Y., Loy, C. C. & Li, Y. *Point-to-voxel knowledge distillation for lidar semantic segmentation* in *Proceedings of the IEEE/CVF Conference on CVPR* (2022), 8479–8488.

180. Xu, J. *et al. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation* in *Proceedings of the IEEE/CVF ICCV* (2021), 16024–16033.

181. Zhu, X. *et al. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation* in *Proceedings of the IEEE/CVF Conference on CVPR* (2021), 9939–9948.

182. Yu, F. & Koltun, V. *Multi-scale context aggregation by dilated convolutions* in *ICLR* (2016).

183. Ren, S., He, K., Girshick, R. & Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeuRIPS* **28** (2015).

184. Popa, A.-I., Zanfir, M. & Sminchisescu, C. *Deep multitask architecture for integrated 2d and 3d human sensing* in *proceedings of the IEEE Conference on CVPR* (2017), 6289–6298.

185. Liu, S., Qi, L., Qin, H., Shi, J. & Jia, J. *Path aggregation network for instance segmentation* in *Proceedings of the IEEE Conference on CVPR* (2018), 8759–8768.

186. Kong, Y. *et al. Research on small object detection methods based on deep learning* in *2022 IEEE 4th International Conference on Power, Intelligent Computing and Systems (ICPICS)* (2022), 680–686.

187. Feng, Q., Xu, X. & Wang, Z. Deep learning-based small object detection: A survey. *Mathematical Biosciences and Engineering* **20,** 6551–6590 (2023).

188. Huan, R., Zhang, J., Xie, C., Liang, R. & Chen, P. MLFFCSP: a new anti-occlusion pedestrian detection network with multi-level feature fusion for small targets. *Multimedia Tools and Applications,* 1–26 (2023).

189. Cheng, Q. & Zhang, S. *Efficiently handling scale variation for pedestrian detection* in *Intelligence Science and Big Data Engineering. Visual Data Engineering: 9th International Conference, IScIDE 2019, Nanjing, China, October 17–20, 2019, Proceedings, Part I 9* (2019), 178–190.

190. Liu, Z. *et al. Improving small-scale pedestrian detection using informed context* in *2019 IEEE Visual Communications and Image Processing (VCIP)* (2019), 1–4.
191. Cao, J., Pang, Y., Han, J., Gao, B. & Li, X. Taking a look at small-scale pedestrians and occluded pedestrians. *IEEE transactions on image processing* **29,** 3143–3152 (2019).
192. Xu, Z., Li, B., Yuan, Y. & Dang, A. Beta r-cnn: Looking into pedestrian detection from another perspective. *NeuRIPS* **33,** 19953–19963 (2020).
193. Liu, T. *et al.* Coupled network for robust pedestrian detection with gated multi-layer feature extraction and deformable occlusion handling. *IEEE transactions on image processing* **30,** 754–766 (2020).
194. Hagn, K. & Grau, O. *Validation of Pedestrian Detectors by Classification of Visual Detection Impairing Factors* in *ECCV* (2022), 476–491.
195. Zou, F., Li, X., Xu, Q., Sun, Z. & Zhu, J. Correlation-and-Correction Fusion Attention Network for Occluded Pedestrian Detection. *IEEE Sensors Journal* **23,** 6061–6073 (2023).
196. Hasan, I., Liao, S., Li, J., Akram, S. U. & Shao, L. *Generalizable Pedestrian Detection: The Elephant in the Room* in *Proceedings of the IEEE/CVF Conference on CVPR* (June 2021), 11328–11337.
197. Madan, M., Reich, C. & Hassenpflug, F. *Drawing and Analysis of Bounding Boxes for Object Detection with Anchor-Based Models* in *Image Analysis* (Springer Nature Switzerland, Cham, 2023), 359–373.
198. Wang, X. *et al. Are We Ready for Vision-Centric Driving Streaming Perception? The ASAP Benchmark* in *Proceedings of the IEEE/CVF Conference on CVPR* (2023), 9600–9610.
199. Chou, F.-C. *et al. Predicting motion of vulnerable road users using high-definition maps and efficient convnets* in *2020 IEEE IV* (2020), 1655–1662.
200. Priisalu, M., Paduraru, C. & Smichisescu, C. *Varied Realistic Autonomous Vehicle Collision Scenario Generation* in *Image Analysis* (Springer Nature Switzerland, Cham, 2023), 354–372.
201. Dalbah, Y., Lahoud, J. & Cholakkal, H. *RadarFormer: Lightweight and Accurate Real-Time Radar Object Detection Model* in *Image Analysis* (Springer Nature Switzerland, Cham, 2023), 341–358.
202. Chen, Y., Liu, J., Zhang, X., Qi, X. & Jia, J. *Voxelnext: Fully sparse voxelnet for 3d object detection and tracking* in *Proceedings of the IEEE/CVF Conference on CVPR* (2023), 21674–21683.
203. Li, Y. *et al.* Deep learning for lidar point clouds in autonomous driving: A review. *IEEE Transactions on Neural Networks and Learning Systems* **32,** 3412–3432 (2020).
204. Li, J., Dai, H., Han, H. & Ding, Y. *MSeg3D: Multi-modal 3D Semantic Segmentation for Autonomous Driving* in *CVPR* (2023), 21694–21704.
205. Li, J., Dai, H. & Ding, Y. *Self-Distillation for Robust LiDAR Semantic Segmentation in Autonomous Driving* in *ECCV* (2022), 659–676.

206. Sun, P. *et al.* RSN: Range Sparse Net for Efficient, Accurate LiDAR 3D Object Detection. *2021 IEEE/CVF Conference on CVPR,* 5721–5730 (2021).
207. Colonnier, F., Seeralan, A. & Zhu, L. *Event-based visual sensing for human motion detection and classification at various distances* in *Pacific-Rim Symposium on Image and Video Technology* (2022), 75–88.
208. Torres, G. F. & Kämäräinen, J. *Depth-Aware Image Compositing Model for Parallax Camera Motion Blur* in *Image Analysis* (Springer Nature Switzerland, Cham, 2023), 279–296.
209. Aakerberg, A., Nasrollahi, K. & Moeslund, T. B. *RELIEF: Joint Low-Light Image Enhancement and Super-Resolution with Transformers* in *Image Analysis* (Springer Nature Switzerland, Cham, 2023), 157–173.
210. Ljungbergh, W., Johnander, J., Petersson, C. & Felsberg, M. *Raw or Cooked? Object Detection on RAW Images* in *Image Analysis* (Springer Nature Switzerland, Cham, 2023), 374–385.
211. Lugmayr, A. *et al. Repaint: Inpainting using denoising diffusion probabilistic models* in *Proceedings of the IEEE/CVF Conference on CVPR* (2022), 11461–11471.
212. Yang, Z. *et al. S3: Neural shape, skeleton, and skinning fields for 3d human modeling* in *Proceedings of the IEEE/CVF Conference on CVPR* (2021), 13284–13293.
213. Fadadu, S. *et al. Multi-view fusion of sensor data for improved perception and prediction in autonomous driving* in *Proceedings of the IEEE/CVF WACV* (2022), 2349–2357.
214. Cui, Y. *et al.* Deep learning for image and point cloud fusion in autonomous driving: A review. *IEEE Transactions on Intelligent Transportation Systems* **23,** 722–739 (2021).
215. Daniel, J. A. *et al.* Fully convolutional neural networks for LIDAR–camera fusion for pedestrian detection in autonomous vehicle. *Multimed Tools Appl* **82,** 25107–25130 (2023).
216. Xu, H. *et al. GHUM & GHUML: Generative 3D Human Shape and Articulated Pose Models* in *IEEE/CVF Conference on CVPR (Oral)* (2020), 6184–6193.
217. Saini, N., Huang, C.-H. P., Black, M. J. & Ahmad, A. SmartMocap: Joint Estimation of Human and Camera Motion Using Uncalibrated RGB Cameras. *IEEE Robotics and Automation Letters* **8,** 3206–3213 (2022).
218. Kocabas, M. *et al. SPEC: Seeing People in the Wild with an Estimated Camera* in *Proc. ICCV (ICCV)* (Oct. 2021), 11035–11045.
219. Gärtner, E., Andriluka, M., Xu, H. & Sminchisescu, C. *Trajectory Optimization for Physics-Based Reconstruction of 3D Human Pose From Monocular Video* in *Proceedings of the IEEE/CVF Conference on CVPR* (June 2022), 13106–13115.
220. Gärtner, E., Andriluka, M., Coumans, E. & Sminchisescu, C. *Differentiable Dynamics for Articulated 3D Human Motion Reconstruction* in *Pro-*

ceedings of the IEEE/CVF Conference on CVPR (June 2022), 13190–13200.

221. Tripathi, S. *et al.* 3D Human Pose Estimation via Intuitive Physics. *arXiv preprint arXiv:2303.18246* (2023).
222. Xu, L., Xu, R. & Jin, S. *HiEve ACM MM Grand Challenge 2020: Pose Tracking in Crowded Scenes* in *Proceedings of the 28th ACM International Conference on Multimedia* (Association for Computing Machinery, Seattle, WA, USA, 2020), 4689–4693.
223. Chang, S. *et al. Towards Accurate Human Pose Estimation in Videos of Crowded Scenes* in *Proceedings of the 28th ACM International Conference on Multimedia* (Association for Computing Machinery, Seattle, WA, USA, 2020), 4630–4634.
224. Yuan, L. *et al. A Simple Baseline for Pose Tracking in Videos of Crowed Scenes* in *Proceedings of the 28th ACM International Conference on Multimedia* (Association for Computing Machinery, Seattle, WA, USA, 2020), 4684–4688.
225. Deng, W. *et al. Split to Learn: Gradient Split for Multi-Task Human Image Analysis* in *2023 IEEE/CVF WACV* (2023), 4340–4349.
226. Das, A. *et al.* Deep Multi-Task Networks For Occluded Pedestrian Pose Estimation. *24th Irish Machine Vision and Image Processing Conference* (2023).
227. Kishore, P. S. R., Das, S., Mukherjee, P. S. & Bhattacharya, U. *ClueNet : A Deep Framework for Occluded Pedestrian Pose Estimation* in *BMVC* (2019).
228. Raj S, S., Prasad, M. V. & Balakrishnan, R. Generative Segment-pose Representation based Augmentation (GSRA) for unsupervised person re-identification. *Image and Vision Computing* **131,** 104632 (2023).
229. You, S., Yao, H. & Xu, C. Multi-Target Multi-Camera Tracking With Optical-Based Pose Association. *IEEE Transactions on Circuits and Systems for Video Technology* **31,** 3105–3117 (2021).
230. Meng, C., He, X., Tan, Z. & Luan, L. Gait recognition based on 3D human body reconstruction and multi-granular feature fusion. *J. Supercomput.* **79,** 12106–12125 (2023).
231. Saini, N., Huang, C.-H. P., Black, M. J. & Ahmad, A. SmartMocap: Joint Estimation of Human and Camera Motion Using Uncalibrated RGB Cameras. *IEEE Robotics and Automation Letters* **8,** 3206–3213 (2023).
232. Ye, V., Pavlakos, G., Malik, J. & Kanazawa, A. *Decoupling Human and Camera Motion from Videos in the Wild* in *IEEE Conference on CVPR* (June 2023).
233. Hara, K. *et al. Predicting Vehicles Appearing from Blind Spots Based on Pedestrian Behaviors* in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)* (2020), 1–8.
234. Shi, J., Yang, H. & Carlone, L. *Optimal Pose and Shape Estimation for Category-level 3D Object Perception* in *Robotics: Science and Systems 2021* **XVIII** (Columbia University, New York City, NY, USA, 2021).

235. Goforth, H., Hu, X., Happold, M. & Lucey, S. Joint pose and shape estimation of vehicles from lidar data. *arXiv preprint arXiv:2009.03964* (2020).
236. Hayakawa, J. & Dariush, B. *Recognition and 3d localization of pedestrian actions from monocular video* in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)* (2020), 1–7.
237. You, Y. *et al. Pseudo-LiDAR++: Accurate Depth for 3D Object Detection in Autonomous Driving* in *ICLR* (2019).
238. Wang, Y. *et al. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving* in *Proceedings of the IEEE/CVF Conference on CVPR* (2019), 8445–8453.
239. Wei, X., Zhang, Y., Li, Z., Fu, Y. & Xue, X. *Deepsfm: Structure from motion via deep bundle adjustment* in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16* (2020), 230–247.
240. Xiao, Y., Xue, N., Wu, T. & Xia, G.-S. *Level-S fM: Structure From Motion on Neural Level Set of Implicit Surfaces* in *Proceedings of the IEEE/CVF Conference on CVPR* (2023), 17205–17214.
241. Zhu, Z. *et al. NICE-SLAM: Neural Implicit Scalable Encoding for SLAM* in *Proceedings of the IEEE/CVF Conference on CVPR* (June 2022), 12786–12796.
242. Sarlin, P.-E. *et al. Back to the Feature: Learning Robust Camera Localization From Pixels To Pose* in *Proceedings of the IEEE/CVF Conference on CVPR* (June 2021), 3247–3257.
243. Zhu, Z. *et al.* Nicer-slam: Neural implicit scene encoding for rgb slam. *arXiv preprint arXiv:2302.03594* (2023).
244. Zhao, C., Sun, Q., Zhang, C., Tang, Y. & Qian, F. Monocular depth estimation based on deep learning: An overview. *Science China Technological Sciences* **63,** 1612–1627 (2020).
245. DeTone, D., Malisiewicz, T. & Rabinovich, A. *Superpoint: Self-supervised interest point detection and description* in *Proceedings of the IEEE Conference on CVPR workshops* (2018), 224–236.
246. Sarlin, P.-E., DeTone, D., Malisiewicz, T. & Rabinovich, A. *Superglue: Learning feature matching with graph neural networks* in *Proceedings of the IEEE/CVF Conference on CVPR* (2020), 4938–4947.
247. Mildenhall, B. *et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis* in *ECCV* (2020).
248. Li, Z., Li, L. & Zhu, J. *READ: Large-Scale Neural Scene Rendering for Autonomous Driving* in *AAAI* (2023).
249. Li, C. *et al.* Elevater: A benchmark and toolkit for evaluating language-augmented visual models. *NeuRIPS* **35,** 9287–9301 (2022).
250. Li, S. *et al. OVTrack: Open-Vocabulary Multiple Object Tracking* in *IEEE-/CVF Conference on CVPR, CVPR* (2023).
251. Paul, M., Danelljan, M., Mayer, C. & Gool, L. V. *Robust Visual Tracking by Segmentation* in *ECCVECCV* (2022).

252. Zhou, T., Porikli, F., Crandall, D. J., Van Gool, L. & Wang, W. A survey on deep learning technique for video segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **45,** 7099–7122 (2022).
253. Zou, X. *et al.* Segment everything everywhere all at once. *arXiv preprint arXiv:2304.06718* (2023).
254. Zhang, H. *et al.* A Simple Framework for Open-Vocabulary Segmentation and Detection. *arXiv preprint arXiv:2303.08131* (2023).
255. Li, F. *et al.* Semantic-SAM: Segment and Recognize Anything at Any Granularity. *arXiv preprint arXiv:2307.04767* (2023).
256. Zou, X. *et al. Generalized decoding for pixel, image, and language* in *Proceedings of the IEEE/CVF Conference on CVPR* (2023), 15116–15127.
257. Vaquero, V. *et al. Deconvolutional networks for point-cloud vehicle detection and tracking in driving scenarios* in *2017 European Conference on Mobile Robots (ECMR)* (2017), 1–7.
258. Caesar, H. *et al. nuScenes: A Multimodal Dataset for Autonomous Driving* in *2020 IEEE/CVF Conference on CVPR, CVPR 2020, Seattle, WA, USA, June 13-19, 2020* (Computer Vision Foundation / IEEE, 2020), 11618–11628.
259. Sharma, D., Hade, T. & Tian, Q. Comparison Of Deep Object Detectors On A New Vulnerable Pedestrian Dataset. *arXiv preprint arXiv:2212.06218* (2022).
260. Xu, Z., Zhuang, J., Liu, Q., Zhou, J. & Peng, S. *Nighttime FIR Pedestrian Detection Benchmark Dataset for ADAS* in *Pattern Recognition and Computer Vision - First Chinese Conference, PRCV 2018, Guangzhou, China, November 23-26, 2018, Proceedings, Part IV* **11259** (Springer, 2018), 322–333.
261. Tumas, P., Nowosielski, A. & Serackis, A. Pedestrian Detection in Severe Weather Conditions. *IEEE Access* **PP,** 1–1 (Mar. 2020).
262. Pham, Q. *et al. A*3D Dataset: Towards Autonomous Driving in Challenging Environments* in *2020 IEEE ICRA, ICRA 2020, Paris, France, May 31 - August 31, 2020* (IEEE, 2020), 2267–2273.
263. Cong, P. *et al. STCrowd: A Multimodal Dataset for Pedestrian Perception in Crowded Scenes* in *IEEE/CVF Conference on CVPR, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022* (IEEE, 2022), 19576–19585.
264. Zhang, S. *et al.* WiderPerson: A Diverse Dataset for Dense Pedestrian Detection in the Wild. *IEEE Trans. Multim.* **22,** 380–393 (2020).
265. Breitenstein, J. & Fingscheidt, T. *Amodal Cityscapes: A New Dataset, its Generation, and an Amodal Semantic Segmentation Challenge Baseline* in *2022 IEEE IV, IV 2022, Aachen, Germany, June 4-9, 2022* (IEEE, 2022), 1018–1025.
266. Wang, X. *et al. When Pedestrian Detection Meets Nighttime Surveillance: A New Benchmark* in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020* (ijcai.org, 2020), 509–515.

267. Neumann, L. *et al. NightOwls: A Pedestrians at Night Dataset* in *Computer Vision - ACCV 2018 - 14th Asian Conference on Computer Vision, Perth, Australia, December 2-6, 2018, Revised Selected Papers, Part I* **11361** (Springer, 2018), 691–705.

268. Rasouli, A., Kotseruba, I., Kunic, T. & Tsotsos, J. K. *PIE: A Large-Scale Dataset and Models for Pedestrian Intention Estimation and Trajectory Prediction* in *2019 IEEE/CVF ICCV, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019* (IEEE, 2019), 6261–6270.

269. Ul Huda, N., Hansen, B. D., Gade, R. & Moeslund, T. B. The Effect of a Diverse Dataset for Transfer Learning in Thermal Person Detection. *Sensors* **20,** 1982 (2020).

270. Zhang, S., Benenson, R. & Schiele, B. *CityPersons: A Diverse Dataset for Pedestrian Detection* in *2017 IEEE Conference on CVPR, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017* (IEEE Computer Society, 2017), 4457–4465.

271. Mikolajczyk, K. & Schmid, C. Scale & Affine Invariant Interest Point Detectors. *International Journal of Computer Vision* **60,** 63–86 (Oct. 2004).

272. McConnell, R. K. *Method of and apparatus for pattern recognition* U. S. Patent No. 4,567,610,Jan. 1986.

273. Muja, M. & Lowe, D. G. *Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration* in *ICCV Theory and Applications* (2009).

274. Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R. & Wu, A. Y. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)* **45,** 891–923 (1998).

275. Fischler, M. A. & Bolles, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* **24,** 381–395 (1981).

276. Hartley, R. & Zisserman, A. *Multiple view geometry in computer vision* (Cambridge university press, 2003).

277. Nocedal, J. & Wright, S. J. *Numerical optimization* (Springer, 1999).

278. Lourakis, M. & Argyros, A. *The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm* tech. rep. (Technical Report 340, Institute of Computer Science-FORTH, Heraklion, Crete, 2004).

279. Agarwal, S., Snavely, N., Seitz, S. M. & Szeliski, R. *Bundle adjustment in the large* in *Computer Vision–ECCV 2010: 11th ECCV, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II 11* (2010), 29–42.

280. Wu, C. Siftgpu: A gpu implementation of scale invariant feature transform (sift)(2007). *URL http://cs. unc. edu/~ ccwu/siftgpu* (2011).

281. Wu, C., Agarwal, S., Curless, B. & Seitz, S. M. *Multicore bundle adjustment* in *CVPR 2011* (2011), 3057–3064.

282. Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K. & Burgard, W. *g 2 o: A general framework for graph optimization* in *2011 IEEE ICRA* (2011), 3607–3613.

283. Noah, S. *Scene Reconstruction and Visualization from Internet Photo Collections* PhD thesis (Ph. D. dissertation, Department of Computer Science and Engineering ..., 2008).

284. Beder, C. & Steffen, R. *Determining an initial image pair for fixing the scale of a 3d reconstruction from an image sequence* in *Joint Pattern Recognition Symposium* (2006), 657–666.

285. Davis, T. A. & Hager, W. *CHOLMOD: supernodal sparse cholesky factorization and update/downdate* 2005.

286. Lourakis, M. I. & Argyros, A. A. SBA: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software (TOMS)* **36,** 1–30 (2009).

A 3D reconstruction systems overview

Open Structure for Motion Library (OpenSFM) [160]- A Structure for Motion system, that is an incremental 3d reconstruction system. Uses Hessian Affine Feature Point detector [271] and Histogram of Oriented Gradients (HOG) [272] descriptors jointly which are nearest neighbor matched [273] across images. A rotation-only transformation is found between the first frames if at least 30% of the points are outliers (to ensure a large enough change in viewpoint). From the initial pair, a sparse 3D cloud is found by the 5-point algorithm or by assuming planar motion of the camera, whichever performs best. The resulting camera matrices are then used for triangulation and bundle adjustment (BA). Additional frames are added according to the largest overlap with the existing pointcloud, and are aligned with the pointcloud to minimize the re-projection error of the pointcloud. BA is applied after adding new images to the pointcloud. Fails to reconstruct the Cityscapes scenes likely because the change in camera rotation is too small between frames.

Bundler [161] is also a SFM system. It detects SIFT [169] features that are matched with approximate nearest neighbors [274]. RANSAC [275] is used to find the fundamental matrix with the 8-point algorithm [276]. The fundamental matrix is refined, its outliers are removed and keypoints tracks are checked for consistency. Levenberg-Marquardt [277] is used to find the first camera matrices and in sparse bundle adjustment [278] for any additional cameras (images chosen in the order of largest number of matches with triangulated points) that are initialized with Direct Linear Transform (DLT) [276]. The initial image pair is chosen such that there is a large enough rotational difference between the images. Finds <10 matches, and fails again likely because the images are blurry and the rotational difference between the camera views is too small.

OpenCV Structure from motion library [162] - A SFM library that uses DAISY features [163]. Finds the essential matrix with RANSAC and the 8-point algorithm. And an inexact Newton method Schur-based solvers [279] to optimize BA. Result of 30 frames - finds relatively few points without clear structure. See Fig. 5

VisualSFM [164] a parallelized SFM pipeline with Bundler. Uses SIFT on the GPU [280] and Multicore Bundle adjustment [281]. Only a thresholded number of large-scale features are matched across images. This unfortunately fails possibly because of image blur or the lack of distinct large-scale structures in the images. The method is unable to find enough SIFT feature points likely because the images are blurry and finds no verified matches between two stereo images. Finally, VisualSFM cannot handle forward motion, not finding a good initial pair of images with enough matches.

ORBSLAM [165]- ORB-feature [166] (a fast feature descriptor combining gradient and binary features) based Simultaneous Localization And Mapping system. ORB features from the left image are matched along epipolar lines with ORB features from the right image, and the disparity is calculated. Points that are further than 40 baselines away from the camera are ignored. The points that are close to the camera are triangulated, and the left camera is considered to be the origin. Additional cameras are added by performing camera position optimization in BA between matched 3D points and keypoints in the new frame, BA of the newly added added keypoints, and by finally performing full BA after loop-closure detection and correction. BA is optimized with Levenberg–Marquardt implemented as g20 [282] Finds too few keypoints, likely due to blur and depth threshold. Results in a too sparse reconstruction. Fig. 6.

COLMAP [167, 168]- an incremental Structure for Motion and Muti-view stereo system. Extracts SIFT [169] features that are exhaustively matched (other matching methods are also available) across all images. The reconstruction is built from an initial pair of images, chosen by [283, 284] additional views are chosen by high inlier ratios that approximate uncertainty estimates and by prioritizing images with uniformly distributed keypoints that match with triangulated points. The method uses a robust RANSAC-based triangulation with adaptive outlier thresholds to add new camera views to the reconstruction. Local BA is performed after a new camera view is added. Finally, a global BA is performed iteratively followed by filtering of outliers and degenerate camera view, and triangulation until the BA converges. BA is performed by Preconditioned Conjugate Gradient [279, 281] for a large number of cameras and by [285, 286] for smaller systems. Converges for 150 scenes on the training and validation set and 150 scenes on the test set out of the 3475 scenes available. See

B Additional Results

B.1 Error distribution per human body joint

In Fig. B it can be seen that in general feet are the hardest to estimate the position of, while the head is the easiest. If the articulated human motion is to be predicted we however need the feet position to be accurate to foresee the pedestrian's future velocity. To improve this temporally smooth human detection and pose estimation methods should be utilized in the future. In Fig. B*Right* a comparison between the error distribution of *Dilational* net and *GRFP* is shown.

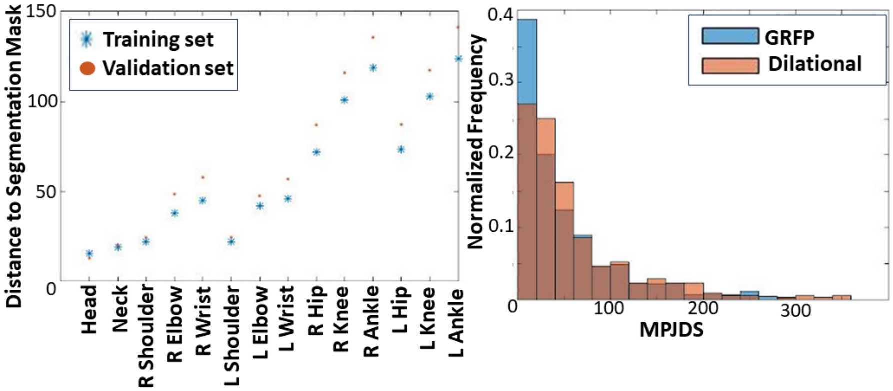


Fig. 20. *Left:* The JDS for different joints. The feet are the most difficult to detect. *Right:* Comparison of the MPJDS rates of the Dilational and GRFP net. GRFP has lower MPJDS than the Dilational net.

It is clear that *GRFP* produces joint estimations that have a lower distance to the human mask.

B.2 Procrustes Analysis

The results of aligning the thresholded pose with its nearest neighbor from Human3.6M is shown in Fig. 21. Because all limb lengths are reconstructed under noise the different limbs get elongated or compressed to a different degree. Therefore scaling the pose according to hip length results in this case in a too large scaling factor because the hip length is compressed in the reconstruction. The backbone is elongated so scaling according to the backbone results in a too small scaling factor. The elongations and compression of the different limbs vary from one triangulated pose to another, making the choice of a scaling factor hard. The height difference between the feet and the heat depends on the pose and is therefore not a suitable measure for scale. The same applies to the distance between the feet and hands.

B.3 Additional Pedestrian Detection and Pose Estimation

OpenPose misses a large number of pedestrians (with variations from frame to frame) as seen in the supplementary video at `https://youtu.be/qpxpdtHbbGA` and Fig. 22. OpenPose misses pedestrians due to large distance to camera, poor lightning and motion blur as seen in the supplementary video. OpenPose can produce impressive results when pedestrians are close to the camera but at a distance it has trouble with occlusions and can produce a number of odd false positive pedestrian detections, as seen in the video. Pedestrian detection is improved by applying FRCNN object detector. But FRCNN still omits distant pedestrians as seen in the video and Fig. 22,Fig. 23, and Fig. 24. On the other

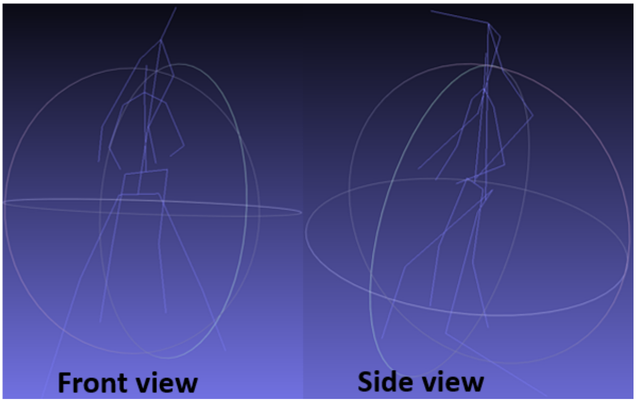


Fig. 21. Skeleton with thresholded limb lengths after scaling according to hip bone length shown with its nearest neighbor from the Human3.6M dataset. The longer skeleton is the thresholded 3D reconstructed human pose. The unthresholded skeleton is shown in Fig. 16 *Left*.

hand the semantic sgementation networks are susceptible for false positive as seen in the video. Finally it can be noted that on close by pedestrians FRCNN produces bboxes that are larger than the pedestrian often leading to better 2D pose estimates than the segmentation network, as seen in Fig. 25.

B.4 Additional DMHS results

The DMHS's accuracy is like OpenPose, depent on the bbox placement. Because the FRCNN produces bboxes that jump from frame to frame as seen in Fig. 26. FRCNN can even jump frames, by being unable to detect pedestrians at some frames. This motivates our suggestion that temporally smoothed methods should be developed for articulated pedestrian detection. DMHS's quality varies from image to image, some samples with quality variations are shown in Fig. 27

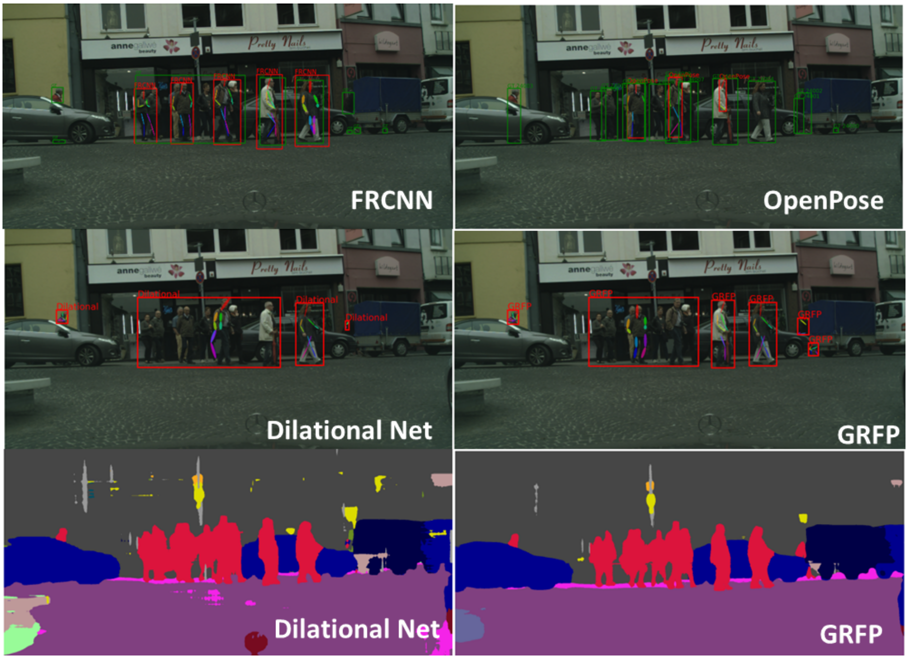


Fig. 22. OpenPose when applied on the whole image detects only a few pedestrians. FRCNN detects some selected pedestrians. The segmentation networks can detect all of the pedestrians, but because they produce only class labels one single BBox is given to multiple pedestrians. GRFP has smoother segmentation than DilationalNet and results in a better separation of the pedestrian Bboxes.

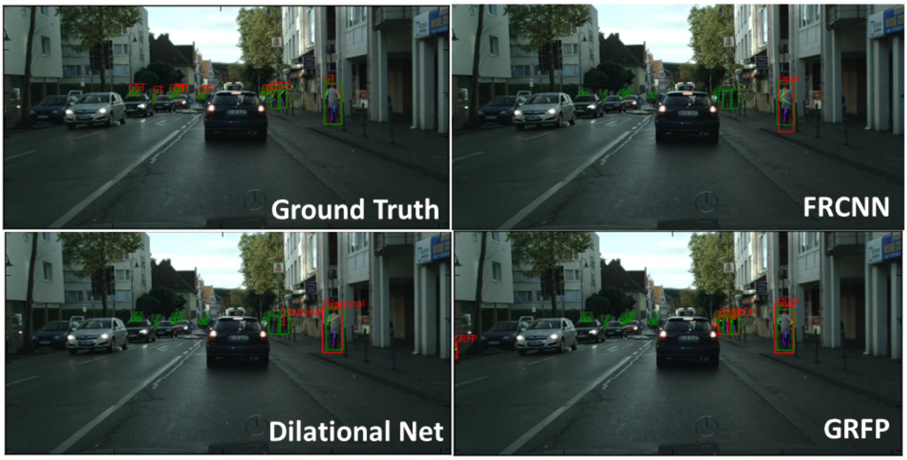


Fig. 23. FRCNN misses a large number of the distant pedestrians. The Dilational net and GRFP detect more distant pedestrians than FRCNN and GRFP results in a more accurate 2D pose for the closest pedestrian to the right.

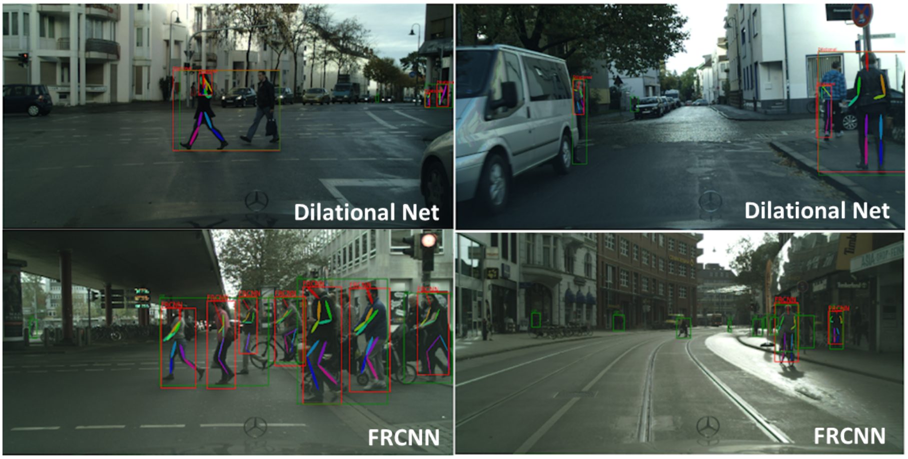


Fig. 24. On the top: Dilational Net can detect pedestrians even in the presence of occlusions, but produces one single bounding box for close by pedestrians. FRCNN can detect pedestrians that are close to the camera well, but fails to detect far away pedestrians.

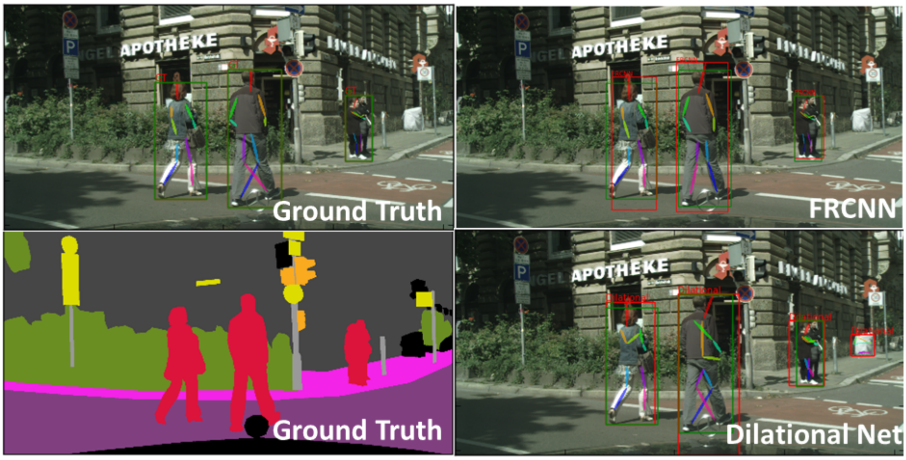


Fig. 25. The placement of the bounding box affects the estimated 2D body pose. FR-CNN produces larger bounding boxes than found by the GT segmentation mask. This produces a more correct 2D body pose.

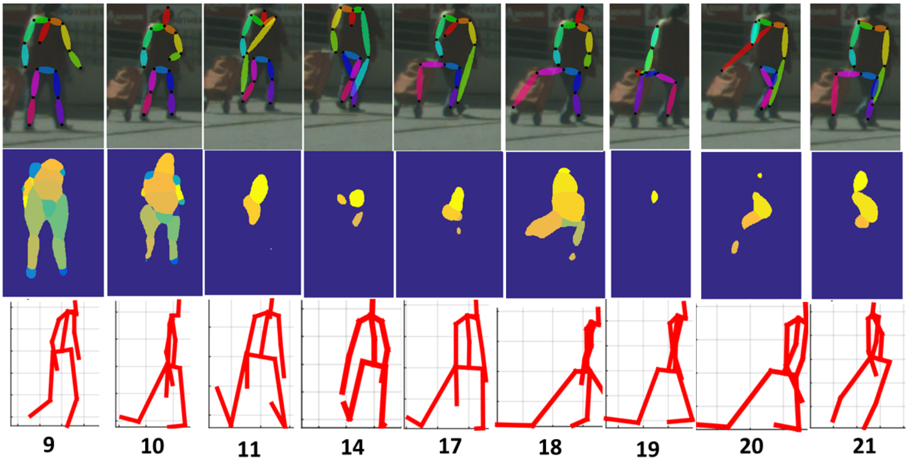


Fig. 26. Consequtive pedestrian detections by FRCNN followed by 2D pose(top row), body parts segmentation (middle row) and 3D pose estimates(bottom row). No pedestrian is detected in frames 12,13 and 15,16. The FRCNN BBoxes jump around the pose estimates to jump. When the pedestrians head is not visible then the body part segmentation fails. The 3D poses do not resemble the true 3D pose as the human appears to be crawling on knees in 3D poses. The 2D poses jump from frame to frame.

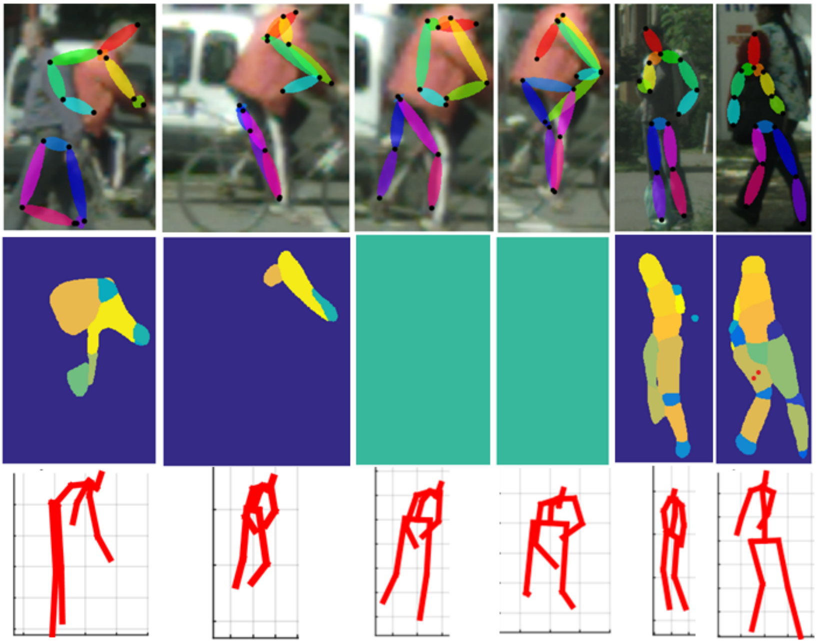


Fig. 27. Some varied results of DMHS.DMHS gets confused in the case of multiple occluding humans. And appears to have trouble with body part segmentation when a human is on a bike. The 2D body pose estimate seems to be greatly affected by the poorly fitting FRCNN bboxes that leave out the pedestrians head.