https://doi.org/10.15388/vu.thesis.776
https://orcid.org/0000-0002-7906-5688

VILNIUS UNIVERSITY

Shubham Anoop Juneja

Investigation of Pre-training in Imitation Learning-based Autonomous Driving

DOCTORAL DISSERTATION

Natural Sciences,
Informatics (N 009)

VILNIUS 2025

The dissertation was prepared between 2020 and 2024 at Vilnius University.

Academic supervisor – Prof. Dr. Virginijus Marcinkevičius (Vilnius University, Natural Sciences, Informatics – N 009).
Academic consultant – Dr. Povilas Daniušis (Vytautas Magnus University, Natural Sciences, Informatics – N 009).

This doctoral dissertation will be defended at a public meeting of the Dissertation Defence Panel:

Chairman – Prof. Habil. Dr. Gintautas Dzemyda (Vilnius University, Natural Sciences, Informatics – N 009).
Members:
Prof. Dr. Gintautas Daunys (Vilnius University, Natural Sciences, Informatics – N 009).
Prof. Dr. Raimundas Matulevičius (University of Tartu, Estonia, Natural Sciences, Informatics – N 009).
Prof. Dr. Darius Plikynas (Vilnius University, Natural Sciences, Informatics – N 009).
Prof. Dr. Artūras Serackis (Vilnius Tech, Technological Sciences, Informatics Engineering – N 007).

The dissertation shall be defended at a public meeting of the Dissertation Defense Panel at 1 p.m. on 27th June 2025 in room 203 of the Institute of Data Science and Digital Technologies of Vilnius University.
Address: Akademijos st. 4, LT-04812, Vilnius, Lithuania
Tel. +370 5 210 9300; e-mail: info@mii.vu.lt

The text of this dissertation can be accessed at the Library of Vilnius University, as well as on the website of Vilnius University:
https://www.vu.lt/lt/naujienos/ivykiu-kalendorius

https://doi.org/10.15388/vu.thesis.776
https://orcid.org/0000-0002-7906-5688

VILNIAUS UNIVERSITETAS

Shubham Anoop Juneja

Paruošiamojo mokymo tyrimas imitaciniu mokymu paremtame autonominiame vairavime

DAKTARO DISERTACIJA

Gamtos mokslai,
Informatika (N 009)

VILNIUS 2025

Disertacija rengta 2020 – 2024 metais Vilniaus universitete.

Mokslinis vadovas:
prof. dr. Virginijus Marcinkevičius (Vilniaus universitetas, gamtos mokslai, informatika – N 009).
Mokslinis konsultantas:
Dr. Povilas Daniušis (Vytauto Didžiojo universitetas, gamtos mokslai, informatika – N 009).

Gynimo taryba:
Pirmininkas – prof. habil. dr. Gintautas Dzemyda (Vilniaus universitetas, gamtos mokslai, informatika – N 009).
Nariai:
prof. dr. Gintautas Daunys (Vilniaus universitetas, gamtos mokslai, informatika – P 009),
prof. dr. Raimundas Matulevičius (Tartu universitetas, Estija, gamtos mokslai, informatika – P 009),
prof. dr. Darius Plikynas (Vilniaus universitetas, gamtos mokslai, informatika – P 009),
prof. dr. Artūras Serackis (Vilniaus Gedimino technikos universitetas, technologijos mokslai, informatikos inžinerija - T 007).

Disertacija ginama viešame Gynimo tarybos posėdyje 2025 m. birželio 27 d. 13 val. Vilniaus universiteto Duomenų mokslo ir skaitemeninių technologijo intituto 203 auditorijoje. Adresas: Akademijos g. 4, LT-04812, Vilnius, Lietuva, tel. +370 5 210 9300; el. paštas: info@mii.vu.lt.

Disertaciją galima peržiūrėti Vilniaus universiteto bibliotekoje ir Vilniaus universiteto interneto svetainėje adresu: *https://www.vu.lt/lt/naujienos/ivykiu-kalendorius*.

ACKNOWLEDGEMENTS

I am deeply grateful to Dr. Virginijus Marcinkevičius, my advisor, for being the guiding light throughout my Ph.D. research. His constant and reliable presence made it possible to tackle deadlines, complete courses, and navigate the entire journey from creating a Ph.D. plan to publishing papers and completing my dissertation. His periodic motivational talks during our meetings kept me going forward.

I am also deeply indebted to my advisor from Neurotechnology and Vytautas Magnus University, Dr. Povilas Daniušis, for constantly inspiring me to pursue a Ph.D. and making it possible through his efforts in submitting applications to Lietuvos Mokslo Taryba two years in a row. I appriciate his continuous mentorship, seeding my work with countless research ideas and directions, and being my only sense of community in research. Above all, I thank him for not giving up on me.

I'd also like to thank Lietuvos Mokslo Taryba and our faculty staff for enabling this research. Last but not least, I am grateful to my family, friends, therapist and ex-colleagues from Neurotechnology for listening to me complain about my Ph.D. ordeals over the past years.

ABSTRACT

Autonomous vehicles (AVs) promise transformative changes in transportation, with SLAM-based methods enabling map-based navigation and learning-based approaches leveraging neural networks for data-driven decisions. While SLAM provides map-based navigation, learning-based methods leverage neural networks for data-driven decisions. This study centres on imitation learning within the learning-based paradigm, specifically addressing its limitation of covariate shifts. The aim is to develop autonomous navigation systems using deep learning and imitation learning, emphasising pre-training techniques. This research starts with reviewing state-of-the-art imitation learning methods and pointing out how pre-training in autonomous driving is under-explored. Most of the approaches in this area of research choose visual encoders pre-trained on the task of ImageNet classification, rather than searching for better alternative approaches. Therefore, the study proposes application of pre-training methods new to the task of end-to-end autonomous driving. It then evaluates these methods against baseline approaches to demonstrate enhanced performance. The first proposed method is termed Visual Place Recognition (VPR) pre-training. It uses VPR as a pre-training task to improve the robustness of autonomous driving agents under varying weather and lighting conditions. It integrates a ResNet-based encoder trained with triplet loss and semantic segmentation for better scene understanding. The empirical evaluations of this method demonstrate a 60.25% route completion surpassing the baseline which achieves 53.20%, in unseen environments and settings. The evaluations result in showing enhanced robustness to covariate shifts and reducing errors in unseen settings, in comparison to baseline approaches. The second proposed method uses the DINO (self-distillation with no labels) method for pre-training. Adopting a self-supervised learning approach, this method trains encoders on ImageNet without labels, allowing for richer feature extraction and better generalisation. The empirical evaluations of this method demonstrate a 62.18% route completion surpassing the baseline which achieves 53.20% in unseen environments and settings, while also surpassing the VPR pre-training method. This research underscores the potential of advanced pre-training methods in overcoming the limitations of traditional ImageNet classification-based pre-training in autonomous driving.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS 5
ABSTRACT 6
ABBREVIATIONS 16
GLOSSARY 18
INTRODUCTION 19
Research Area 19
Research Object 20
Research Aim and Objectives 20
Research Methods 21
Scientific Novelty 21
Practical Significance 22
Statements to be Defended 23
Approbation and Publications of the Research 23
Outline of the Dissertation 24
1. End-to-End Autonomous Driving: Literature Review 26
1.1. Introduction to Autonomous Vehicles 26
1.1.1. What are Autonomous Vehicles? 27
1.1.2. Implications of Autonomous Vehicles 28
1.2. Autonomous Driving 30
1.2.1. Modular Approach 30
1.2.2. End-to-End Approach 31
1.2.3. Comparison of Modular and End-to-End Approaches 32
1.3. End-to-End Learning Methods 33
1.3.1. Imitation Learning for Autonomous Driving . . . 34
1.3.2. Reinforcement Learning for Autonomous Driving 37
1.4. Foundations of Imitation Learning-Based Autonomous Driving 39

1.4.1. Earliest Application of a Neural Network for Autonomous Driving . . . 40
1.4.2. Introducing CNNs to Driving . . . 40
1.4.3. Adoption of Modern CNN . . . 41
1.4.4. Dataset Aggregation . . . 43
1.5. Advances in End-to-End Autonomous Driving . . . 44
1.5.1. Architectural Advancements . . . 45
1.5.2. Data-Centric Advancements . . . 48
1.6. Pre-training . . . 50
1.6.1. Pre-training in Autonomous Driving . . . 52
1.6.2. Potential Pre-training Paradigms . . . 55
1.7. Autonomous Driving Research in Lithuania . . . 57
1.8. Conclusions of Chapter 1 . . . 58
2. Research Methodology . . . 60
2.1. Visual Place Recognition Pre-training . . . 60
2.1.1. Pre-training of the Visual Encoder using VPR . . . 61
2.1.2. VPR Pre-trained Agent Training . . . 64
2.2. DINO Pre-training . . . 68
2.2.1. Pre-training Visual Encoder using DINO . . . 70
2.2.2. DINO Agent Training . . . 72
2.3. Experimental Setup . . . 74
2.3.1. Simulation Environment . . . 74
2.3.2. Benchmark Standard . . . 76
2.3.3. Experimentation Design . . . 76
2.3.4. Data Collection . . . 77
2.3.5. Baseline Methods . . . 79
2.3.6. Implementation Details of Proposed & Baseline Methods . . . 81
2.3.7. Training Details . . . 83

2.3.8. Metrics 84
2.4. Conclusions of Chapter 2 86
3. Empirical Investigation 88
3.1. Visual Place Recognition Pre-training Experiments 88
3.2. DINO Pre-training for Autonomous Driving Experiments 94
3.3. Extended Analysis 99
3.3.1. Comparison of the Proposed Methods 102
3.4. Conclusions of Chapter 3 103
GENERAL CONCLUSIONS 105
BIBLIOGRAPHY 106
LIST OF AUTHOR PUBLICATIONS 120
CURRICULUM VITAE 121
SUMMARY IN LITHUANIAN 122

LIST OF TABLES

1.1 Research over the years solely relying on ImageNet-based pre-training. 53

2.1 Distribution of towns for training, evaluation and testing, following the benchmark standard [137]. 77

2.2 Distribution of weather conditions for training, evaluation and testing, following the benchmark standard [137]. 79

3.1 Highest route completion (%) scores of driving agents under training and new (testing) conditions, across all DAgger iterations reported. 89

3.2 Highest distance completion (%) scores of driving agents under training and new (testing) conditions across all DAgger iterations reported. 90

3.3 Route completion (%) of every DAgger iteration under train town & weather conditions. 93

3.4 Distance completion (%) of every DAgger iteration under train town & weather conditions. 93

3.5 Route completion (%) of every DAgger iteration under new town & weather conditions. 94

3.6 Distance completion (%) of every DAgger iteration under new town & weather conditions. 95

3.7 Highest route completion (%) of driving agents under training and new (testing) conditions, across all DAgger iterations reported. 98

3.8 Highest distance completion (%) of driving agents under training and new (testing) conditions, across all DAgger iterations reported. 98

3.9 Number of collisions and infractions across the compared pre-training methods normalised by per distance travelled per kilometre (lower is better). 102

S.1 Vairavimo agentų geriausi maršruto įveikimo rezultatai (%) mokymo ir naujomis (testavimo) sąlygomis visose DAgger iteracijose. 143

S.2 Vairavimo agentų geriausi atstumų įveikimo rezultatai (%) mokymo ir naujomis (testavimo) sąlygomis visose DAgger iteracijose. 144

S.3 Vairavimo agentų geriausi maršruto įveikimo rezultatai (%) mokymo ir naujomis (testavimo) sąlygomis visose DAgger iteracijose. 148

S.4 Vairavimo agentų geriausi atstumų įveikimo rezultatai (%) mokymo ir naujomis (testavimo) sąlygomis visose DAgger iteracijose. 151

S.5 Lyginamų paruošiamojo mokymo metodų susidūrimų ir pažeidimų dažnis, normalizuotas pagal nuvažiuotą atstumą vienam kilometrui. 152

LIST OF FIGURES

1.1 Modular pipeline for autonomous driving [113]. 31

1.2 End-to-End pipeline for autonomous driving [113]. . . . 32

1.3 Imitation learning transfers behaviours by extracting datasets from demonstrations and training machine learning methods. 35

1.4 Divergence of an imitation learning trained agent from the path exposed at train time (red) to a new path (blue) due to erroneous decisions. 36

1.5 A representation of covariate shift across training and testing set distributions. 37

1.6 Steps taken in the DAgger method starting with an initial dataset [107]. 44

1.7 Conditional Imitation Learning [33] divides the decision-making into multiple branches. 46

1.8 DARB [104] starts with an initial dataset, then extends the existing DAgger method by introducing critical states sampling and a replay buffer pipeline. 49

1.9 A learning method (e.g. neural network) can be pre-trained on a source task (left) and later it can be fine-tuned on a target task (right). 51

2.1 The figure illustrates the SegVPR decoder structure consisting of a segmentation decoder, multi-scale attention module and pooling module. 62

2.2 The figure illustrates the multi-scale attention module used in SegVPR that employs multiple spatial scales to capture objects of different sizes, and produces an attention map. 62

2.3 The figure illustrates the overall block diagram of the proposed visual place recognition pre-training method, where at first, an image encoder is pre-trained on the VPR task (top) followed by weight transfer to train for the task of end-to-end driving (bottom). 65

2.4 The figure illustrates the overall block diagram of the DINO pre-training method (top), using a teacher-student architecture and exponential moving average (EMA) to update the teacher network weights from student network. Teacher and student are trained on crops of the original full size image. Later illustrating weight transfer to train for the task of end-to-end driving (bottom). . . . 69

2.5 The image shows the CARLA simulator's capability of simulating a real-world environment with traffic, pedestrians, environmental elements, and weather conditions. 75

2.6 The figure shows two weather conditions for evaluation (a) and (b), that are used as a part of evaluation set to test in known conditions, followed by weather conditions (c) and (d), that are unseen by the agent and used in testing. 78

2.7 General architecture of ResNet34 and ResNet50 configurations [51], mentioning the arrangements of the residual blocks, filter sizes and depth at each convolution layer. . 80

3.1 Route completion (%) of agents on the Leaderboard benchmark under training conditions (top) and testing conditions (bottom), evaluated three times over different seeds and plotted along with the average of performance. . . . 91

3.2 Distance completion (%) of agents on the Leaderboard benchmark under training conditions (top) and testing conditions (bottom), evaluated three times over different seeds and plotted along with the average of performance. 92

3.3 Route completion (%) of agents on the Leaderboard benchmark under training conditions (top) and testing conditions (bottom), evaluated three times over different seeds and plotted along with the average of performance. . . . 96

3.4 Distance completion (%) of agents on the Leaderboard benchmark under training conditions (top) and testing conditions (bottom), evaluated three times over different seeds and plotted along with the average of performance. 97

3.5 Number of collisions with static elements per kilometre over three random seed evaluations, along with the mean (lower is better). 100

3.6 Number of collisions with pedestrians per kilometre over three random seed evaluations, along with the mean (lower is better). 100

3.7 Number of collisions with vehicles per kilometre over three random seed evaluations, along with the mean (lower is better). 101

3.8 Number of red light infractions per kilometre over three random seed evaluations, along with the mean (lower is better). 101

S.1 Paveikslėlyje pavaizduota SegVPR dekoderio struktūra, sudaryta iš segmentavimo dekoderio, daugiamačio dėmesio modulio ir daugiamačio sutelkimo modulio. 131

S.2 Paveikslėlyje pavaizduotas SegVPR architektūroje naudojamas daugiamačio dėmesio modulis, kuris naudoja kelis erdvinius mastelius, kad būtų užfiksuoti skirtingo dydžio objektai, ir būtų sudarytas dėmesio žemėlapis. . . 132

S.3 Paveikslėlyje pavaizduota bendra siūlomo vizualinio vietos atpažinimo paruošiamojo mokymo metodo blokinė schema, kurioje iš pradžių vaizdo enkoderis yra paruošiamojo mokymo būdu apmokomas VPR užduoties (viršuje), po kurio yra atliekamas parametrų perkėlimas, kad būtų galima mokyti ištisinio vairavimo užduočiai (apačioje). . 134

S.4 Paveikslėlyje pavaizduota bendra DINO paruošiamojo mokymo metodo blokinė schema (viršuje), kurioje naudojama mokančiojo tinklo-mokomojo tinklo architektūra ir eksponentinis slenkantis vidurkis (angl. exponential moving average, EMA) mokančiojo tinklo parametrams atnaujinti iš mokomojo tinklo. Mokantysis tinklas ir mokomasis tinklas yra mokomi naudojant originalaus viso dydžio vaizdo iškarpas. Vėliau iliustruojamas parametrų perkėlimas, kad būtų galima mokyti atlikti ištisinio vairavimo užduotį (apačioje). 138

S.5 Agentų maršruto įveikimas (%) naudojant Leaderboard lyginamąjį standartą mokymo sąlygomis (viršuje) ir testavimo sąlygomis (apačioje), įvertintas tris kartus su skirtingais atsitiktinės inicializacijos parametrais ir pavaizduotas kartu su efektyvumo vidurkiu. 145

S.6 Agentų atstumo įveikimas (%) naudojant Leaderboard lyginamąjį standartą mokymo sąlygomis (viršuje) ir testavimo sąlygomis (apačioje), įvertintas tris kartus su skirtingais atsitiktinės inicializacijos parametrais ir pavaizduotas kartu su efektyvumo vidurkiu. 146

S.7 Agentų maršruto įveikimas (%) naudojant Leaderboard lyginamąjį standartą mokymo sąlygomis (viršuje) ir testavimo sąlygomis (apačioje), įvertintas tris kartus su skirtingais atsitiktinės inicializacijos parametrais ir pavaizduotas kartu su efektyvumo vidurkiu. 149

S.8 Agentų atstumo įveikimas (%) naudojant Leaderboard lyginamąjį standartą mokymo sąlygomis (viršuje) ir testavimo sąlygomis (apačioje), įvertintas tris kartus su skirtingais atsitiktinės inicializacijos parametrais ir pavaizduotas kartu su efektyvumo vidurkiu. 150

ACO	Action Conditioned Contrastive Policy Pre-training
AD	Autonomous Driving
ALVINN	Autonomous Land Vehicle In a Neural Network
AWS	Amazon Web Services
BAR34IC	Baseline Agent with ResNet34 encoder trained on ImageNet Classification
BAR50IC	Baseline Agent with ResNet50 encoder trained on ImageNet Classification
BEV	Bird's-Eye View
BERT	Bidirectional Encoder Representations from Transformers
BYOL	Bootstrap Your Own Latent
CARLA	Car Learning to Act
CILRS	Conditional Imitation Learning with ResNet
CNN	Convolutional Neural Network
DAgger	Data Aggregation
DARB	DAgger with Replay Buffer
DASGIL	Domain Adaptation for Semantic and Geometric-aware Image-based Localisation
DINO	Self-Distillation with No Labels
EMA	Exponential Moving Average
GPU	Graphics Processing Unit
GPS	Global Positioning System
GTA	Grand Theft Auto
IDM	Inverse Dynamics Model
ILSVRC	Large Scale Visual Recognition Challenge
LBC	Learning By Cheating
LLM	Large Language Model
MDP	Markov Decision Process
NLP	Natural Language Processing
NHTSA	National Highway Traffic Safety Administration
NVME	Non-Volatile Memory Express
PaLM	Pathways Language Model

PPGeo	Policy Pre-training for Autonomous Driving via Self-Supervised Geometric Modelling
RL	Reinforcement Learning
RGB	Red Green Blue
SGD	Stochastic Gradient Descent
SimCLR	Simple Framework for Contrastive Learning of Visual Representation
SLAM	Simultaneous Localisation and Mapping
TNG	Topological Navigation Graph
VKT	Vehicle Kilometres Travelled
ViT	Vision Transformer
VPR	Visual Place Recognition
VPT	Video Pre-training
YUV	Luminance, Blue Projection and Red Projection

Agent	An entity that perceives its environment and takes actions to maximise a goal.
Autonomous Driving	The capability of a vehicle to navigate and operate without human intervention using sensors and decision-making algorithms.
Covariate shift	A distribution shift where the input distribution changes between training and deployment while the target function remains unchanged.
DAgger	Dataset Aggregation, an imitation learning algorithm that iteratively collects expert demonstrations to correct mistakes and improve policy performance.
Descriptor	A feature vector or signature representing key characteristics of an image, object, or data point.
DINO	Self-Distillation with No Labels, a self-supervised learning method for training vision transformers without labelled data.
Expert	A model or human providing high-quality demonstrations or decisions, often used in imitation learning.
ImageNet	A large-scale image dataset used for training deep learning models, particularly in computer vision.
Representation	A transformed version of raw data that preserves relevant features for learning and decision-making.
Pre-training	The process of training a model on a large dataset before fine-tuning it on a specific task, improving performance and generalisation.
Visual Encoder / Vision Encoder	A neural network component that processes images into feature representations for tasks like classification, detection, or scene understanding.
VPR	Visual Place Recognition, a technique for recognising previously visited locations using visual data.

INTRODUCTION

RESEARCH AREA

Autonomous driving technology represents a transformative shift in transportation, promising to redefine automation, road safety, traffic efficiency, and accessibility. The need for research in this field is driven by the complexity and potential impact of self-driving vehicles. Central to this research is the challenge of developing reliable systems that can perceive, understand, and navigate in diverse and unpredictable environments. Advances in cutting-edge algorithms, particularly in the areas of computer science and robotics, form the backbone of how autonomous vehicles interpret sensor data to make real-time decisions. Safety remains a major concern, navigating research to explore robust algorithmic developments that can handle scenarios ranging from dense traffic to severe weather conditions, ensuring that autonomous vehicles can operate safer than human drivers in all conditions.

Automating the ability to navigate has been approached through various conceptual paradigms in autonomous driving and robotics research. The two most influential paradigms have been SLAM (Simultaneous Localisation and Mapping) based and learning-based methods. SLAM-based algorithms help vehicles build and update a map of an unknown environment while simultaneously keeping track of their location within it, crucial for real-time navigation in complex and dynamic settings. On the other hand, the rise in artificial intelligence capabilities has led to an increased reliance on learning-based approaches, particularly through use of neural network-based systems. These methods leverage massive datasets to teach systems how to perceive, decide, and act in diverse driving scenarios, enhancing their ability to make split-second decisions. Together these methodologies underpin the significant strides in autonomous driving research.

Building on the advancements previously discussed, this thesis seeks to extend the research by introducing new perspectives and approaches. This research starts with reviewing state-of-the-art imitation learning

methods and pointing out how pre-training in autonomous driving is under-explored. The majority of approaches in this area of research choose visual encoders pre-trained on the task of ImageNet classification, rather than searching for better alternative approaches. Therefore, the study proposes application of pre-training methods novel to the task of end-to-end autonomous driving. It then evaluates these methods against baseline approaches to demonstrate enhanced performance.

RESEARCH OBJECT

The research object is imitation learning-based autonomous driving techniques with focus on exploration of pre-training methods and their effect over a driving agent's ability to navigate in unseen environment settings.

RESEARCH AIM AND OBJECTIVES

The research aim is to implement and research autonomous driving algorithms based on imitation learning and deep neural networks for autonomously navigating in environment that simulate real world conditions, in-order to explore pre-training techniques and enhance generalisation over seen and unseen environmental settings.

To accomplish the research aim, the following objectives were carried out:

1. Pursue a study of the state-of-the-art methods in imitation learning based end-to-end autonomous driving and identify the current state of pre-training of the visual encoders of autonomous driving agents.
2. Identify and propose a task for pre-training the visual encoder of the driving agent that is better related to the task of driving than the traditionally used ImageNet classification.
3. Identify and propose a self-supervised pre-training task for the autonomous driving agent's visual encoder aimed to generalise bet-

ter than the traditionally used ImageNet classification pre-training approach.

4. Test empirically the proposed methods against appropriate baseline autonomous driving agents and evaluate the results.

RESEARCH METHODS

The research in this thesis was performed based upon these scientific methods:

1. A literature review is conducted outlining imitation learning based autonomous driving methods.
2. Qualitative and quantitative data collection is carried out, adhering to multiple metrics.
3. Proposed methods are evaluated by carrying out multiple experiment reruns with varying random seeds.
4. Constructive research is used to propose enhancements and improvements on the real world problems and new methods to improve the theory are proposed.
5. Software development methods are used to implement the proposed method and the experimental part of this thesis, implementing pre-training and driving algorithms, and, additionally, evaluation systems.

SCIENTIFIC NOVELTY

The thesis contributes to the development of imitation learning-based end-to-end trained autonomous driving methods. The main contributions of the thesis can be outlined as follows:

1. Extending the under-explored research on pre-training methods for end-to-end autonomous driving by proposing to discard the

reliance on supervised image classification pre-training of the visual encoder.

2. This thesis proposes visual place recognition as a pre-training task for autonomous driving. It also empirically shows how such pre-training out performs the commonly used pre-training technique.

3. Another pre-training method for autonomous driving, self-distillation with no labels (DINO) pre-training is proposed and shown to be effective with the support of experiments.

PRACTICAL SIGNIFICANCE

This thesis enhances the performance of autonomous driving methods, additionally makes, training more efficient. The most important practical contributions are the following:

1. The experiments performed using pre-training methods, namely visual place recognition and DINO show higher resistance to changes in the environment when deployed in simulation environments. This means that such practices can lead the way to reliable driving in environments that are not exposed to learner and hence minimising training data requirements.

2. The experiments also show faster convergence to higher performance whenever the proposed methods are trained. This shows reduced expensive GPU compute hours and hence contributes to producing lower carbon footprint.

3. The thesis also makes training code for training autonomous driving methods publicly available, and mentions other important repositories.

4. This thesis provides evidence supporting the hypothesis using industry and research standard tools such as the simulator, machine learning frameworks, etc. which makes the findings easily transferable to on-going research works in industry and academia.

STATEMENTS TO BE DEFENDED

The following claims are defended in this thesis:

1. Pre-training the visual encoder over the task of visual place recognition using triplet loss instead of the commonly used classification task on the ResNet architecture enhances the driving performance of imitation learning-based autonomous driving system on route completion and distance completion metrics.
2. Pre-training the visual encoder on the ImageNet dataset using the self-distillation with no labels (DINO) method instead of the commonly used supervised image classification task on the ResNet architecture produces richer features for imitation learning-based autonomous driving which enables better driving performance as per route completion and distance completion metrics.
3. On comparison of the visual place recognition pre-training against DINO pre-training, the DINO pre-training method reports higher performance while proving to be superior in unseen environments, by completing more routes and causing lesser collisions with static elements, pedestrians and vehicles, and also by causing fewer red light infractions.

APPROBATION AND PUBLICATIONS OF THE RESEARCH

The results obtained in this thesis were published in four papers: two in peer-reviewed periodic scientific journals and two at scientific conference proceedings. The following list presents the publications and presentations at conferences:

Papers in periodic scientific journals:

[A.1] Juneja, S., Daniušis, P., & Marcinkevičius, V. (2023). Visual place recognition pre-training for end-to-end trained autonomous driving agent. IEEE access, 11, 128421-128428.

[A.2] Juneja, S., Daniušis, P., & Marcinkevičius, V. (2024). DINO Pre-training for Vision-based End-to-end Autonomous Driving. Baltic Journal of Modern Computing, Vol. 12 (2024), No. 4, pp. 374–386.

Papers (and work presented) in peer-reviewed scientific conference proceedings:

[B.1] Juneja, S., Marcinkevicius, V., & Daniušis, P. Combining Multiple Modalities with Perceiver in Imitation-based Urban Driving. All Sensors 2021. 18^{th} July, 2021. Nice, France.

[B.2] Juneja, S., Daniušis, P., & Marcinkevičius, V. (2024). Monocular Depth Estimation Pre-training for Autonomous Driving. AI Sys 2024. 30^{th} September, 2024. Venice, Italy.

Additional work published during the studies but not included:

[C.1] Daniušis, P., Juneja, Valatka, L., & Petkevičius, L. Topological Navigation Graph framework. Autonomous Robots, vol. 45, no. 5, pp. 633-646.

[C.2] Daniušis, P., Juneja, S., Kuzma, L., & Marcinkevičius, V. (2022). Measuring Statistical Dependencies via Maximum Norm and Characteristic Functions. arXiv preprint arXiv:2208.07934.

OUTLINE OF THE DISSERTATION

This dissertation consists of an introduction, 3 chapters, conclusions, and a summary in the Lithuanian language. The introduction section provides an introduction to the research and an overview of the dissertation. The first chapter presents a literature review covering imitation learning-based autonomous driving methods and related foundational topics, such as autonomous driving, imitation learning, and pre-training approaches. The second chapter describes the proposed methods and the experiments conducted. The third chapter presents and analyses the

results obtained from the experiments. Finally, the conclusions drawn from the presented research are listed in the general conclusions section. The bibliographic references are included at the end of the dissertation. The dissertation consist of 156 pages, 24 figures and 12 tables.

1. END-TO-END AUTONOMOUS DRIVING: LITERATURE REVIEW

This chapter reports a study on the current state of imitation learning and end-to-end autonomous driving. It covers topics starting from the idea of autonomous vehicles and their associated implications on the various branches of research within the field of autonomous driving. It later presents the state-of-the-art methods used in end-to-end autonomous driving and the concept of pre-training. Additionally, this chapter also captures the trends in current state-of-the-art research to justify the direction of our research. This review of literature presents the background of the research published in [A.1], [A.2], [B.1] & [B.2].

1.1. INTRODUCTION TO AUTONOMOUS VEHICLES

Automation has become a pivotal force driving innovation and efficiency across various industries. The integration of automated systems enhances productivity, minimises human error, and enables the execution of complex tasks beyond human capabilities. In the manufacturing sector, the adoption of robotics and automation technologies has revolutionised production processes, leading to significant cost reductions and improved product quality. According to Company [35], the "Factory of the Future" leverages advanced robotics to increase efficiency and flexibility in manufacturing operations. In the financial industry, algorithmic trading has transformed market dynamics by increasing the speed and volume of trades, enhancing market liquidity and efficiency. Hendershott et al. [55] discuss how algorithmic trading improves liquidity and reduces transaction costs in financial markets. The healthcare sector has also benefited from automation through robotic-assisted surgeries, which enhance precision and patient outcomes. Herron [56] highlights the advancements in surgical robotic systems and their impact on modern medicine. These examples underscore the indispensable role of automation in modern society, driving advancements that improve efficiency, safety, and quality across various sectors.

Among the domains benefiting from automation, autonomous vehi-

cles (AVs) stand out as a particularly impactful and rapidly evolving area of research. The development of AVs involves a multidisciplinary approach, integrating artificial intelligence, control technologies, computer vision, and sensor technologies [101]. AVs have the potential to revolutionise transportation by improving road safety, reducing traffic congestion, and providing mobility solutions for those unable to drive [79]. Human error is a significant factor in traffic accidents. The National Highway Traffic Safety Administration (NHTSA) reports that approximately 94% of serious crashes are due to human error [3, 8]. By eliminating the human factor, AVs could significantly reduce the number of traffic accidents. Fagnant and Kockelman [44] discuss the potential benefits of AVs, including reductions in crashes, energy consumption, and parking needs, along with an improved traffic flow.

1.1.1. What are Autonomous Vehicles?

According to Du [43], an autonomous vehicle—also known as a self-piloting auto-mobile, driverless car, computer-driven car, or wheeled mobile robot—is a type of intelligent vehicle controlled by an onboard computer system [42]. Essentially, it is a fast, wheeled autonomous mobile robot that relies on a variety of sensors to perceive its surroundings. These sensors gather information about the vehicle's environment, including road conditions, vehicle position, and nearby obstacles. Based on this data, the computer system autonomously manages the vehicle's movements, enabling it to navigate safely, reliably, and without human intervention. While the term "autonomous vehicle" can encompass a wide range of vehicles, including cars, trucks, buses, and even drones, our research work primarily focuses on autonomous passenger cars due to the significant industry focus on their development and potential impact on personal transportation [43, 97]. With the focus remaining on passenger cars, the majority of this AV technology can be applied on a wider scope and holds potential applications in various domains. For example, autonomous trucks are being developed to improve logistics and long-haul transportation, while autonomous buses could revolutionise public transit systems.

1.1.2. Implications of Autonomous Vehicles

The widespread adoption of AVs promises to fundamentally transform transportation systems and urban environments. Othman [97] comprehensively explores several critical implications of AV deployment, ranging from vehicle ownership patterns and fleet utilisation to urban infrastructure and city planning. These implications suggest potentially profound societal benefits, including enhanced mobility, reduced environmental impact, and more efficient use of urban space, underlining the significance of continued research and development in autonomous driving technologies. The following sections examine the most important of these implications in detail, highlighting both opportunities and challenges that emerge from existing literature.

1.1.2.1. Vehicle ownership and vehicle utilisation

The adoption of autonomous vehicles (AVs) promises significant changes, including a reduction in vehicle ownership and an increase in vehicle utilisation. Studies indicate that vehicle utilisation could increase from around 5% in conventional vehicles [47] up to 75% [19], leading to shorter vehicle lifespans and a faster adoption of newer, cleaner technologies [45]. Simulations have suggested that a single AV could replace more than 10 conventional vehicles when used as a shared mode [97].

1.1.2.2. Optimising passenger wait time

Although shared AVs can reduce fleet size and increase vehicle utilisation, they also provide higher-quality service through lower passenger waiting times. Studies have found that passengers perceive waiting times for current modes of transports as significantly longer than the actual duration, i.e. one min of in-vehicle time could range from 1.4 to 2.5 minutes as per the perception of the passenger [2, 46, 57, 122]. Currently, average transit waiting times in the US and Canada are 40 and 20 minutes respectively [82]. In contrast, shared AVs are projected to offer average waiting times of only 5 minutes, along with lower trip

costs. This suggests that shared AVs could be highly competitive with traditional public transit, potentially attracting many users away from those services. Transit agencies should be aware of this disruptive threat and prepare accordingly as AVs become more widely available.

1.1.2.3. Impact on public behaviour

One of the key advantages of AVs is that passengers will be able to engage in other activities during travel, rather than viewing trip time as an economic loss [97]. However, AVs may also motivate longer trips, increased travel, and additional trips, leading to higher vehicle-kilometres travelled (VKT) [86, 97]. This increased VKT could in turn increase emissions and fuel consumption [32, 88]. Additionally, the low waiting times and costs of AVs could attract new travel demand, further increasing VKT and potentially worsening traffic [86].

1.1.2.4. Road capacities and intersections

AVs have the potential to increase road and intersection capacities by enabling shorter following distances between vehicles and narrower lane widths, due to the high level of vehicle-to-vehicle communication and the elimination of human factors [48]. However, this increased capacity may not be fully realised until AVs achieve high market penetration [86].

1.1.2.5. Land use

AVs have the potential to significantly reduce parking demand and the required number of parking spaces [86, 88, 112]. Studies show AVs could enable up to 2.5x higher parking space utilisation through techniques like vehicle blocking and coordination [112]. This would free up valuable land currently dedicated to parking, allowing it to be repurposed for other uses that can increase property values [32].

1.2. AUTONOMOUS DRIVING

The promising benefits highlighted by the implications of AVs justify the extensive research investment in the development of the technology behind autonomous driving. Meanwhile, enabling vehicles to drive autonomously has been the subject of intensive investigation spanning multiple decades. Autonomous driving, which is often defined as the capacity of a vehicle to navigate and maneuver without direct human control, has had two prominent emerging approaches. These approaches are the modular approach and the end-to-end approach [26, 98, 113]. This section examines the modular and end-to-end approaches to autonomous driving, followed by a comparative assessment of their relative strengths and limitations.

1.2.1. Modular Approach

The modular approach to autonomous driving, also known as the mediated approach [23], decomposes the driving task into a series of specialised components that process information sequentially from sensors to actuators [84]. This architecture comprises several core functional modules: localisation and mapping, perception, assessment, planning and decision-making, vehicle control, and human-machine interface [130]. In a typical implementation [5, 17, 77, 118, 123, 138], sensor data flows through localisation and object detection components, then may proceed to planning and decision-making modules, before finally generating motor commands via the control module [12, 84]. We portray a generic form of the modular approach that accepts data through input sensors, processes it through a pipeline of modules (such as perception module, localisation module, etc.) and produces actuator commands in Figure 1.1 as according to Tampuu et al. [113]. This modular decomposition offers a key advantage: it breaks down the complex challenge of autonomous driving into more manageable sub-problems [30]. Furthermore, these individual components align with established research fields such as robotics, computer vision, and vehicle dynamics, enabling direct application of existing expertise and methodologies from these domains.

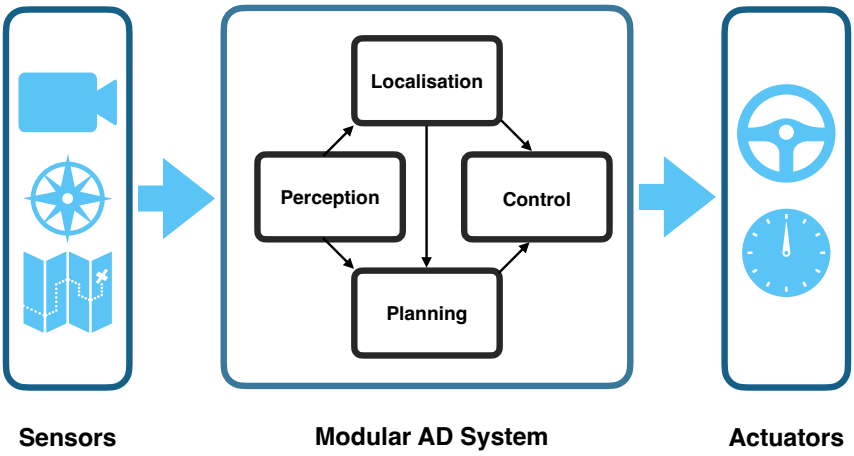


Figure 1.1: Modular pipeline for autonomous driving [113].

1.2.2. End-to-End Approach

The end-to-end approach to autonomous driving employs deep neural networks to learn a direct mapping between sensory inputs and control outputs, treating the entire driving task as a single optimisation problem [24, 98]. The sensory inputs in autonomous driving are usually camera inputs while the control outputs are often driving commands to adjust the steering and acceleration actuators [113]. We portray a generic example of the end-to-end approach in Figure 1.2, where sensor input is processed through a neural network architecture and actuator commands are predicted. This approach heavily relies upon learning from demonstrations based on the experience of a driver or another agent. The end-to-end outlook to driving has gained significant traction with recent advances in deep learning and has formed the primary focus of this research. The training of end-to-end models generally follows one of two distinct learning paradigms [113]: Reinforcement Learning (RL), which develops driving policies through environmental interaction and trial-and-error exploration [111] or imitation learning, which learns driving behaviour by mimicking expert demonstrations through supervised learning [10].

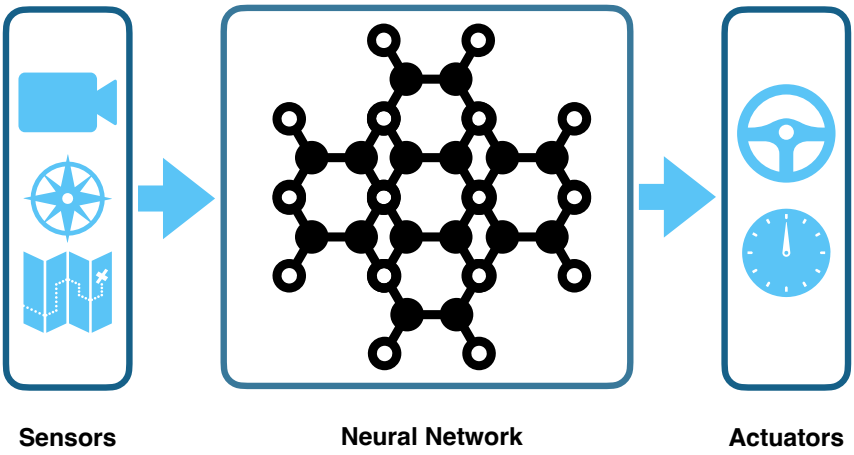


Figure 1.2: End-to-End pipeline for autonomous driving [113].

1.2.3. Comparison of Modular and End-to-End Approaches

The modular approach to autonomous driving offers notable advantages, primarily its interpretability and the ability to develop components independently [113]. This structure allows researchers and engineers to easily trace the source of errors or unexpected behaviours and enables specialised teams to focus on improving specific modules without disrupting the entire system [126, 132]. However, these benefits are counterbalanced by significant challenges. The predefined inputs and outputs of each module can result in sub-optimal performance across diverse driving scenarios, as the rigid architectural design may not adapt seamlessly to complex or unpredictable situations [132]. Moreover, the engineering complexity of designing and maintaining these interconnected modules demands substantial effort and expertise [41, 113]. A critical limitation emerges from the data processing within these modules, where compression techniques like 3D bounding boxes can lead to substantial information loss, potentially compromising the system's ability to make nuanced and effective driving decisions [41, 132].

The end-to-end approach to autonomous driving presents considerable advantages, particularly in its ability to learn task-specific feature representations that can potentially outperform traditional methods (e.g.

module and heuristic-based) [113]. These systems offer a simplified architectural design with fewer components compared to modular approaches, and their success in complex domains like gaming suggests significant potential for autonomous driving [13, 90, 121]. However, these benefits are accompanied by critical challenges. The neural network's black-box nature renders its decision-making process nearly opaque, making it difficult to interpret or diagnose the sources of errors or unexpected behaviours [73, 126, 132]. Moreover, the approach is vulnerable to covariate shift, where the model may struggle to generalise beyond its training scenarios, as its actions continuously reshape the observed environment [10, 34, 104, 107]. An additional significant concern is the model's susceptibility to adversarial attacks, where carefully crafted inputs could potentially manipulate the system's perception and decision-making, raising substantial safety concerns for a technology responsible for human transportation [113].

The end-to-end approach offers a promising research direction due to its potential to overcome the limitations of rigid, compartmentalised modular systems [26, 113]. By learning holistic representations directly from data, end-to-end models can capture complex, nuanced driving interactions that traditional modular architectures might miss. The ability of the end-to-end approach to optimise entire driving tasks through advanced machine learning techniques—particularly deep neural networks—suggests a more adaptive and potentially more intelligent solution to autonomous driving challenges. While currently facing interpretability and generalisation challenges, the end-to-end approach represents a fundamental shift towards more flexible, data-driven autonomous driving systems that could ultimately provide superior performance by learning from comprehensive driving experiences. Motivated by these advantages, our research focuses on end-to-end approaches to further explore their potential in autonomous driving.

1.3. END-TO-END LEARNING METHODS

Further exploration of end-to-end methods is central to this research. In this section, we review the two prominent end-to-end methods of autonomous driving introduced in Section 1.2.2.

1.3.1. Imitation Learning for Autonomous Driving

End-to-End autonomous driving systems commonly rely on imitation learning, also known as behaviour cloning [113]. This approach uses an extended version of supervised learning techniques where a machine learning model learns to replicate expert actions. For autonomous vehicles, the system observes and learns from human drivers, attempting to reproduce their driving decisions such as steering angles, acceleration, and braking patterns based on sensory inputs. The appeal of imitation learning lies in its straightforward data collection process i.e. recording human driving behaviours, which provides abundant training data. This has proven effective for basic driving tasks, particularly lane following [16, 103]. However, the method faces significant challenges when encountering complex or uncommon traffic situations, as these scenarios are often under-represented in training data [34].

According to Chen et al. [26], Chitta et al. [31], Codevilla et al. [34], Ozaibi et al. [98], Zhang and Cho [133], the objective can be mathematically formulated as:

$$\arg\min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(s,a)\in\mathcal{D}} \mathcal{L}(\pi_\theta(s), a) \tag{1.1}$$

where

- $\pi_\theta(s)$ is the policy parametrised by θ, representing the action predicted by the agent given state s,
- a is the ground-truth action taken by the expert in state s,
- $\mathcal{D}$ is the dataset consisting of state-action pairs (s, a) sampled from the expert's demonstrations,
- $\mathcal{L}$ is the loss function, which measures the discrepancy between the actions predicted by the policy $\pi_\theta(s)$ and the actual actions a taken by the expert.

The dataset $\mathcal{D}$ can be generated by collecting demonstrations performed by an expert human [16], heuristic-based method [41] or an

another trained policy [24, 137]. The method that generates such a dataset is often referred to as expert policy π_E. Therefore, the objective of imitation learning is to train the agent's policy π_θ to closely approximate the expert policy π_E. In modern methods, the choice of learning method to estimate π_θ is often a neural network algorithm [26, 113]. We show the schematics of imitation learning in the form of a diagram in Figure 1.3.

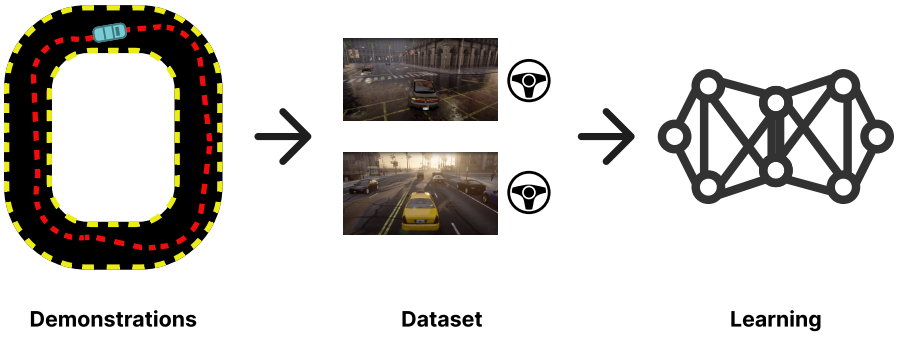


Figure 1.3: Imitation learning transfers behaviours by extracting datasets from demonstrations and training machine learning methods.

1.3.1.1. Challenges

Despite of the fact that imitation learning is a straight forward approach in terms of training models, it faces multiple challenges when it comes to scaling and improving. Here we list out the major challenges:

- **Covariate shift:** While the model effectively learns to replicate expert responses to situations created by human drivers, it faces challenges when dealing with scenarios resulting from its own decisions. This creates a recursive challenge: the model's driving choices influence subsequent observations, requiring it to navigate situations that emerge from its own actions rather than expert demonstration. We show this divergence in Figure 1.4. This phenomenon, known as the covariate shift problem (or distribution shift) [42], [43], occurs when the autonomous system encounters scenarios that deviate from its training examples. The alternate

name distribution shift represents the literal change in data distribution as shown in Figure 1.5. The discrepancy between real-world driving conditions and the training data can lead to significant challenges. For instance, if the training data predominantly features optimal driving positions - such as maintaining perfect lane centring - the model may lack the necessary experience to execute corrective manoeuvrers when it deviates from these ideal conditions.

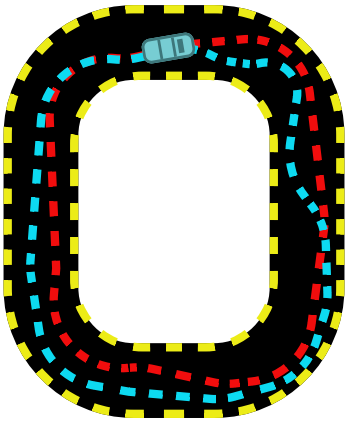


Figure 1.4: Divergence of an imitation learning trained agent from the path exposed at train time (red) to a new path (blue) due to erroneous decisions.

- **Inability to improve beyond expert demonstrations:** By definition, imitation learning models can only learn to perform as well as the expert demonstrations they are trained on [137]. They cannot discover novel or more efficient strategies that might surpass human capabilities.
- **Dataset bias:** Imitation learning for self-driving is susceptible to dataset bias because datasets are often dominated by common behaviours like driving straight at a constant speed [113]. This can lead to the model over-fitting to common cases and struggling to generalise to rare, more complex scenarios [34, 37, 117].
- **Domain shift:** Imitation-learned driving policies often suffer significant performance drops when deployed in environments that

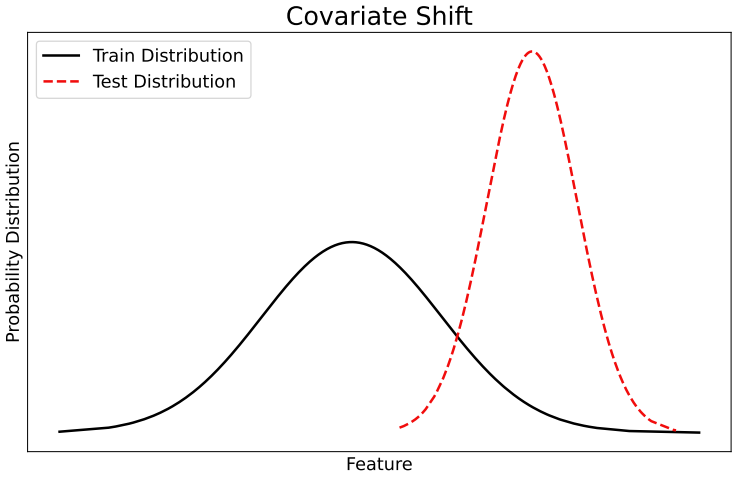


Figure 1.5: A representation of covariate shift across training and testing set distributions.

differ from their training domain (e.g., transferring from simulation to real-world). Even minor discrepancies between the expert demonstration domain and the learner's deployment domain can mislead the model and result in unstable control behaviour [93].

1.3.2. Reinforcement Learning for Autonomous Driving

RL [98, 111] represents a machine learning approach where an intelligent agent develops strategies through iterative environmental interaction. The method is formalised using the Markov Decision Process (MDP) framework, which characterises learning through a structured set of components, which are [98]:

- $\mathcal{S}$: the set of states that can be taken by an agent,
- $\mathcal{A}$: as the set of actions that an agent can perform,
- $P(s' \mid s, a)$: as a transition probability function which gives out the probability of reaching state s' from state s by performing action a,

- $\mathcal{R}(s, a)$: as a reward function that produces reward values based on the state of the agent and the last action performed.

In this framework, an agent navigates through a series of states, selecting actions that maximise cumulative rewards. The policy $\pi(s)$ serves as the agent's decision-making strategy, mapping observed states to chosen actions. These policies can be deterministic, selecting specific actions for each state, or stochastic, probabilistically determining action selection. Central to RL is the concept of return ($\mathcal{G}_t$), representing the cumulative discounted rewards from a given time step. The introduction of a discount factor γ (where $0 \leq \gamma < 1$) ensures that immediate rewards are prioritised while preventing the return from diverging to infinity, thus maintaining mathematical stability and convergence. The transition function $P(s' \mid s, a)$ and reward function $\mathcal{R}(s, a)$ provide the mathematical scaffolding, enabling the agent to learn optimal decision-making strategies through systematic exploration and exploitation of the environment. The total return is given by the following equation:

$$\mathcal{G}_t = \mathcal{R}_{t+1} + \gamma \mathcal{R}_{t+2} + \gamma^2 \mathcal{R}_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k \mathcal{R}_{t+k+1} \tag{1.2}$$

The goal of RL is to find an optimal policy π^* that maximises the expected return for each state s and action a. By iterating over different policies and updating them based on observed returns, the agent learns to choose actions that maximise the expected cumulative reward over time. While we present the core formulation of the reinforcement learning algorithm, recent works have preferably used the evolved versions such as deep Q-learning [134] and policy gradient methods [40, 85].

Reinforcement learning driving policies can utilise identical input structures as those employed in imitation learning systems. The output architecture can remain consistent with imitation learning models, eliminating the need for structural modifications to the neural network. The key distinction lies in the learning mechanism - RL systems derive their learning signals from computed rewards at each time step, rather than relying on human demonstrations. This removes the requirement for collecting and labelling expert driving data.

1.3.2.1. Challenges

RL based methods face multiple challenges in being a viable option for extending the research related to autonomous driving [113]. We list them as follows:

- **Learning in the real world is difficult and can be dangerous:** A crucial challenge in training RL policies for driving is ensuring sufficient exploration without damaging the vehicle or other objects.
- **Simulations are not perfect representations of reality:** While training and testing models in simulated environments like CARLA [41] and GTA V is common practice, transferring these models to the real world presents challenges. Input discrepancies between simulations and reality can lead to poor generalisation, requiring adaptation techniques.
- **Defining appropriate rewards can be complex:** In RL, the model aims to maximise rewards, making the selection of positive and negative rewards crucial for shaping desired behaviours.
- **RL can be less data-efficient than imitation learning:** RL typically requires more data and longer training times compared to imitation learning.

1.4. FOUNDATIONS OF IMITATION LEARNING-BASED AUTONOMOUS DRIVING

The state-of-the-art advancements in end-to-end learning for autonomous driving build upon numerous landmark studies that have established a foundation. In this section, we highlight the key works that have significantly influenced the progression of research leading to this study. While we highlight the studies, we do not compare their performance metrics, as evaluation methods vary widely and continue to evolve.

1.4.1. Earliest Application of a Neural Network for Autonomous Driving

One of the earliest and most influential works in end-to-end learning for autonomous driving is ALVINN (Autonomous Land Vehicle In a Neural Network), developed by Pomerleau [103] in the late 1980s. This system used a three-layer back-propagation neural network to learn the task of road following. ALVINN took input from a video camera and a laser range finder, processing the visual and range data to output the desired steering direction for the vehicle. The network was trained primarily on simulated road images, due to the logistical challenges of collecting a large and diverse real-world dataset. However, successful tests were conducted on the Carnegie Mellon NAVLAB, an autonomous navigation test vehicle, demonstrating ALVINN's capability to navigate real-world roads under certain conditions. Significantly, ALVINN exhibited the ability to adapt its internal representation based on the training data, developing distinct features for roads of fixed width versus those with varying widths, highlighting the flexibility of the neural network approach.

While the system's performance was limited by the computational capabilities of the time, ALVINN laid critical groundwork for future end-to-end autonomous driving research. The significance of ALVINN extends beyond its immediate technical achievements. It provided a conceptual breakthrough by showing that machine learning could potentially replace hand-coded driving rules, a radical idea at the time that would become increasingly influential in subsequent decades of autonomous driving research.

1.4.2. Introducing CNNs to Driving

Building on the foundational ideas introduced by ALVINN, Muller et al. [92] proposed taking end-to-end learning a step further by addressing the unique challenges of navigating off-road environments. While ALVINN focused on road-following tasks using a three-layer neural network, this work employed a more advanced 6-layer convolutional

neural network (CNN) to handle the complexities of obstacle avoidance in diverse and unstructured terrains. Instead of a land vehicle, this work demonstrated the abilities of the proposed method on a 50 cm-long robot truck.

Another key distinction from ALVINN is the use of stereo cameras as input, which allows the system to gather richer spatial information about the environment. CNN enabled processing of raw YUV images from two forward-facing cameras mounted on a small robot truck. The use of CNN instead of a fully connected neural network leveraged the localised feature extraction capabilities of CNNs, enabling it to handle higher-resolution images efficiently. This architectural shift significantly enhanced the ability to detect obstacles and predict steering angles in real time, even under challenging conditions.

The training process built on ALVINN's supervised learning approach, using human drivers to demonstrate obstacle avoidance, but expanded it to more diverse off-road scenarios with varying lighting, weather, and terrains. By mapping raw stereo camera inputs directly to steering commands, the system enabled bypassing traditional feature engineering and depth map calculations, showcasing a design philosophy aimed at adaptability and robustness. Despite a 35.8% classification error rate on the test set, the system effectively showed navigation through obstacles in real-world scenarios, demonstrating strong generalisation capabilities. This work significantly extended ALVINN's foundational ideas, proving the viability of end-to-end learning for off-road navigation in the case of a robot truck. The use of CNNs as a learner for such tasks has since become a standard in this area of research.

1.4.3. Adoption of Modern CNN

CNNs revolutionised this area of research by automating feature extraction from raw data, enabling the detection of complex patterns and relationships directly from training examples. This transformation was driven by two key advancements. The first was the availability of large, labelled datasets, particularly the Large Scale Visual Recognition Challenge (ILSVRC), commonly known as the ImageNet dataset [74].

The second was the emergence of massively parallel GPUs, which substantially improved the efficiency of deep network training. A pivotal demonstration of these advancements came from Krizhevsky et al. [75], who achieved remarkable image classification accuracy on the ILSVRC challenge using the ImageNet dataset.

AlexNet [75] introduced and popularised several key concepts, such as ReLU activation, dropout regularisation, overlapping max pooling, and a deep network architecture. These contributions not only enabled AlexNet's success but also laid the groundwork for subsequent advancements in application convolutional neural networks such as object detection [62], semantic segmentation [52], and others. By significantly enhancing the capability and efficiency of CNNs, AlexNet accelerated progress in computer vision and related fields. In contrast, the "Off-road Obstacle Avoidance through End-to-End Learning" research [92], conducted nearly half a decade earlier, predated these advancements and thus lacked many of the concepts that later improved the effectiveness and flexibility of CNNs.

The work of Bojarski et al. [16] has exemplified this leap, applying a deep CNN with the new advancements to steer a car in varied conditions using raw video input from a single camera. This system effectively eliminated intermediate steps like explicit lane detection or semantic abstraction performed in the alternate methods at the time, by optimising all stages simultaneously for better performance.

The network architecture included five convolutional layers followed by three fully connected layers. The system achieved this with minimal explicit supervision, using only steering angles as training signals while autonomously learning internal representations of road features. Before deployment, the model's performance was tested in a simulation environment that used pre-recorded videos to evaluate how the network responded to various driving scenarios. Once validated, the model was deployed on the NVIDIA Drive PX platform and tested on real roads under diverse conditions, including highways, residential streets, and unmarked or unpaved roads. This research has led to the use of deep architectures in most of the following works that work on autonomous driving.

1.4.4. Dataset Aggregation

Early end-to-end autonomous driving methods, such as ALVINN [103], employed supervised learning to learn the driving task. According to Bagnell [10] and Ross et al. [107], the classical supervised learning approach does not meet the requirements of imitation learning. The distinction between supervised learning and imitation learning lies in their objectives and operational contexts. Supervised learning is typically used in static settings where the goal is to minimise prediction errors in independent and identically distributed (i.i.d.) data, and the model's outputs do not influence the data distribution. In contrast, imitation learning aims to mimic a teacher's decisions in dynamic environments where the learner's actions can affect future observations, creating a feedback loop. This fundamental difference means that errors in imitation learning can cascade, leading to compounding mistakes, unlike in supervised learning where errors remain isolated. This corresponds to the concept shown in Figure 1.4.

The Dataset Aggregation (DAgger) [107] algorithm is a prominent solution to the cascading error problem. Unlike naive supervised learning approaches that train only on the expert's demonstrations, DAgger iteratively refines the policy by interleaving execution and learning. During each iteration, the current policy is executed in the environment, and the expert provides corrective actions whenever the policy diverges from optimal behaviour. These corrections are aggregated into a growing dataset, which is then used to retrain the policy. We represent the process in algorithm form in Algorithm 1. By allowing the policy to encounter states it would encounter under its own control rather than solely under the expert's guidance, DAgger ensures that the training data reflects the distribution of states the policy will actually encounter. This is aimed at avoiding errors that lead to continually exiting the appropriate trajectory, commonly known as error compounding. Therefore, DAgger helps trains the policy to recover from its mistakes, ultimately leading to more robust behaviour that favours generalisability.

Algorithm 1 Dataset Aggregation (DAgger) Algorithm

Initialize policy π_0, dataset $\mathcal{D} \leftarrow \emptyset$
Set expert policy π^* and number of iterations N
for $i = 1$ to N **do**
  Execute current policy π_{i-1} in the environment to generate states $\{s_t\}_{t=1}^{T}$
  Query expert policy π^* to obtain actions $\{a_t^*\}_{t=1}^{T}$ for each state $\{s_t\}_{t=1}^{T}$
  Aggregate data: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t^*)\}_{t=1}^{T}$
  Train a new policy π_i on the aggregated dataset $\mathcal{D}$
end for
Return: Final policy π_N

DAgger and its improved variants [104, 133] have been widely adopted as an approach in the applications of imitation learning due to their ability to address the challenges posed by error compounding and covariate shift. By ensuring the learner encounters and learns from states induced by its own actions, DAgger enables policies to generalise more effectively to real-world scenarios where deviations from expert demonstrations are inevitable. We portray the classical form of DAgger in Figure 1.6.

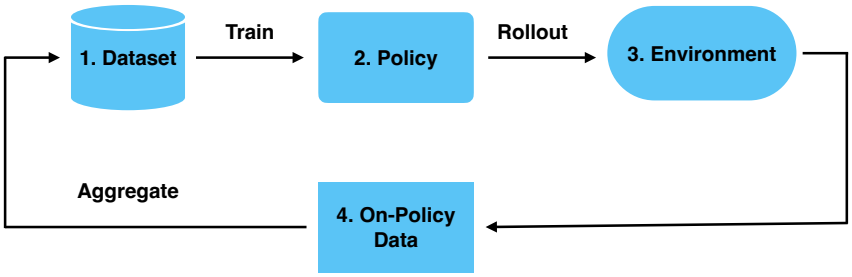


Figure 1.6: Steps taken in the DAgger method starting with an initial dataset [107].

1.5. ADVANCES IN END-TO-END AUTONOMOUS DRIVING

The work by Bojarski et al. [16] showed strong potential in the field of autonomous driving and robotics, concerning the advancements in deep

learning. Therefore, this work served as an invitation for many deep learning researchers to extend the aforementioned areas of research. In this section we highlight the recent advancements that form the current state of research on autonomous driving and partly robot navigation. As these methods rapidly come close to over-saturating their chosen metrics and benchmarks, the benchmark settings for evaluation evolve constantly. Therefore, making it difficult to compare one method to another as the experimentation and evaluation settings change in almost every research work published. We segment these works by the concepts they attempt to improve, such as architectural and data-centric advancements.

1.5.1. Architectural Advancements

One significant advancement in end-to-end learned autonomous driving is the introduction of conditional imitation learning by Codevilla et al. [33]. In their work, the authors propose a model that learns driving behaviours by imitating expert demonstrations while conditioning on high-level navigational commands, such as turning left or right at intersections. This approach allows the vehicle to make decisions based on both visual inputs and intended routes, effectively bridging perception and planning. Their experiments demonstrated that conditional imitation learning enhances the model's ability to handle complex urban driving scenarios, reducing errors associated with ambiguous situations and improving overall driving performance.

Codevilla et al. [33] introduce a conditional imitation learning architecture for end-to-end autonomous driving that employs a multi-branch neural network. This network processes raw sensory inputs, such as camera images, and produces control commands like steering and acceleration. Each branch of the network is conditioned on high-level navigational commands such as turning left, turning right, or going straight which allows the model to learn different behaviours corresponding to each possible route instruction. By integrating perception and decision-making in a unified framework, this architecture enables the model to be trained end-to-end and also navigate complex driving scenarios by following high-level directions while reacting to en-

vironmental cues. A simplified version of the proposed architecture is presented in Figure 1.7, where outputs of an image encoder and a measurement encoder are fused and transformed, then converted to speed and action commands on multiple branches. The further refined version of this architecture by another study [34], using a ResNet [51] for perception (altogether termed CILRS) became a standard in many following studies [24, 104, 126, 127, 137]. The improved work in Codevilla et al. [34] reports a jump from 66% success rate in completing routes (also known as route completion) to 90% with empty traffic conditions in the NoCrash benchmark [34]. Whereas when exposed to dense traffic conditions, the score only improved from 13% to 24%.

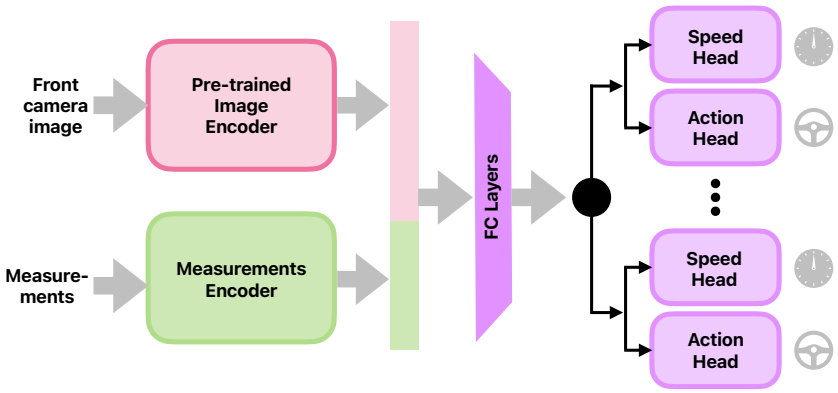


Figure 1.7: Conditional Imitation Learning [33] divides the decision-making into multiple branches.

Extending the work on conditional imitation learning, Xiao et al. [126] introduced an architecture that effectively fuses multiple sensor modalities to enhance driving performance. By integrating data from RGB and depth cameras, their model learns a richer representation of the environment, capturing both visual and spatial information crucial to safe navigation. This multimodal fusion addresses the limitations of handling complex scenarios where reliance on a single modality might fail due to sensor noise or occlusions. The experimental results demonstrate that the multimodal approach significantly outperforms unimodal baselines, exhibiting improved robustness and generalisation. Xiao et al. [126] perform experimentation in a altered setting that test

with respect to modalities and report a 50% improvement in completing routes over the CILRS [34] method.

Extending the exploration of multi-modality for end-to-end trained driving models, Chitta et al. [31] introduced the TransFuser architecture. This model addresses limitations concerning the integration of multi-modal sensor data, along with generalising to unseen environments. The TransFuser employs transformer networks to fuse high-dimensional inputs from cameras and LiDAR sensors, effectively capturing both spatial and temporal dependencies in the data. This sensor fusion allows the model to generate more context-aware and robust driving policies within an imitation learning framework. By leveraging transformer-based architectures, the TransFuser demonstrates improved performance as such architectures are designed to be attention-oriented [120]. Chitta et al. [31] propose a benchmark standard of their own, i.e., Longest6 Benchmark and report a route completion score of 93.38% which surpasses their expert performance by 4.71%.

Hu et al. [60] propose another architectural advancement called ST-P3, an interpretable end-to-end vision-based framework for autonomous driving tasks that improves feature learning for perception, prediction, and planning. Given a sequence of surrounding camera video, ST-P3 first extracts features and lifts them to 3D space using depth prediction. Features in both spatial and temporal domains are fused by an egocentric-aligned accumulation scheme. For perception, this scheme aligns and aggregates features (past and present) in 3D space to preserve geometric information before Bird's Eye View (BEV) transformation. For prediction, a dual pathway scheme is introduced to account for past motion variations, thereby enhancing future prediction. This dual pathway modelling produces a more robust representation of the scene. For planning, ST-P3 incorporates prior knowledge from features in the early stage of the network and devises a refinement module to generate a final trajectory using high-level commands in the absence of HD maps. Hu et al. [60] evaluate over self-proposed benchmark, i.e., the Town05 Long benchmark which tests completion of routes in the Town05 map of the CARLA simulator [41]. ST-P3 reaches a route completion of 83.15%, tested against CILRS and Transfuser which are able to complete 56.36% and 7.19% of the routes respectively.

A very recent method, called ThinkTwice proposed by Jia et al. [66] introduces enhancing the decoder module of the neural network. It emphasises a coarse-to-fine refinement strategy where the decoder predicts a trajectory and action and subsequently refines this prediction using spatial-temporal priors and dense supervision. The architecture integrates three primary modules: the Look Module, which retrieves features from safety-critical areas; the Prediction Module, which anticipates future scenarios based on the ego vehicle's actions; and the Refinement Module, which adjusts the initial predictions using offset calculations. ThinkTwice employs a BEV representation derived from multi-sensor inputs, fusing camera and LiDAR data via a geometric transformation pipeline. The method demonstrates state-of-the-art performance on autonomous driving benchmarks, outperforming prior methods by leveraging a scalable decoder architecture and dense feature supervision. Jia et al. [66] evaluate on the Town05 Longest benchmark and they report route completion score of 77.2%, surpassing many other methods.

1.5.2. Data-Centric Advancements

As end-to-end learning primarily focuses on learning from data, research has consistently highlighted the critical role of high-quality data in improving the performance of learned methods [34, 104, 107]. In this subsection we highlight the research that explores data-centric approaches to improve the learning capabilities.

Since the introduction of the DAgger method and its wide-spread adoption across many methods in the field of automation and robotics there have been improved version of DAgger proposed. One of which that focuses specifically on end-to-end autonomous driving is called SafeDAgger [133]. The proposal of this novel extension to the DAgger algorithm aims to reduce the number of queries to a reference policy (or expert policy) during imitation learning. SafeDAgger introduces a safety policy that predicts when the primary policy is likely to deviate from the reference policy and switches control to the reference only in such cases. This significantly improves query efficiency by limiting interactions with the reference policy to critical moments, making the approach more

practical for applications where querying a human or expert system is costly. The method is evaluated in a racing simulator and demonstrates improved training efficiency and policy performance, highlighting its potential for safer and more effective end-to-end autonomous driving systems.

While SafeDAgger's proposal introduces reliance on prediction of errors, a method proposed by Prakash et al. [104] introduces having a replay buffer in the DAgger process. Their work called DARB [104] proposes an enhanced data aggregation framework for training end-to-end driving policies. Building on the DAgger algorithm, this work introduces two key modifications: a critical state sampling mechanism that prioritises scenarios with high utility for learning safe driving behaviours, such as near failures, and a replay buffer that balances expert data with on-policy data to mitigate bias and improve generalisation. Figure 1.8 shows how the modifications fit into the pipeline of DAgger. These innovations address limitations in standard DAgger, such as poor generalisation and over-fitting to training conditions.

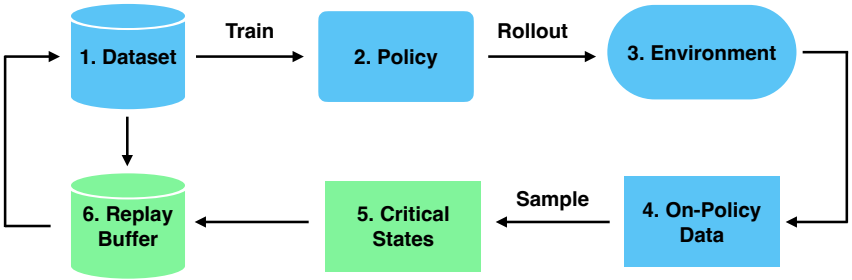


Figure 1.8: DARB [104] starts with an initial dataset, then extends the existing DAgger method by introducing critical states sampling and a replay buffer pipeline.

Other than improving how corrective data is aggregated, some works attempt to upgrade the quality of demonstrations. Research by Chen et al. [24] called Learning by Cheating (LBC), presents a two-stage training approach. The authors simplify the challenging task of training such systems by initially creating a "privileged agent" that has access to comprehensive environmental data, such as ground-truth layouts and the positions of all traffic participants. This agent is trained using

expert trajectories and acts as a teacher in the second stage, where a "sensorimotor agent" learns to operate using only visual input from a forward-facing camera. This decomposition of learning tasks allows the privileged agent to focus solely on action and later provide robust, adaptive supervision to the sensorimotor agent, which learns to interpret visual data. Experimental results show that this methodology achieves improved success rates on the benchmarks and also significantly reduces traffic violations and collisions compared to prior state-of-the-art models. Based on experiments conducted and reproduced by Chen et al. [24] LBC reaches a route completion score of 85% on the NoCrash benchmark, while CILRS reaches 67% in the dense traffic setting.

Heading in a similar direction as LBC [24], a method proposed by Zhang et al. [137] extends the idea of improving demonstrations. Instead of relying solely on privileged agents or human experts, this paper introduces Roach, a reinforcement learning (RL) expert designed to map BEV images to continuous driving actions. Unlike traditional approaches reliant on rule-based systems or expert demonstrations, Roach provides well-informed supervision signals. This agent enables to collect better demonstrations to further train imitation learning agents. Roach eliminates the need for human intervention by probing in not only demonstration generation phase but also while carrying out DAgger iterations. Without such dependence on human interventions, the Roach expert shows better scores than the state-of-the-art works publish around this time, highlighting the importance of quality in demonstrations. Zhang et al. [137] method establish and evaluated over the offline version of the Leaderboard benchmark and report a 78% route completion where their baseline only reaches 35%.

1.6. PRE-TRAINING

Pre-training refers to the process of training a model (e.g. neural-network) on a related task (or a source task) or dataset prior to fine-tuning it on the target task [53, 89, 139]. It is often used to initialise the model with weights that are already tailored to capture general patterns or features, which can then be adapted to the specific problem during the fine-tuning phase. Pre-training may be supervised, e.g., a CNN

trained on ImageNet can be fine-tuned on medical image classification [76] or self-supervised, e.g., a transformer being trained on a text corpus to predict the next token [89]. Figure 1.9 portrays the core concept of pre-training where weights learned from one task are transferred to another neural network to learn another task.

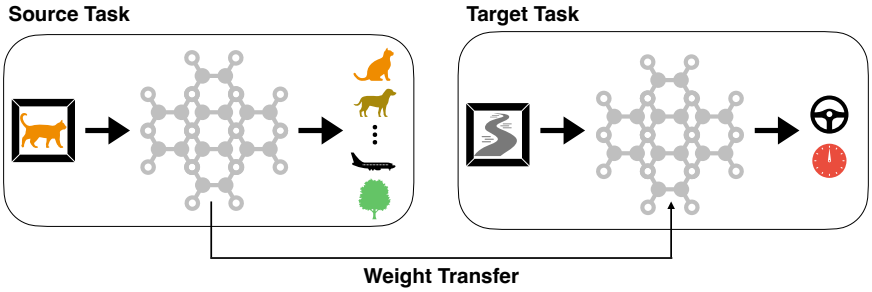


Figure 1.9: A learning method (e.g. neural network) can be pre-trained on a source task (left) and later it can be fine-tuned on a target task (right).

The practice of pre-training has become the cornerstone in the development of neural networks, particularly in domains like natural language processing (NLP), computer vision, and speech recognition. Advances in pre-training methodologies and the availability of large-scale datasets have significantly enhanced the performance of models across a variety of tasks. We give an overview of how pre-training has been adopted in the aforementioned domains:

1. **NLP:** Models such as BERT [39], GPT [120], T5 [106] and their variants have set new benchmarks by leveraging self-supervised pre-training on massive text corpora [89]. These models are fine-tuned for specific downstream tasks such as translation, summarisation, and sentiment analysis, demonstrating state-of-the-art performance.

2. **Computer Vision:** Pre-trained models such as ResNet [51], EfficientNet [114], and Vision Transformers (ViT) [40] trained on datasets like ImageNet are widely used for tasks ranging from object detection to medical imaging. Recent efforts in self-supervised

learning (e.g., SimCLR [27], BYOL [50]) are gaining traction enabling effective pre-training without labelled data [22].

3. **Speech Recognition:** Models like Wav2Vec [9] and Whisper [105] use self-supervised pre-training on raw audio data, significantly reducing the need for transcribed datasets and enabling robust performance on tasks like speech-to-text and speaker identification [115].

Research shows that scaling up the model size and pre-training data leads to significant performance improvements, as observed with GPT-3, PaLM, and similar large language models (LLMs) [89]. These advancements underline the importance of massive datasets and computational resources, with many leading developments being driven by large-scale AI research labs. The use of the existing or specially crafted pre-trained models as a base for fine-tuning has become standard. Fine-tuning allows these general-purpose models to adapt to domain-specific tasks with relatively small labelled datasets. Few-shot and zero-shot learning capabilities, enabled by large pre-trained models, are also gaining prominence, reducing the need for extensive task-specific data.

1.6.1. Pre-training in Autonomous Driving

While some may argue that pre-trained representations may not be sample efficient against learning a skill from scratch [110], ImageNet-based pre-training remains the most prevalent approach for initialising visual encoders in imitation learning-based for autonomous driving. The use of such approach leverages the extensive dataset and task diversity of ImageNet [38], enabling models to develop robust feature representations that are transferable across various domains. According to our survey, we observe that the trend of using ImageNet-based pre-training has remained consistent over recent years. We present this trend in the form of a table with the year of publication in Table 1.1. While the use of ImageNet pre-trained models could be beneficial, it may not be an appropriate fit for the task of driving as the target task highly differs from the source task.

Table 1.1: Research over the years solely relying on ImageNet-based pre-training.

Publication title	**Year of publication**
Exploring the Limitations of Behaviour Cloning for Autonomous Driving [34]	2019
Learning by Cheating [24]	2020
Exploring data aggregation in policy learning for vision-based urban autonomous driving [104]	2020
Learning Situational Driving [95]	2020
Multimodal End-to-End Autonomous Driving [126]	2020
End-to-End Urban Driving by Imitating a Reinforcement Learning Coach [137]	2021
ST-P3: End-to-End Vision-based Autonomous Driving via Spatial-Temporal Feature Learning [60]	2022
TransFuser: Imitation with Transformer-Based Sensor Fusion for Autonomous Driving [31]	2022
Scaling Vision-based End-to-End Driving with Multi-View Attention Learning [127]	2023
Think Twice before Driving: Towards Scalable Decoders for End-to-End Autonomous Driving [66]	2023
Exploring the Causality of End-to-End Autonomous Driving [78]	2024

Despite significant advancements in pre-training, the application of pre-training techniques remains relatively under-explored when it comes to end-to-end autonomous driving. This gap presents an opportunity to explore how pre-training can enhance imitation learning models. In this subsection, we highlight the handful of efforts that have been made using pre-training in and around our area of interest.

The recent work by Baker et al. [11] named Video Pre-Training (VPT) proposes a semi-supervised imitation learning approach that leverages

large-scale, unlabelled video datasets for pre-training in sequential decision-making domains. VPT demonstrates how a small amount of labelled data can train an inverse dynamics model (IDM), which generates pseudo-labels for vast amounts of unlabelled video. The labelled data and IDM predictions are used for behavioural cloning, allowing the model to learn a general behavioural prior without requiring extensive labelled datasets. The proposed method is trained for and tested in the Minecraft video game environment. The method bridges gaps in pre-training for imitation learning by making effective use of noisy, internet-scale datasets, offering a scalable and adaptable framework that could generalise to other domains with rich, unlabelled video data.

VPT [11] uses pre-training and imitation learning in the Minecraft environment, with some methods focus solely on the task of driving. One of which is proposed by Zhang et al. [135]. They label their method as Action-Conditioned Contrastive Policy Pre-training (ACO). The proposed method leverages a large corpus of unsaturated YouTube driving videos to learn action-relevant visual features. Similar to VPT [11], this method carries out training of an IDM with a small labelled dataset to generate pseudo-action labels for the video frames. These labels enable the creation of contrastive pairs conditioned on action similarity, allowing the model to focus on learning features critical for driving decisions rather than general visual representations. ACO demonstrates significant improvements in downstream tasks such as imitation learning within the CARLA simulator [41], outperforming traditional methods like ImageNet-based classification pre-training by approximately 30%. Incorporating real-world driving diversity into the pre-training process, ACO enhances the sample efficiency and generalisability of policies.

The second work that also leverages pre-training for the task of autonomous driving is called "Policy Pre-training for Autonomous Driving via Self-Supervised Geometric Modelling" (PPGeo) [125]. PPGeo introduces a novel self-supervised framework designed to address the challenges of pre-training in autonomous driving. Unlike conventional pre-training methods in vision tasks, which often focus on achieving view and translation invariance, PPGeo is tailored to learn driving policy representations by leveraging the unique requirements of driving scenarios. The framework operates in two stages: first, a geometric modelling

phase generates depth and ego-motion predictions from unlabelled driving videos; second, the visual encoder is trained to predict ego-motion from a single frame while optimizing photometric reconstruction errors. This approach enables the encoder to focus on driving policy-relevant information, such as traffic signals and immediate obstacles, rather than irrelevant scene details. Extensive evaluations across diverse driving tasks in the CARLA simulator demonstrate the superiority of PPGeo over baseline methods like ImageNet pre-training. Wu et al. [125] report that PPGeo completes 96.7% of the routes while using an ImageNet classification pre-trained model only completes 87.3% of the routes, which improves over the baseline by 10%. This framework not only enhances policy learning but also contributes to depth and odometry estimation tasks, showcasing its effectiveness in autonomous driving applications.

1.6.2. Potential Pre-training Paradigms

We propose the task of Visual Place Recognition (VPR) as one of two potential pre-training paradigms. As per Masone and Caputo [83], VPR is a task that enables systems to recognise previously visited locations using visual data. It involves extracting features from images to match scenes under varying conditions like lighting, weather, or viewpoints. VPR is widely used in autonomous vehicles for navigation and localisation, allowing them to recognise roads and landmarks for safe driving [128]. In robotics, it helps in mapping environments and enabling efficient path planning in dynamic settings [83]. It is also employed in geotagging photos and enhancing location-based services by recognizing and associating images with specific places [14]. These systems leverage cameras and computational models to process visual data, often using deep learning techniques to ensure robust performance across different conditions [83]. VPR has traditionally been integrated into robotics navigation through SLAM methods, aiding in map building and localisation. To our knowledge, VPR has not yet been combined with end-to-end trained autonomous driving methods. We hypothesise that bridging this gap could enhance navigation systems by leveraging VPR's robustness in dynamic environments and end-to-end methods' adaptability to complex scenarios. As per Masone and Caputo [83],

VPR has been approached in many ways from scale invariant feature extraction to using representation learning with neural networks. Most modern approaches have remained CNN centric forming descriptors to represent visual cues from images. Domain Adaptation for Semantic and Geometric-aware Image-based Localisation (DASGIL) [59], a recent approach follows this trend by forming global descriptors. While this approach brings in improvements over previously proposed approaches, it gets outperformed by down scaled descriptor size. SegVPR [99] shows improvement over DASGIL by 16.5% with a compact descriptor and certain other nuances. SegVPR states that a smaller and aggregated descriptor size retains much more useful information in a descriptor.

Self-supervised learning has emerged as a powerful paradigm for pre-training models without requiring labelled data, making it highly scalable and cost-effective. Methods such as self-distillation with no labels (DINO) [22], MoCo [29], BYOL [50], SimSiam [28], and SwAV [21] all leverage different strategies to learn meaningful representations from unlabelled data. DINO, for instance, uses a teacher-student framework enabling it to capture rich visual features and achieves an improved performance on downstream tasks. Similarly, MoCo adapts the contrastive learning framework, emphasising stable training dynamics. In contrast, BYOL and SimSiam eliminate the need for negative samples with BYOL using a momentum encoder and SimSiam relying solely on a Siamese network. SwAV, on the other hand, introduces clustering-based contrastive learning, which reduces computational overhead while maintaining strong performance. The key benefit of self-supervised learning lies in its ability to pre-train models on vast amounts of unlabelled data, which is particularly advantageous in domains where labelled data is scarce or expensive to obtain. By learning generalizable features during pre-training, self-supervised methods enable models to perform well on downstream tasks with minimal fine-tuning. For example, DINO demonstrates that it can uncover interpretable features, such as object boundaries and semantic parts, which are crucial for tasks like image classification and segmentation. Therefore, we propose self-supervised pre-training as another potential paradigm in this research, particularly, using the methodology of DINO [22] as pre-training in autonomous driving.

1.7. AUTONOMOUS DRIVING RESEARCH IN LITHUANIA

In recent years, Lithuanian researchers has been actively engaged in autonomous driving research with a focus on reinforcement learning, sensor technologies, and vehicular communication systems. Petryshyn et al. [102] use the AWS DeepRacer platform to train self-driving models, exploring various reward functions and sensor configurations to improve obstacle avoidance and track completion. These efforts have demonstrated that a continuous reward function, which provides more fine-grained feedback to the agent, significantly enhances the learning process. Additionally, incorporating LiDAR sensors alongside cameras has proven crucial for improving an agent's environmental awareness and navigation capabilities, leading to a 95.8% reduction in collision rates and a 79% decrease in trial circuit completion time in one study. Another study from Lithuania [109] explores the use of the Unity ML-Agents toolkit to train kart agents to navigate a racing track in a simulated environment using RL algorithms. The research compares the performance of several different RL algorithms and configurations on the task of training kart agents to successfully traverse a racing track, identifying the most effective approach for navigating the track and avoiding obstacles. The study utilises the Unity game engine as the simulation environment, with 24 agents (karts) being trained independently to speed up the learning process. Whereas another research form Lithuania focuses on Vehicles-to-Everything communication [119], which involves integrating various sensors, communication standards, and machine learning methods to enable communication between vehicles, infrastructure, pedestrians, and other traffic elements. This includes vehicle-to-vehicle communication for real-time data exchange, vehicle-to-infrastructure communication for interaction with road infrastructure, vehicle-to-pedestrian communication to enhance safety, vehicle-to-device communication, vehicle-to-network communication, and vehicle-to-cloud communication. This work provides a comprehensive overview of the technologies and data involved in Vehicles-to-Everything communication, highlighting the importance of this field for the development of autonomous driving. The work also points out the gaps, challenges, and future research directions, emphasising the need for reliable, efficient, and secure communication in autonomous

vehicle networks. It is noteworthy that in the private sector, Neurotechnology[1] conducted research and development of various vision-guided robotic systems since the early 2000s. Taking inspiration from neural networks, perception-action loop, and work of Gaussier and Zrehen [49] and others, Neurotechnology's researchers developed various incarnations of vision-guided mobile manipulators, which mostly are not documented in scientific literature. These research efforts are closely related to autonomous driving since they include autonomous navigation behaviours. In this line of research, Daniušis et al. [36] [C.1] proposed a topological navigation graph framework as a way to create goal-directed navigation systems from non-goal-directed, imitation-learning components for robust trajectory imitation. The TNG framework represents the environment as a directed graph composed of deep neural networks. This work also provides an empirical evaluation of the TNG framework in both simulated and real-world environments such as a shopping area space and an office environment space.

1.8. CONCLUSIONS OF CHAPTER 1

This section reviews the literature that forms the background of end-to-end driving research and pre-training for end-to-end autonomous driving. We draw the following conclusions from the presented literature review:

- Autonomous driving may take one of two approaches, modular or end-to-end. While the modular approach assigns singular tasks to each module of a system and is heavy on the engineering side, the end-to-end approach learns to drive in a holistic and data-driven way through demonstrations. Based on the advantages of the end-to-end approach, this research deviates towards them.
- The end-to-end approach to autonomous driving could be learned using imitation learning or RL. Imitation learning is the simplistic approach that stems from supervised learning whereas RL learns the policy through interaction with the environment which makes it slower and resource heavy.

[1] http://www.neurotechnology.com

- Imitation learning suffers from the problem of covariate shift, where the difference in data distribution at train time and test time causes cascading errors.

- Early methods show the use of neural networks and DAgger them forms the basis of current state of autonomous driving that relies on imitation learning. Later methods work on various aspects, such as using a conditional architecture, introducing and improving multi-modality, enhancing demonstration quality, and more. These directions directly or indirectly aim to solve covariate shift and other related issues.

- Pre-training has become a standard in many areas of deep learning research and applications. There is a strong trend in relying on ImageNet-based pre-training for autonomous driving, as per research that has been proposed in recent years.

- While the ImageNet-based pre-training is useful it may also be suboptimal. Only a handful of methods have explored variability in pre-training methods for the case of autonomous driving. Therefore we point out that there seems to be a gap in research in this upcoming area. Following that, we mention potential pre-training paradigms that can append the value when applied to autonomous driving agents.

2. RESEARCH METHODOLOGY

This chapter describes the following proposed methods:

1. Visual Place Recognition Pre-training for Autonomous Driving [A.1][68]
2. DINO Pre-training for Autonomous Driving [A.2][69]

These methods are implemented and evaluated, and compared against baseline methods, which are also established in this chapter. Additionally, this chapter also reveals the experiment design for the evaluation of the proposed methods against baseline methods.

2.1. VISUAL PLACE RECOGNITION PRE-TRAINING

This method is proposed to address the issue of covariate shift in imitation learning for autonomous driving, specifically focusing on weather and lighting variations. We hypothesise that autonomous driving relies heavily on specific visual features that might not be effectively captured by ImageNet pre-training, which is based on image classification, a task distant from driving. Therefore, we propose pre-training the visual encoder of an agent on VPR because VPR datasets [83] inherently incorporate weather and lighting variations to achieve place retrieval under changing conditions. By transferring an encoder that is pre-trained in such a way to the driving agent, the method aims to improve the agent's ability to adapt to unseen weather and lighting conditions and mitigate the effects of covariate shift. As covariate shift is one of the major problems in imitation learning-based methods (see Section 1.3.1.1).

Generally, an imitation learning-based autonomous driving method is portrayed as an agent, that consists of a neural network architecture (e.g. Figure 1.7). We term the method proposed in this section as the VPR pre-training method and the outcome of the method is termed as the VPR pre-trained agent. The formation of the VPR pre-trained agent is described in two parts. At first, the proposed pre-training of the visual

encoder is described, in Section 2.1.1. Following pre-training, incorporation of the pre-trained visual encoder into an agent and training over the task of autonomous driving is described, in Section 2.1.2.

2.1.1. Pre-training of the Visual Encoder using VPR

VPR is a fundamental computer vision task that aims to identify and recognise previously visited locations using only visual information from images or video sequences. The primary objective of VPR is to determine whether a given query image corresponds to a location that exists within a reference database of images, effectively answering the question "Have I been here before?". VPR based systems are trained against challenging real-world conditions such as variations in lighting conditions (day vs. night), seasonal changes (summer vs. winter), different weather conditions, varying viewpoints, and dynamic objects in the scene. Through exposure to these variations during training, neural networks learn to develop feature representations that remain robust across environmental changes.

The VPR pre-training method that is proposed extends upon a recent VPR method, SegVPR [99], for pre-training the visual encoder of the autonomous driving agent. SegVPR performs training using a single ImageNet pre-trained ResNet50-based [51] visual encoder. During training, this encoder is shared over two tasks:

1. Visual place recognition (VPR) task: this task serves the main purpose;
2. Semantic segmentation (SemSeg) task: this task serves an auxiliary purpose.

By sharing the visual encoder, the two tasks enable learning features that combine place-specific information and semantic information from the scene. The combination of the two tasks helps in guiding the overall neural network towards which area of the scene to look at. This is further assisted by the use of multi-scale attention and pooling mechanisms in the decoder that is visualised in Figure 2.1.

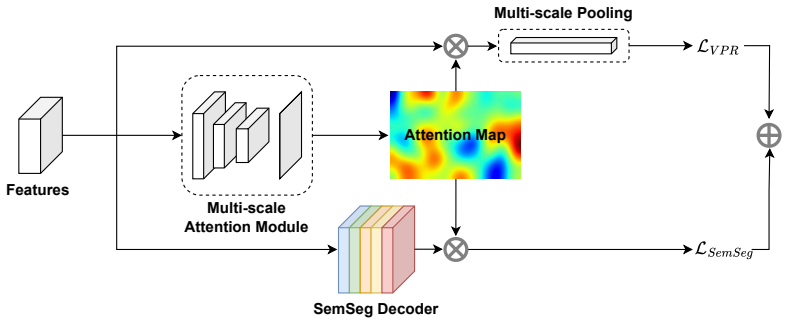


Figure 2.1: The figure illustrates the SegVPR decoder structure consisting of a segmentation decoder, multi-scale attention module and pooling module.

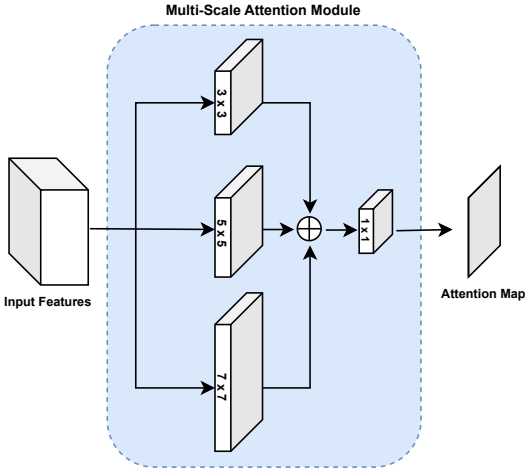


Figure 2.2: The figure illustrates the multi-scale attention module used in SegVPR that employs multiple spatial scales to capture objects of different sizes, and produces an attention map.

A multi-scale attention module is used to focus on the salient regions of the input image, and additionally to guide the semantic segmentation during training. In general, a multi-scale attention module assesses features at different resolutions, assigns weights as per their importance then fuses features to a single representation. This module takes the output of the encoder from the fourth convolutional layer as input,

and passes through a filter of kernel sizes 3, 5, and 7, shown in Figure 2.2. After upsampling and channel-wise concatenation of the outputs of the module, an attention map is formed. The scores held in the attention map indicate where the focus is allotted. SegVPR uses multi-scale pooling to extract semantic and appearance information at different semantic information levels. This mechanism uses information from the fourth and fifth convolution layers of the encoder. These features are weighted with the attention map scores to produce a global descriptor (synonymous with representation).

For training to learn extraction of descriptors for the VPR task, a weakly supervised triplet margin loss is used [7]. Therefore, given an input image to the SegVPR architecture, its multi-scale pooling module produces a descriptor (as in Figure 2.1), which we represent as F. As per the classic triplet training method query, positive and negative descriptors samples are drawn from a gallery of samples. The positive and query samples belong to nearby GPS coordinates while negative samples are sampled from a distant location. For every sample triplet, the VPR loss is given by

$$\mathcal{L}_{VPR} = h(d(F_{query}, F_{pos}) + m - d(F_{query}, F_{neg})), \tag{2.1}$$

where h is the Hinge loss $h(x) = max(x, 0)$, d is the Euclidean distance, $m > 0$ is a fixed margin, and F_{query}, F_{pos}, and F_{neg} represent query, positive, and negative triplet samples respectively.

The semantic segmentation loss $\mathcal{L}_{SemSeg}$ is given by the formula,

$$\mathcal{L}_{SemSeg} = -\frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} y_i \cdot \log p_i^{y_i}(M^i \cdot f_d^i), \tag{2.2}$$

which is equal to a cross-entropy loss that is computed for each class y_i at pixel i from the image $\mathcal{I}$, where M_i is an attention map related to the feature f_d^i and p_i denotes the probability of class y_i. Both f_d^i and p_i are outputs of the segmentation decoder module, while M_i is the output of the multi-scale attention module.

The overall loss function is a sum of VPR loss and semantic segmen-

tation loss [99]:

$$\mathcal{L}_{VPR-SemSeg} = \mathcal{L}_{VPR} + \alpha \cdot \mathcal{L}_{SemSeg}, \tag{2.3}$$

where $\alpha > 0$ is a scalar weight for semantic segmentation loss.

SegVPR uses a specially built dataset captured in the CARLA 0.9.10 simulator. The dataset includes GPS information and pixel-wise semantic annotation with 25 semantic classes. It captures more than $40,000$ images (10091 per scenario), collected across Town03 and Town10 maps with varying weather from Clear Noon and Hard Rain Sunset.

2.1.2. VPR Pre-trained Agent Training

To exploit the robust visual representations learned across changing weather conditions presented during pre-training, we integrate the visual encoder from the SegVPR architecture into our agent's neural network framework. In this subsection we elaborate on the training and architecture of the VPR pre-trained agent that embeds the pre-trained visual encoder. The architecture can be seen in Figure 2.3.

Our agent's architecture builds on CILRS [34], where the neural network is conditioned on high-level navigational commands. These commands, generated by a route planner (provided by simulation software) based on target destinations, guide the agent's decision-making process. For collecting initial driving demonstrations, we utilise the automated framework based on a RL expert proposed in Roach [137], which form the training and validation datasets. This methodology not only eliminates the need for human operators but also ensures the collection of consistent and high-quality data. Following the pre-training phase, we train our proposed agent. We then implement the DAgger process [107], where our initially trained agent actively generates driving behaviours while being supervised by the Roach agent. When discrepancies arise between the actions proposed by our agent and those of the Roach agent, these instances are recorded and aggregated into the training dataset. This aggregation of corrective demonstrations follows the methodology established in the original DAgger algorithm [107], enabling our agent to learn from the expert's corrective actions.

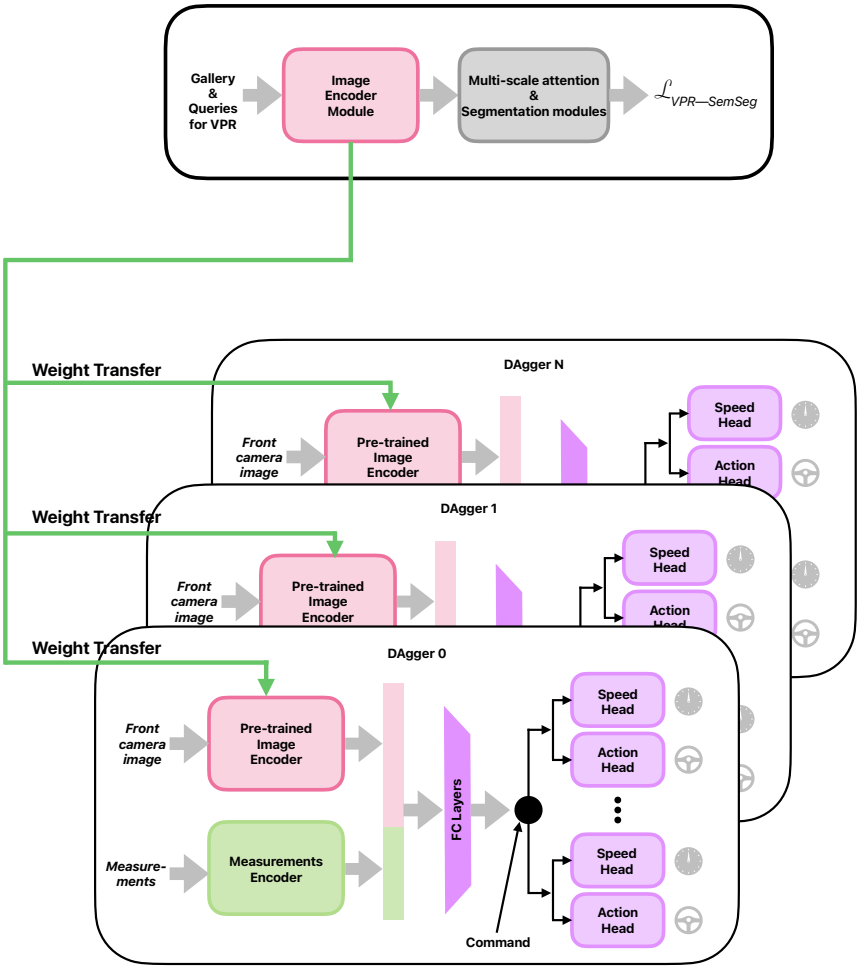


Figure 2.3: The figure illustrates the overall block diagram of the proposed visual place recognition pre-training method, where at first, an image encoder is pre-trained on the VPR task (top) followed by weight transfer to train for the task of end-to-end driving (bottom).

The proposed VPR pre-trained agent's architecture (Figure 2.3) comprises two parallel encoding streams: a measurements encoder that processes the current speed and one-hot encoded high-level commands, and the SegVPR encoder that processes visual input. The outputs of both encoders are concatenated and processed through a joint module consisting of fully connected layers, which reduces the dimensionality of the combined features. This joint representation is then fed into spe-

cialised action branches, where each branch corresponds to a specific high-level command, following the branched architecture established in CILRS and Roach. During execution the branch corresponding to the current high-level command generates the low-level driving commands while during training non-active branches are masked to zero.

Let $X \in \mathbb{R}^{224\times224\times3}$ be an input image from the front camera sensor. The agent maps X onto an action in $\mathbb{R}^2$ vector that consists of a throttle and steering value for the vehicle. Therefore, the agent is represented by the following equation:

$$\widehat{\mathbf{a}}(X, u|\theta, \xi, \phi, \psi) := \sum_{i=0}^{n} c_i \phi_i(X, u|\theta, \xi, \phi, \psi), \tag{2.4}$$

where $\phi_i(X, u|\theta, \xi, \phi, \psi)$ corresponds to the output of i^{th} action branch of $f_A(f_J(f_E(X|\theta), f_M(u|\xi)|\phi)|\psi)$ out of all n action branches. Where,

- X is the input image,
- f_E is the image encoder with parameters θ pre-trained on the VPR task (i.e., SegVPR encoder),
- f_M is the measurements encoder network with parameters ξ, while u is a vector holding measurements (current speed and high-level command),
- f_J is another neural network module with parameters ϕ that concatenates the image and measurements encodings and downsizes it,
- f_A is the actions branches module with parameters ψ which calculates a low-level command for each high-level command,
- c_i represents a vector that keeps one of n action branches that is intended for the input image X and nullifies all other action branches.

Based on previous works [33, 34, 137], we zero-index the action branches.

To simplify the comparison with a baseline and following the approach taken by other works [137], we use the loss function as the sum

of action loss and a speed prediction regularisation,

$$\mathcal{L}_{Agent}(\theta, \xi, \phi, \psi) = \mathcal{L}_A(\theta, \xi, \phi, \psi) + \lambda_S \cdot \mathcal{L}_S, \tag{2.5}$$

where the action loss $\mathcal{L}_A$ is equal to L1 loss between expert action $\hat{\mathbf{a}}$ and predicted action $\mathbf{a}$, given by

$$\mathcal{L}_A = \|\hat{\mathbf{a}} - \mathbf{a}(X, u|\theta, \xi, \phi, \psi)\|_1, \tag{2.6}$$

and the speed prediction regularisation $\mathcal{L}_S$ between measured speed $\hat{s}$ and predicted speed s is given by

$$\mathcal{L}_S = |\hat{s} - s|. \tag{2.7}$$

The regularisation effect is regulated with a scalar value λ_s taken as `1e-5`.

We reveal neural network hyper-parameter choices with the rest of the implementation details in Section 2.3.6. We also formulate the entire process of training the proposed VPR pre-trained agent for driving in the form of an algorithm in Algorithm 2.

Algorithm 2 The VPR pre-trained driving agent

Input: Initial dataset D collected using the Roach agent, trained SegVPR encoder f_E.
Output: trained $agent$

1: **for** DAgger iteration `i = 0 to 5` **do**
2: Initialise agent architecture $agent_i$.
3: Initialise $agent_i$'s image encoder with f_E.
4: Train $agent_i$ on D.
5: Collect dataset $D_i = (X, u, \pi^*(X, u))$, where X and u are input image and measurements (speed and high-level command), correspondingly, and π^* is the supervising Roach agent's output, measured in situations when there is a disagreement between the predictions of Roach and $agent_i$.
6: Aggregate dataset: $D \leftarrow D \cup D_i$.
7: **end for**
8: Return best $agent_{i^*}$ as per performance on the Leaderboard benchmark.

2.2. DINO PRE-TRAINING

The majority of the methods [24, 31, 34, 60, 66, 95, 126, 127, 137] [A.1] rely upon supervised learning for pre-training. Supervised learning relies heavily on labelled data, where each data point is associated with a specific label or category. This approach can be expensive to scale due to the need for manual annotation. In contrast, self-supervised learning leverages unlabelled data and generates artificial supervision signals from the data itself. We propose the use of self-distillation with no labels (DINO) method as a pre-training method to improve an agent's ability to adapt to new, unseen situations when encountering covariate shift. Similarly to the VPR pre-trained agent, we propose pre-training of an agent's visual encoder using DINO as a pre-training method. We hypothesise that the heavy use of labels in supervised pre-training, like ImageNet classification, limits the model's ability to learn a wide array of features and thus generalise to new situations. DINO, a self-supervised learning method was recently proposed, which has shown the ability to learn richer and more diverse features without relying on specific labels. DINO employs multi-crop training and a contrastive loss to learn inherent semantic information from images without explicit labels, demonstrating the effectiveness of self-supervised learning in capturing a broader understanding of data. DINO has also shown an inherent understanding of the semantic information within an image, which is useful for various computer vision tasks, including autonomous driving.

Following the VPR pre-trained agent's [A.1] structure as proposed in Section 2.1, we structure the current proposal's architecture similarly. We term the method proposed in this section as the DINO pre-training method and the outcome of the method is termed as the DINO pre-trained agent. The formation of the DINO pre-trained agent is described in the following two parts. At first, the proposed pre-training of a visual encoder, i.e., DINO pre-training in the context of our use, is described in Section 3.4. Following pre-training, the method of incorporating the pre-trained visual encoder into an agent and training over the task of autonomous driving is described in Section 2.2.2. The whole of DINO pre-training along with training of the agent process is illustrated in Figure 2.4.

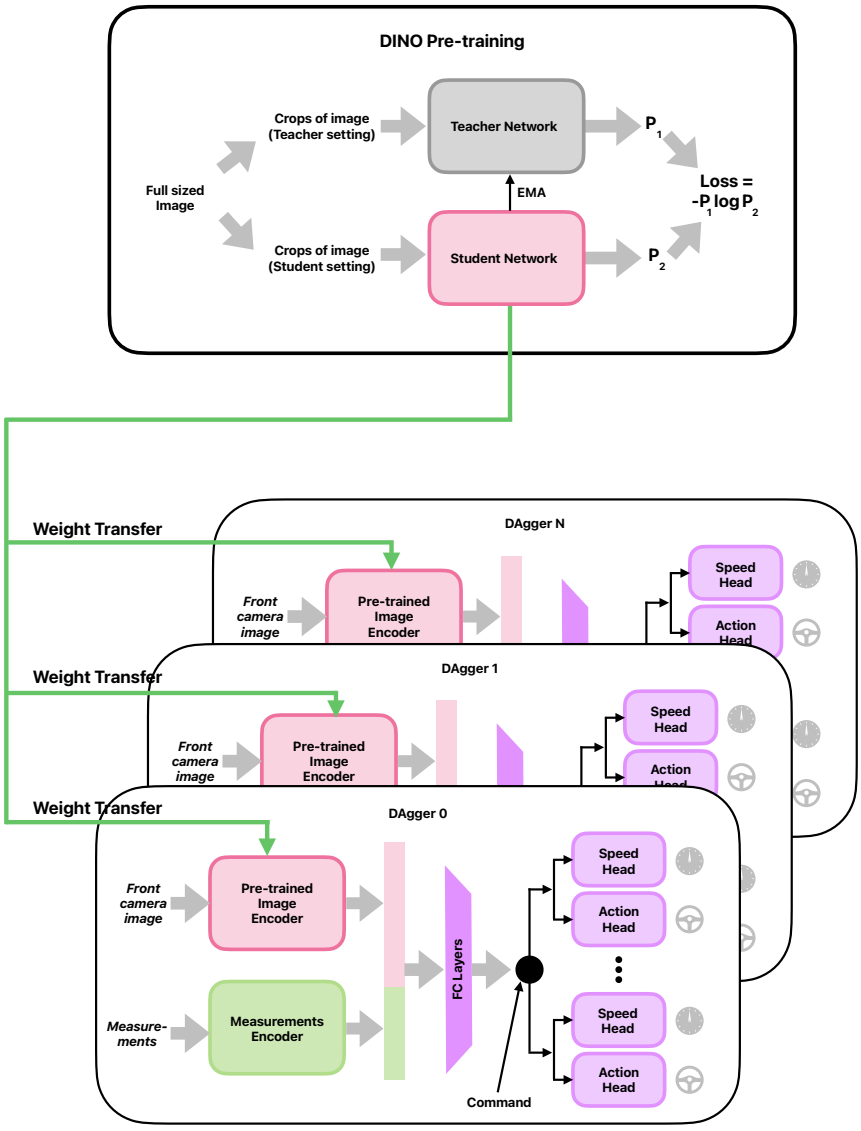


Figure 2.4: The figure illustrates the overall block diagram of the DINO pre-training method (top), using a teacher-student architecture and exponential moving average (EMA) to update the teacher network weights from student network. Teacher and student are trained on crops of the original full size image. Later illustrating weight transfer to train for the task of end-to-end driving (bottom).

Self-supervised training maximises the utility of existing data through innovative training paradigms. While traditional supervised learning focuses on direct task-specific training, self-supervised approaches may instead optimise for auxiliary tasks that indirectly benefit the target objective [125]. The effectiveness of this approach scales with dataset size, i.e., larger pre-training datasets when combined with appropriate self-supervised paradigms, generally tend to perform better. Following this principle, we employ DINO as a pre-training method. DINO follows a self-supervised learning framework that trains on the ImageNet [38] dataset containing approximately 1 million images. Rather than using conventional supervised classification, DINO leverages two key techniques: multi-crop training and self-distillation.

Like other knowledge distillation methods [25], DINO employs twin networks i.e., student and teacher networks, with identical parameter counts. The student network g_{θ_s} with parameters θ_s is trained to emulate the outputs of its teacher counterpart g_{θ_t} with parameters θ_t. When presented with an input x, both networks generate K-dimensional probability distributions, denoted as P_s and P_t respectively. These distributions are then processed through a modified softmax function, where a temperature parameter controls the distribution sharpness. For the student network, the probability P_s is calculated using temperature parameter τ_s as shown in equation 2.8:

$$P_s(x)^{(i)} = \frac{\exp\left(g_{\theta_s}(x)^{(i)}/\tau_s\right)}{\sum_{k=1}^{K}\exp\left(g_{\theta_s}(x)^{(k)}/\tau_s\right)}. \tag{2.8}$$

Similarly for the teacher network, the probability P_t is calculated using temperature parameter τ_t as shown in equation 2.9:

$$P_t(x)^{(i)} = \frac{\exp\left(g_{\theta_t}(x)^{(i)}/\tau_t\right)}{\sum_{k=1}^{K}\exp\left(g_{\theta_t}(x)^{(k)}/\tau_t\right)}. \tag{2.9}$$

The temperature control parameters are conditioned $\tau_s > 0$, $\tau_t > 0$, and initially set to 0.1 and 0.04, respectively.

The teacher network is co-trained along with the student network, but

is frozen during an epoch. Instead, the exponential moving average of the weights is copied from the student network to the teacher network, using the momentum encoder technique [54]. The update rule used during training is:

$$\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s, \tag{2.10}$$

where λ follows a cosine schedule from 0.996 to 1 during training. With the use of a fixed teacher network within an epoch, the learning takes place by minimising cross-entropy w.r.t. the student network parameters θ_s, as in the following equation:

$$\min_{\theta_s} H(P_t(x), P_s(x)), \tag{2.11}$$

where $H(P_t, P_s) = -P_t \log P_s$.

To leverage the self-supervision, DINO uses multi-crop augmentation training [20]. At first, a set of multiple views or crops V of an image are formed in two settings. The first setting creates two views called global views x_1^g and x_2^g, which are crops at a resolution of 224×224 that cover more than 50% of the image. The second setting creates several views called local views which are of a resolution 96×96 that cover less than 50% of the image. Once the views are created, the global views are passed through the teacher network, and all views including global and local views are passed through the student network. Thereafter, a modified version of the loss function mentioned in eq. 2.11 is used to adapt to a self-supervised setting in the following way:

$$\min_{\theta_s} \sum_{x \in \{x_1^{g1}, x_2^{g2}\}} \sum_{x' \in V, x' \neq x} H(P_t(x), P_s(x')). \tag{2.12}$$

We additionally formulate this pre-training procedure in Algorithm 3.

The work presented in the DINO research demonstrates the effectiveness of the approach on both convolutional neural networks and transformer architectures [22]. For our implementation and experiments, instead of training a DINO model from scratch, we use their convolutional neural network variant based on ResNet50. Choosing this architecture facilitates direct comparisons with our previous work, VPR pre-trained agent [A.1], and the many other methods that we base our

baseline agents upon.

Algorithm 3 DINO Pre-training Procedure for each Epoch

Input: Student network g_{θ_s}, teacher network g_{θ_t}
Output: Trained student and teacher networks

Initialize teacher parameters: $\theta_t \leftarrow \theta_s$
for each mini-batch x from the data loader **do**
    Generate random views using multi-crop augmentation: $x_1, x_2 \leftarrow \text{augment}(x), \text{augment}(x)$
    Compute student outputs: $s_1, s_2 \leftarrow g_{\theta_s}(x_1), g_{\theta_s}(x_2)$
    Compute teacher outputs: $t_1, t_2 \leftarrow g_{\theta_t}(x_1), g_{\theta_t}(x_2)$
    Compute loss: loss $\leftarrow \frac{1}{2}\left(H(t_1, s_2) + H(t_2, s_1)\right)$
    Update student network g_{θ_s} via Stochastic Gradient Descent
    Update teacher parameters: $\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s$
end for
return Trained networks

2.2.2. DINO Agent Training

The DINO driving agent proposed by us follows the framework established in the previously proposed method [A.1], as detailed in Section 2.1.2. Like the VPR pre-trained agent, our architecture employs a CILRS-based decoder [34]—a widely adopted approach in autonomous driving literature [137] [A.1] where the navigation system's high-level commands activate specific decoder branches. We begin by collecting initial demonstration data using Roach [137]. This initial dataset is used to train our agent, which incorporates a pre-trained encoder. The trained agent is then deployed in a simulated environment under training conditions, with the Roach agent providing supervision. When our agent's decisions deviate from the Roach agent's optimal actions, the Roach agent intervenes with corrections, and these corrective demonstrations are preserved for subsequent DAgger iterations. Each iteration combines the original dataset with these corrected demonstrations for retraining. Following established benchmarks [58, 137], we perform this data aggregation and retraining cycle five times.

The agent's architecture follows the same design as our VPR pre-trained agent (detailed in Section 2.1.2). Briefly, it comprises a pre-

trained vision encoder for front-view RGB images, a measurements encoder for vehicle speed and one-hot encoded high-level commands [34, 137] [A.1], and command-specific action branches that output low-level driving commands through a fully-connected join module.

Mathematically representing the agent similar to the VPR pre-training method, let $X \in \mathbb{R}^{224\times224\times3}$ be the front-view image from the vehicle. Similarly to the VPR pre-training method in section 2.1.2, we formulate the network representation as $f_A(f_J(f_E(X|\theta), f_M(u|\xi)|\phi)|\psi)$. The distinction made for the DINO pre-trained agent is that f_E is an image encoder with parameters θ pre-trained using the DINO method. The command selection mechanism remains identical, yielding the action prediction using the equation 2.4.

We maintain the training equations for the driving agent by using the same loss function defined in equation 2.5. The loss function sums action loss (equation 2.6) with speed loss (equation 2.7) that is regulated by a scalar value λ_s. We reveal neural network hyper-parameter choices with rest of the implementation details in Section 2.3.6. We also formulate this method in the form of an algorithm in Algorithm 4.

Algorithm 4 The DINO pre-trained driving agent

Input: Initial dataset D collected using the Roach agent, trained DINO encoder f_E.
Output: trained $agent$

1: **for** DAgger iteration `i = 0 to 5` **do**
2: Initialise agent architecture $agent_i$.
3: Initialise $agent_i$'s image encoder with f_E.
4: Train $agent_i$ on D.
5: Collect dataset $D_i = (X, u, \pi^*(X, u))$, where X and u are input image and measurements (speed and high-level command), correspondingly, and π^* is the supervising Roach agent's output, measured in situations when there is a disagreement between the predictions of Roach and $agent_i$.
6: Aggregate dataset: $D \leftarrow D \cup D_i$.
7: **end for**
8: Return best $agent_{i^*}$ as per performance on the Leaderboard benchmark.

2.3. EXPERIMENTAL SETUP

The development and validation of autonomous driving systems present unprecedented challenges in terms of safety demonstration. Research by Kalra and Paddock [70] highlights a critical issue: to prove their reliability in preventing fatalities and injuries, autonomous vehicles would need to be driven for hundreds of millions, and in some cases, billions of miles. This presents an insurmountable obstacle for traditional testing methods. Even under aggressive testing scenarios, it would take existing fleets tens to hundreds of years to accumulate the necessary mileage – a time frame that is entirely impractical if the goal is to verify their safety before public release. The aforementioned research underlines a fundamental problem: conventional road testing alone cannot provide sufficient evidence to demonstrate the safety of autonomous vehicles.

Given these limitations, research [70] emphasises the strong need for innovative approaches to safety validation. Developers and researchers must explore novel methods to demonstrate the reliability and safety of autonomous systems. However, the analysis also acknowledges that even with advanced testing methodologies, it may be impossible to establish the absolute safety of autonomous vehicles with complete certainty. Some degree of uncertainty will persist. Therefore, as traditional testing methods prove insufficient, research and development have turned to alternative approaches, including sophisticated driving simulators and standardised benchmarks, to supplement real-world testing and accelerate the safety validation process.

2.3.1. Simulation Environment

For the simulation environment we choose the CARLA simulator, version 0.9.11. CARLA [41] is a comprehensive open-source simulator that excels in various aspects of autonomous driving research. One of its strengths is the ability to synthesise high-fidelity sensory data from multiple sensors, including RGB cameras, depth cameras, LiDAR, and radar. This allows for the efficient collection of data, especially in adverse weather conditions that are difficult and costly to replicate in

Figure 2.5: The image shows the CARLA simulator's capability of simulating a real-world environment with traffic, pedestrians, environmental elements, and weather conditions.

real-world testing. As an open-source platform, CARLA fosters collaboration and reduces entry barriers for individual researchers and institutions, unlike commercial simulators that often require substantial financial investment.

According to an exhaustive comparison research [80], CARLA is particularly well-suited for end-to-end driving policy design due to its ability to simulate the entire autonomous driving pipeline. Its comprehensive nature integrates functionalities found in other simulator types, encompassing traffic flow simulation, sensory data generation, driving policy evaluation, and vehicle dynamics simulation, making it ideal for testing and validating complete autonomous driving systems. Researchers can evaluate the performance of driving policies in a safe and controlled environment, enabling the exploration of complex scenarios and edge cases without real-world consequences. While CARLA's vehicle dynamics fidelity may be less accurate compared to specialised commercial simulators, its ability to simulate a complete driving environment, coupled with its open-source accessibility, makes it a powerful choice for end-to-end driving policy design and research. This is also confirmed with majority of the published research [33, 34, 127, 137] re-

lated to end-to-end autonomous driving to be using the CARLA simulator. Figure 2.5 shows an example landscape from the CARLA simulator.

2.3.2. Benchmark Standard

The CARLA autonomous driving Leaderboard benchmark is a challenge designed to assess the performance of autonomous driving agents in realistic simulated driving scenarios. The primary objective of this challenge is to evaluate how well autonomous agents can navigate a series of pre-defined routes within the CARLA simulator. These routes are designed to test various aspects of autonomous driving, including navigating freeways, urban environments, residential areas, and rural settings. The challenge also incorporates different weather conditions, such as daylight, sunset, rain, clear sky, and more. Participating agents are tasked with driving from a designated starting point to a destination point on each route. They receive route information in the form of GPS coordinates, map coordinates, or high-level instructions. The challenge is designed to be fair and reproducible, allowing for a standardised evaluation of different autonomous driving approaches and facilitating direct comparisons between them.

2.3.3. Experimentation Design

The core task of an agent being evaluated is to complete a set of pre-defined routes. At each time step, the agent receives observations from the front camera and commands from a high-level planner to guide it toward its destination. Based on these inputs, the agent generates throttle and steering angle commands to navigate the vehicle. Each route is defined by a starting point and an ending point, represented by GPS coordinates. Between these points lies a sequence of waypoints that define the path. The route terminates if the agent falls out of the path suggested by the route planned and also when an agent gets stuck for more than 30 seconds. The high-level planner generates commands based on the agent's current position and the GPS coordinates of the nearest waypoint. The planner's commands can take one of six possible values: left, right, straight, lane to follow, change lane left, and change

Table 2.1: Distribution of towns for training, evaluation and testing, following the benchmark standard [137].

Training towns	**Evaluation towns**	**Testing towns**
Town 1	Town 1	Town 2
Town 3	Town 3	Town 5
Town 4 - train routes	Town 4 - train routes	Town - 4 test routes
Town 6	Town 6	

lane right. These commands are updated whenever the agent reaches a new waypoint.

The benchmark utilises CARLA's town environments and weather settings in its evaluation protocol. Both environments and weather conditions are divided into distinct training and testing sets. The towns follow a geographical split, ensuring that models must generalise to entirely new urban layouts and road networks not encountered during training. For weather conditions, the benchmark incorporates diverse training scenarios including clear noon, wet noon, clear sunset, and hard rain noon, while reserving wet sunset and soft rain sunset exclusively for testing. Examples of these weather conditions are illustrated in Figure 2.6. The benchmark evaluation is conducted using two sets: an evaluation set comprising conditions from the training set, and a testing set featuring previously unexposed conditions. The distribution of towns and weather conditions across these sets is detailed in Table 2.1 and Table 2.2, respectively. To evaluate the agent's capability to navigate through traffic, the benchmark standard operates under a busy traffic density mode. The evaluation is carried out three times by changing random seed values and the average of the scores are calculated.

2.3.4. Data Collection

During the data collection phase, we follow the methodology established by Roach [137], which provides a robust approach to gathering high-quality driving data in the CARLA simulator. This method em-

(a) Wet noon

(b) Clear sunset

(c) Soft rain sunset

(d) Wet sunset

Figure 2.6: The figure shows two weather conditions for evaluation (a) and (b), that are used as a part of evaluation set to test in known conditions, followed by weather conditions (c) and (d), that are unseen by the agent and used in testing.

Table 2.2: Distribution of weather conditions for training, evaluation and testing, following the benchmark standard [137].

Training weather	**Evaluation weather**	**Testing weather**
Wet noon	Wet noon	Wet sunset
Clear sunset	Clear sunset	Soft rain sunset
Clear noon		
Hard rain noon		

ploys a reinforcement learning coach that has been trained to exhibit expert-level driving behaviour, serving as an optimal demonstrator for imitation learning models. The data collection process involves the coach autonomously navigating through various scenarios while recording observations, actions, and relevant driving metrics. To execute data collection we use the already available Roach models[1] and repository which also consist of the settings set by the Leaderboard benchmark. This approach is particularly valuable as it ensures consistent, expert-level demonstrations across diverse driving conditions, eliminating the variability and potential inconsistencies often associated with human demonstrations. The collected dataset encompasses driving scenarios that follow the Leaderboard benchmark guideline for training. The protocol followed by Roach and defined by the Leaderboard benchmark states collection of 160 episodes where each episode is a route, which accounts for 12 hours of driving data. These episodes are collected on train towns and train weather conditions as previously mentioned in Table 2.1 and Table 2.2. To maintain uniformity and comparability, we perform training with the same collected dataset across all compared agents.

2.3.5. Baseline Methods

To evaluate our proposed methods, we establish baseline methods for comparison. Similar to the proposed methods, the baselines are reimplementations of CILRS [34] approach, drawing inspiration from recent research [137] to provide robust benchmarks for performance evalua-

[1]Roach code and model: https://github.com/zhejz/carla-roach

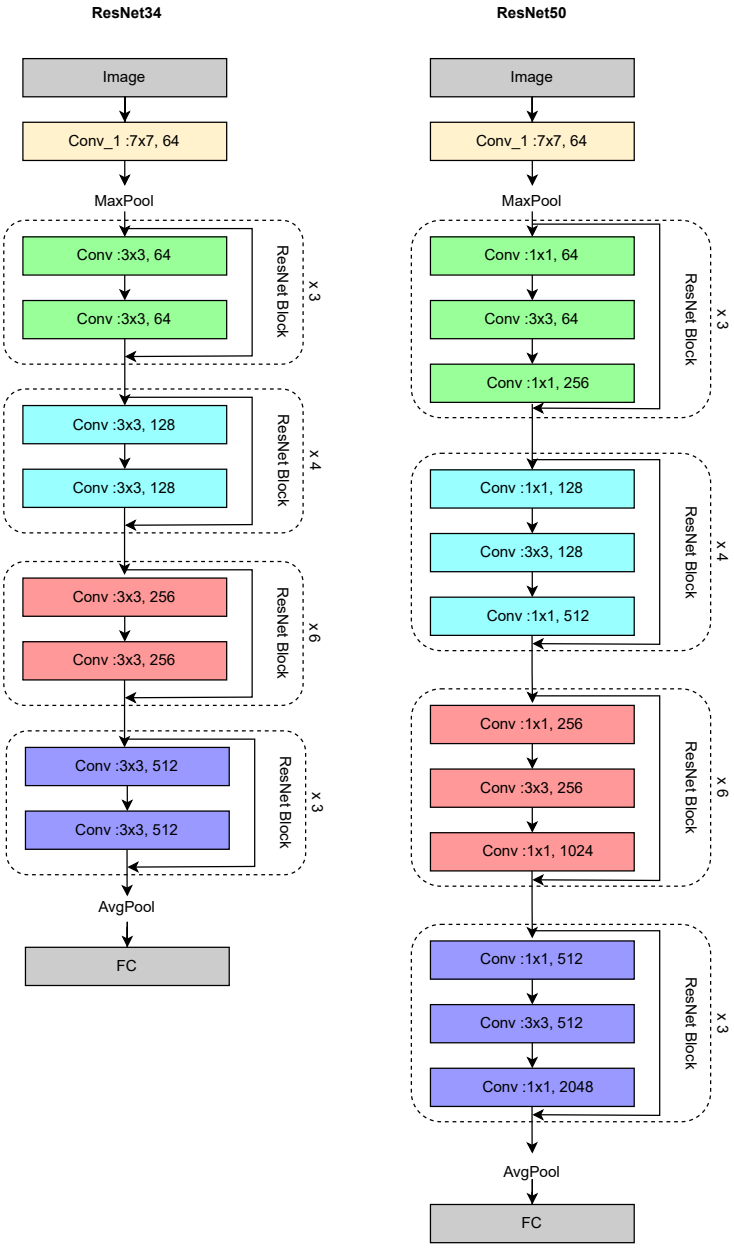


Figure 2.7: General architecture of ResNet34 and ResNet50 configurations [51], mentioning the arrangements of the residual blocks, filter sizes and depth at each convolution layer.

tion. Since our proposed methods focus on the pre-training of visual encoders, our baseline methods utilise already available visual encoders pre-trained on the ImageNet dataset. The rest of the architecture (i.e., decoder) remains identical to the architectures used in the proposed agents in sections 2.1 and 2.2. Recent works have predominantly employed two variants of pre-trained visual encoders: ResNet34 and ResNet50 (See Figure 2.7). Accordingly, we implement two baseline agents:

- Baseline Agent with ResNet34 encoder trained on ImageNet Classification (BAR34IC)
- Baseline Agent with ResNet50 encoder trained on ImageNet Classification (BAR50IC)

These baseline agents are trained upon the same loss function as the proposed methods, mentioned in equation 2.5. Whereas the pre-trained visual encoders utilised in the baselines, are pre-trained on the task of image classification over the ImageNet dataset [38].

2.3.6. Implementation Details of Proposed & Baseline Methods

We implement all the highlighted methods using the PyTorch [100] framework. The architecture is kept uniform across both proposed methods, where the encoder is formed by a ResNet50 and the decoder follows a conditional architecture as mentioned in Section 2.1.2 and Section 2.2.2. For the baseline methods, we maintain the same uniformity in architectures for the BAR50IC method. While giving the BAR34IC method a minute distinction of following the ResNet34 architecture in the encoder. Visual encoders used in both baseline methods are pre-trained on the ImageNet dataset over the task of image classification. Whereas, the visual encoders used in the proposed methods are initialised with the weights obtained by the appropriate proposed pre-training methods. Rather than performing pre-training of each proposed method from scratch, we use the already available pre-trained weights from the repository of SegVPR[2] and DINO[3]. This helps us in cutting down the requirement of

[2]SegVPR code and models: https://github.com/valeriopaolicelli/SegVPR
[3]DINO code and models: https://github.com/facebookresearch/dino

high amounts of resources and in saving train time.

Once pre-trained weights are transferred to the visual encoder , the proposed agents are trained over the same dataset of collected demonstrations. The initial demonstrations used to train remain the same across all methods, collected according to the technique specified in Section 2.3.4. Every trained agent is deployed as per the DAgger procedure [107] to collect and aggregate additional data where the agent fails to follow the supervising method (i.e. the Roach agent). Then a new agent is trained using the aggregated data and initial data to improve on the recorded failures. Basing our work on cited literature and following the Leaderboard benchmark standard, the aggregation is performed for 5 DAgger iterations.

Apart from the visual encoder f_M for every agent (proposed agents and baseline agents), we initialise rest of the agent with randomly initialised weights. We structure rest of the agent in the following way:

- The measurement encoder f_M is a stack of 2 fully-connected layers with the output dimension set to 128 at each layer.
- The join module f_J consists of 3 fully-connected layers with the output dimension set to 512, 512 and 256.
- Each of the action branches f_A holds 3 fully connected layers with the output dimensions set to 256, 256 and 2, respectively.

All modules consisting of fully connected layers use a rectified linear unit activation, except the last layers in action branches.

2.3.6.1. Implementation Challenges

We base most of the implementation decisions on the state-of-the-art methods [137]. Our implementations differ only in the input resolutions of images used for training the agent. Due to lack of heavy computing resources, we train our agents on images with a resolution down-scaled to 224×224 pixels instead of a roughly $\frac{256 \times 900}{224 \times 224} \approx 4.6$ times higher resolution [137]. This step allows us to train faster with the limited

amount of computational resources available, as our resolution of choice operates with ≈ 4.6 times less memory. Another challenge that arises due to opting for a smaller resolution for our experiments is that we become restricted from comparing our results with published results by other methods. To overcome this we form baselines based on state-of-the-art methods [137] and position our proposed methods as incremental changes to the baselines. With this approach, the experiments performed are able to show if the contributions of the proposed methods really improve performances or not. The final challenge we face and which remains is a long compute time. To run the experiments of training the agents and later evaluating them, it takes time spans of over a month for each agent. This limits us from reimplementing other published methods (other than our baselines, as the baselines are based on other methods) with our chosen settings. Therefore, to train the 4 methods we present later in the results, a compute time of over 5 month timespan was utilised.

Given the right number of required resources to train under with true resolutions used by other methods, the baseline can achieve a substantially higher score as per Zhang et al. [137]. Additionally, it may also enable comparison with state-of-the-art methods and perhaps those works. Due to lack of resources, we focus our implementation with downscaled settings on demonstrating the outcomes of our proposed methods rather than producing deployable agents.

2.3.7. Training Details

For every DAgger iteration of training for all methods, the entire neural network architectures with all the parameters are tuned against the loss function in equation 2.5. We carry out training for 20 epochs with a learning rate of $1e-4$ and weight decay of $1e-5$. The learning rate is stepped down to $1/10^{th}$ from epoch 15. We train in batches of 256 samples per batch. The training is run on an RTX 3090 with the data stored on Gen4 NVME solid state drives for faster reading of data. Training a single iteration takes around 20 to 35 hours. The variability in time is mostly caused by the increments in the size of the full dataset with every DAgger iteration as previously mentioned in section 2.3.6.

2.3.8. Metrics

To quantify and compare performance across methods, we employ two primary metrics. They are listed and defined as follows:

- **Route completion:** This metric quantifies the percentage of routes successfully completed by the agent under a given combination of settings. The mathematical representation of this metric is as follows:

$$\text{route completion} = \left(\frac{\text{number of completed routes}}{\text{total number of routes}} \right) \times 100. \tag{2.13}$$

  This metric accounts for every route that reaches the destination as per route planner's recommended path.

- **Distance completion:** This metric quantifies the average percentage of distance completed over all routes travelled under a given combination of settings. Mathematical representation of this metric is as follows:

$$\text{distance completion} = \left(\frac{\sum_{i=1}^{N} \text{completed distance}_i}{\sum_{i=1}^{N} \text{total route distance}_i} \right) \times 100, \tag{2.14}$$

  where,

  - N: total number of routes.
  - Completed Distance$_i$: distance successfully completed for the i-th route.
  - Total Route Distance$_i$: total distance of the i-th route.

  This metric terminates accounting distances when the agent exits the planned path which is guided by the route planner. Therefore distance is only accounted for as long as the agent follows the path to reach destination.

The route completion metric indicates the agent's overall success rate in completing routes, while the distance completion metric reveals the

average point of failure when routes are not completed successfully. Together, these metrics provide complementary insights into the agent's performance: route completion captures the frequency of successful navigation, while distance completion quantifies the extent of progress in unsuccessful attempts.

To extend the comparison beyond understanding if the agent completes routes and travels longer distances, we enlist fine-grained metrics. These metrics question the quality of performance over the behaviour of the agents. The fine-grained metrics are as follows:

- **Collision static:** Number of collisions with static elements that form the scene layout (such as traffic signal poles, trees, railings, pillars, etc.), normalised per kilometre travelled. The formula is given by:

$$\text{collision static} = \frac{1}{N} \sum_{i=1}^{N} \frac{\text{static collisions}_i}{\text{distance travelled}_i}, \tag{2.15}$$

  where N = total number of evaluation routes.

- **Collision pedestrian:** Number of collisions with pedestrians encountered on the route, normalised per kilometre travelled. The formula is given by:

$$\text{collision pedestrian} = \frac{1}{N} \sum_{i=1}^{N} \frac{\text{pedestrian collisions}_i}{\text{distance travelled}_i}, \tag{2.16}$$

  where N = total number of evaluation routes.

- **Collision vehicle:** Number of collisions with vehicles encountered on the route, normalised per kilometre travelled. The formula is given by:

$$\text{collision vehicle} = \frac{1}{N} \sum_{i=1}^{N} \frac{\text{vehicle collisions}_i}{\text{distance travelled}_i}, \tag{2.17}$$

  where N = total number of evaluation routes.

- **Red light infraction:** Number of red light traffic signals crossed, normalised per kilometre travelled. The formula is given by:

$$\text{red light infraction} = \frac{1}{N}\sum_{i=1}^{N}\frac{\text{red lights crossed}_i}{\text{distance travelled}_i}, \tag{2.18}$$

  where N = total number of evaluation routes.

The above listed metrics only account for the distance travelled and events occurred that take place while the agent is on the path recommended by the route planner. As soon as the agent leaves the path recommended by the route planner, the experiment terminates and accounting stops.

2.4. CONCLUSIONS OF CHAPTER 2

The chapter establishes research methodology of this study about pre-training for end-to-end autonomous driving. Based on the literature mentioned in Chapter 1, two methods are proposed to further explore the under-explored application of pre-training for navigating autonomous vehicles that learn from experience. We formally list out the contributions of this chapter in the following list:

- The visual encoder of autonomous driving agents is often pre-trained on a task that may not be directly linked to the task of driving. This makes it difficult for the agent to adapt to the task of driving and resist a fall of performance when facing covariate shift.
- VPR pre-training is proposed for the task of autonomous driving, to address how pre-training the visual encoder of an agent can help in holding up performance when encountering the covariate shift problem in imitation learning.
- Visual encoders used in autonomous driving are also heavily guided by labels when pre-trained on an image classification task. This limits the agent that uses this visual encoder from learning

a wide array of features as the pre-training is strongly limited to only classifying images.

- The use of the DINO pre-training method is proposed for the task of autonomous driving, based on how self-supervised pre-training can be beneficial in forming initialised weights over weights formed with supervised pre-training. To form better learned feature representations that help better resisting covariate shift.
- To evaluate the proposed methods, this chapter also defines the experimentation plan that establishes the benchmark standard and simulation environment. Additionally, it also formulates the specifications of the baseline methods that would contribute to relative comparisons.

3. EMPIRICAL INVESTIGATION

The experimental plans detailed in the previous chapter layout how the proposed methods that are to be evaluated against baseline methods. Extending that, this chapter presents and discusses the empirical findings and evaluations. The plan of the experiments is as follows:

1. We evaluate the VPR pre-trained agent against the BAR34IC and BAR50IC baseline agents using the route completion and distance completion metrics, which account for reachability and distance travelled.
2. Similarly we also evaluate the DINO pre-trained agent against BAR34IC and BAR50IC baseline agents over the route completion and distance completion metrics accounting for reachability and distance travelled.
3. Extending the research, we compare the proposed methods and the baseline methods over fine-grained metrics, which are collisions static, collisions pedestrian, collisions vehicle and red light infractions in order to evaluate the behaviours of agents while they drive.

The performed experiments are aimed to systematically evaluate the benefits of pre-training approaches over the defined performance metrics, examining whether exploring such a paradigm for autonomous driving can be a valuable endeavour for the current state of research in this research area.

3.1. VISUAL PLACE RECOGNITION PRE-TRAINING EXPERIMENTS

To weigh the benefits of the proposed method i.e., the VPR pre-trained agent, we compare the performance against our baselines. We track the performance over our primary metrics, i.e. route completion and distance completion. On observing the highest performing DAgger

iterations of every method in comparison, VPR pre-trained agent leads in completing routes against our baselines. When driving in environments that are already exposed to the policy (train environment settings), the VPR pre-trained agent completes a higher number of routes — 4% and 16.66% more than the BAR50IC and BAR34IC baselines, respectively. When driving in environments that are not present in the training data (new environment settings), the VPR pre-trained agent completes a higher number of routes — 7.05% and 10.90% more than the BAR50IC and BAR34IC baselines, respectively. The described results can be seen in Table 3.1 which compares the route completion scores of the VPR pre-trained agent against baseline agents.

Table 3.1: Highest route completion (%) scores of driving agents under training and new (testing) conditions, across all DAgger iterations reported.

Pre-training method	**Train town & weather**	**New town & weather**
BAR34IC	64.67 ± 2	49.35 ± 9
BAR50IC	77.33 ± 4	53.20 ± 1
VPR Pre-trained (ours)	**81.33** ± 4	**60.25** ± 2

To evaluate while focusing on the second primary metric, i.e. distance completion, we also look at the same DAgger iteration observed previously. Under the train environment settings, the VPR pre-trained agent reaches farther by 2.61% and 12.95%, whereas for new environment settings VPR pre-trained agent reaches farther by 13.78% and 10.26% on average, compared to the BAR50IC and BAR34IC baselines, respectively. These results can be seen in Table 3.2 which compares the distance completion scores of VPR pre-trained agent against baseline agents.

Additionally, we reveal the performances of the compared methods at every DAgger iteration and of every random seed. For both train and new environments, the VPR pre-trained agent steadily converges to its highest performance, already in the fourth iteration as compared to the baseline methods. This can be observed in the Figure 3.1. The same trend is also valid for the distance completion metric as well, as reported in Figure 3.2. We also reveal the route completion and distance

Table 3.2: Highest distance completion (%) scores of driving agents under training and new (testing) conditions across all DAgger iterations reported.

Pre-training method	**Train town & weather**	**New town & weather**
BAR34IC	79.02 ± 2	75.75 ± 7
BAR50IC	89.36 ± 2	72.23 ± 6
VPR Pre-trained (ours)	**91.97** ± 3	**86.01** ± 0

completion scores of DAgger iterations for each method in Tables 3.3, 3.4, 3.5 and 3.6.

Calculating results from experiments conducted in both train and new environments allows for the assessment of covariate shift. Our reported findings indicate that the VPR pre-trained agent not only achieves higher scores in completing routes and covering longer distances but also demonstrates enhanced resilience in comparison to baseline methods, especially in unseen conditions. This suggests enhanced generalisation capabilities, particularly when the visual encoder is trained beyond the conventional task of image classification on the ImageNet dataset. Hence, this approach enables a stronger resistance to covariate shift.

One of the notable differences between the VPR pre-trained and the baseline methods that influence the results is the task of training. The VPR pre-training method employs triplet loss for visual place recognition with the incorporation of semantic segmentation as the task of training, as opposed to baselines that rely on a classification loss. Such a setting prioritises understanding places in changing weather and lighting conditions, over forming unique encodings of object classes. According to the results, this choice enables creation of a visual encoder with superior weight initialisation to transfer to the task of autonomous driving.

Another distinction between the VPR pre-trained method and the baselines that influences the reported results is the dataset. The VPR pre-trained method's dataset is formed out of images that are relevant to the task of driving, whereas the baselines rely on the ImageNet dataset that

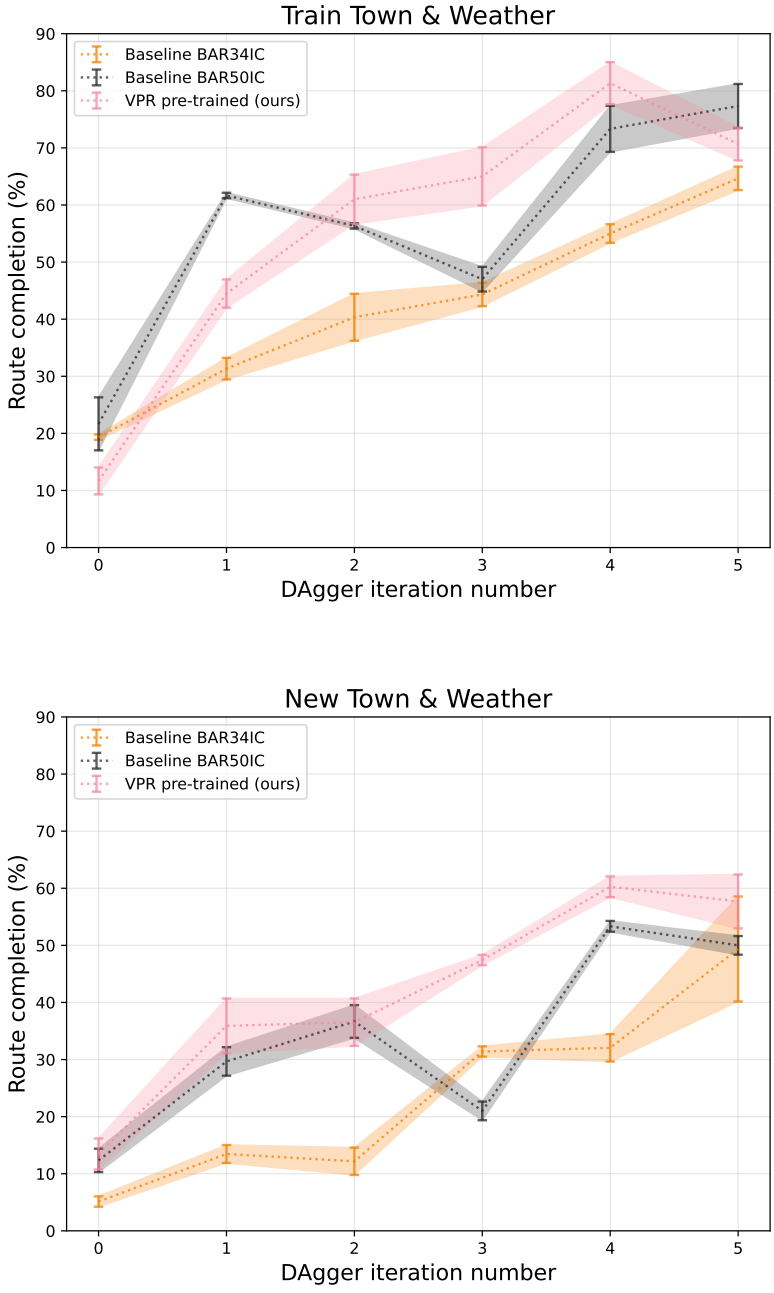


Figure 3.1: Route completion (%) of agents on the Leaderboard benchmark under training conditions (top) and testing conditions (bottom), evaluated three times over different seeds and plotted along with the average of performance.

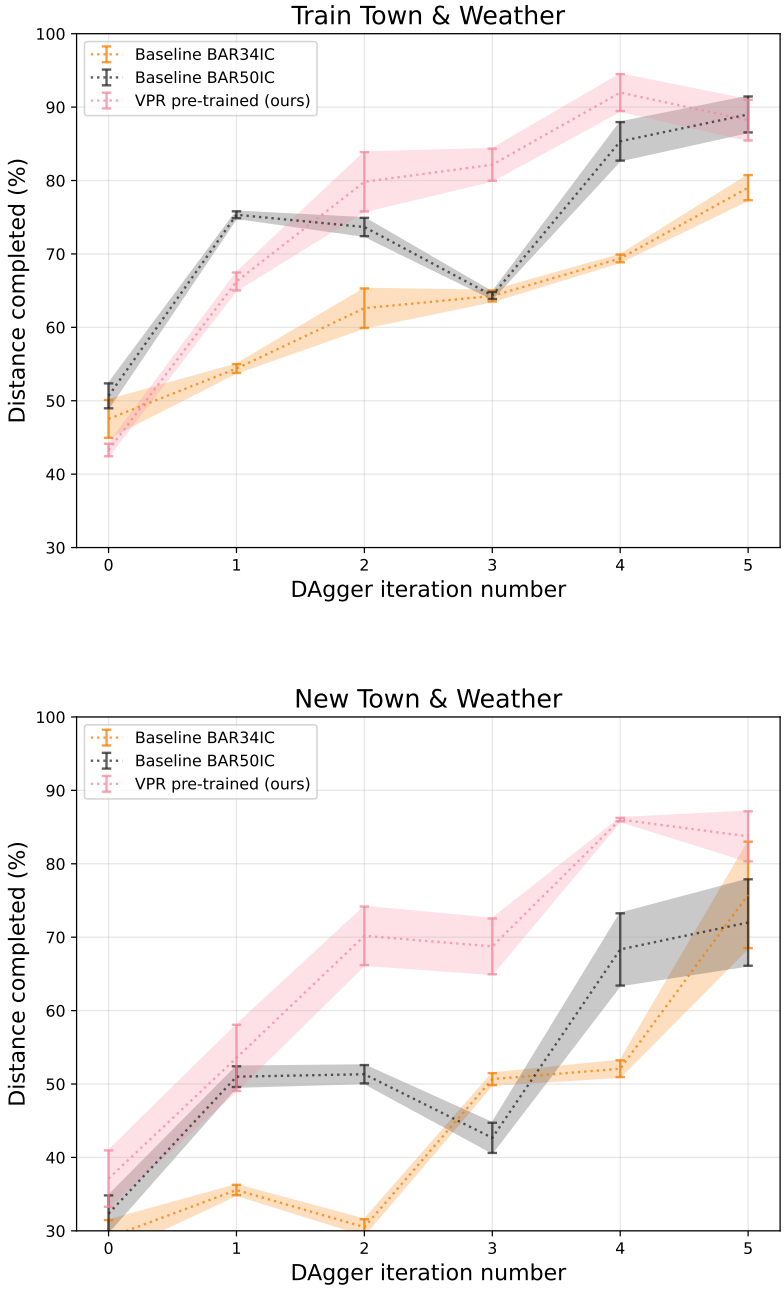


Figure 3.2: Distance completion (%) of agents on the Leaderboard benchmark under training conditions (top) and testing conditions (bottom), evaluated three times over different seeds and plotted along with the average of performance.

consists of irrelevant image classes such as cats, dogs, various objects, etc. We regard this distinction as a key factor, given that exposure to relevant data distributions is fundamental to the effectiveness of machine learning.

Table 3.3: Route completion (%) of every DAgger iteration under train town & weather conditions.

Pre-training method	**DAgger iterations**		
	0	1	2
BAR34IC	19.33 ± 0	31.33 ± 2	40.33 ± 4
BAR50IC	21.67 ± 5	61.67 ± 0	56.33 ± 0
VPR Pre-trained	11.67 ± 2	44.49 ± 2	61.0 ± 4
DINO Pre-trained	31.0 ± 4	56.33 ± 2	72.0 ± 2
Pre-training method	**DAgger iterations**		
	3	4	5
BAR34IC	44.33 ± 2	55.0 ± 2	64.67 ± 2
BAR50IC	47.0 ± 2	73.33 ± 4	77.33 ± 4
VPR Pre-trained	65.0 ± 5	$\mathbf{81.33} \pm 4$	70.67 ± 3
DINO Pre-trained	67.33 ± 0	67.33 ± 0	72.67 ± 3

Table 3.4: Distance completion (%) of every DAgger iteration under train town & weather conditions.

Pre-training method	**DAgger iterations**		
	0	1	2
BAR34IC	47.52 ± 3	54.39 ± 1	62.61 ± 3
BAR50IC	50.67 ± 2	75.33 ± 0	73.67 ± 1
VPR Pre-trained	43.28 ± 1	66.25 ± 1	79.82 ± 4
DINO Pre-trained	59.33 ± 4	72.67 ± 2	86.04 ± 1
Pre-training method	**DAgger iterations**		
	3	4	5
BAR34IC	64.27 ± 1	69.38 ± 1	79.02 ± 2
BAR50IC	64.33 ± 0	85.33 ± 3	89.36 ± 2
VPR Pre-trained	82.15 ± 2	$\mathbf{91.97} \pm 3$	88.22 ± 3
DINO Pre-trained	82.0 ± 0	84.0 ± 1	86.0 ± 1

Table 3.5: Route completion (%) of every DAgger iteration under new town & weather conditions.

Pre-training method	**DAgger iterations**		
	0	1	2
BAR34IC	5.13 ± 1	13.46 ± 2	12.18 ± 2
BAR50IC	12.33 ± 2	29.67 ± 2	36.67 ± 3
VPR Pre-trained	13.46 ± 3	35.9 ± 5	36.54 ± 4
DINO Pre-trained	11.33 ± 1	40.0 ± 1	43.33 ± 3
Pre-training method	**DAgger iterations**		
	3	4	5
BAR34IC	31.41 ± 1	32.05 ± 2	49.36 ± 9
BAR50IC	21.0 ± 2	53.20 ± 1	50.0 ± 2
VPR Pre-trained	47.44 ± 1	60.25 ± 2	57.69 ± 5
DINO Pre-trained	53.33 ± 2	61.0 ± 5	**62.18** ± 7

3.2. DINO PRE-TRAINING FOR AUTONOMOUS DRIVING EXPERIMENTS

To assess the effectiveness of the proposed DINO pre-training agent, we conduct a comparative analysis against two baseline methods and the previously introduced VPR pre-training agent. We perform this analysis by observing the performances of the denoted methods over our primary metrics, route completion and distance completion.

DINO pre-training outshines all methods in new environments for the metric of completing most routes. We compare the highest performing DAgger iterations of every method, and the results show DINO pre-training agent completes 1.93% higher number of routes on average than the previously proposed VPR pre-trained agent. Compared to the BAR34IC and BAR50IC baselines, pre-training using the DINO method benefits in completing 12.83% and 8.98% higher number of routes on average, respectively. Meanwhile, the DINO pre-trained agent falls behind the VPR pre-trained agent's route completion ability by 8.66% when exposed to train environments, and comes close the BAR50IC baseline in performance. The results are reported in Table 3.7 which

Table 3.6: Distance completion (%) of every DAgger iteration under new town & weather conditions.

Pre-training method	**DAgger iterations**		
	0	1	2
BAR34IC	29.07 ± 2	35.57 ± 1	30.51 ± 1
BAR50IC	32.33 ± 2	51.0 ± 1	51.33 ± 1
VPR Pre-trained	37.12 ± 4	53.56 ± 5	70.18 ± 4
DINO Pre-trained	37.33 ± 1	69.67 ± 1	73.0 ± 4
Pre-training method	**DAgger iterations**		
	3	4	5
BAR34IC	50.67 ± 1	52.08 ± 1	75.75 ± 7
BAR50IC	42.67 ± 2	68.33 ± 5	72.23 ± 6
VPR Pre-trained	68.75 ± 4	$\mathbf{86.01} \pm 0$	83.74 ± 3
DINO Pre-trained	76.33 ± 5	82.67 ± 5	80.33 ± 4

compares the route completion scores of DINO pre-trained agent against baseline agents and the VPR pre-trained agent. Following the distance completion metric scores, the DINO pre-training agent outperforms the BAR34IC and BAR50IC baseline agents by 6.92% and 10.44% on average respectively, in new environment settings. Meanwhile in train environment settings, DINO pre-training falls behind by 3.32% in comparison to the better performing baseline method, i.e. BAR50IC. In comparison to the VPR pre-trained agent, DINO pre-trained agent tends to cover shorter distances when not completing routes. This is reported in Table 3.8 which compares the distance completion scores of DINO pre-trained agent against baseline agents and the VPR pre-trained agent.

We also reveal the results over both the primary metrics of all DAgger iterations in Figures 3.3 and 3.4. Additionally, we reveal the route completion and distance completion scores of DAgger iterations for each method in Tables 3.3, 3.4, 3.5 and 3.6. The results indicate that the DINO pre-trained method converges to higher scores earlier than other methods and holds the lead for the case of route completion in new environment settings. In comparison to the baseline methods, DINO pre-training provides substantially better results.

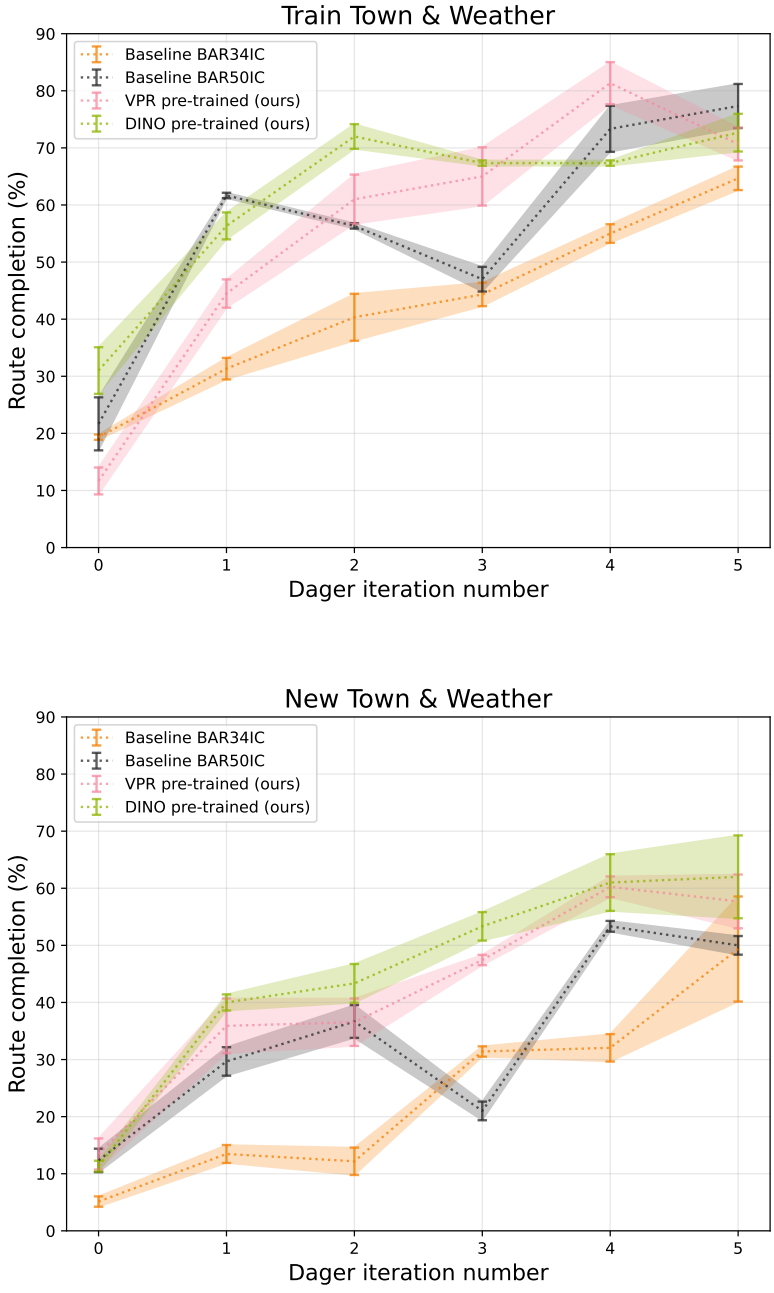


Figure 3.3: Route completion (%) of agents on the Leaderboard benchmark under training conditions (top) and testing conditions (bottom), evaluated three times over different seeds and plotted along with the average of performance.

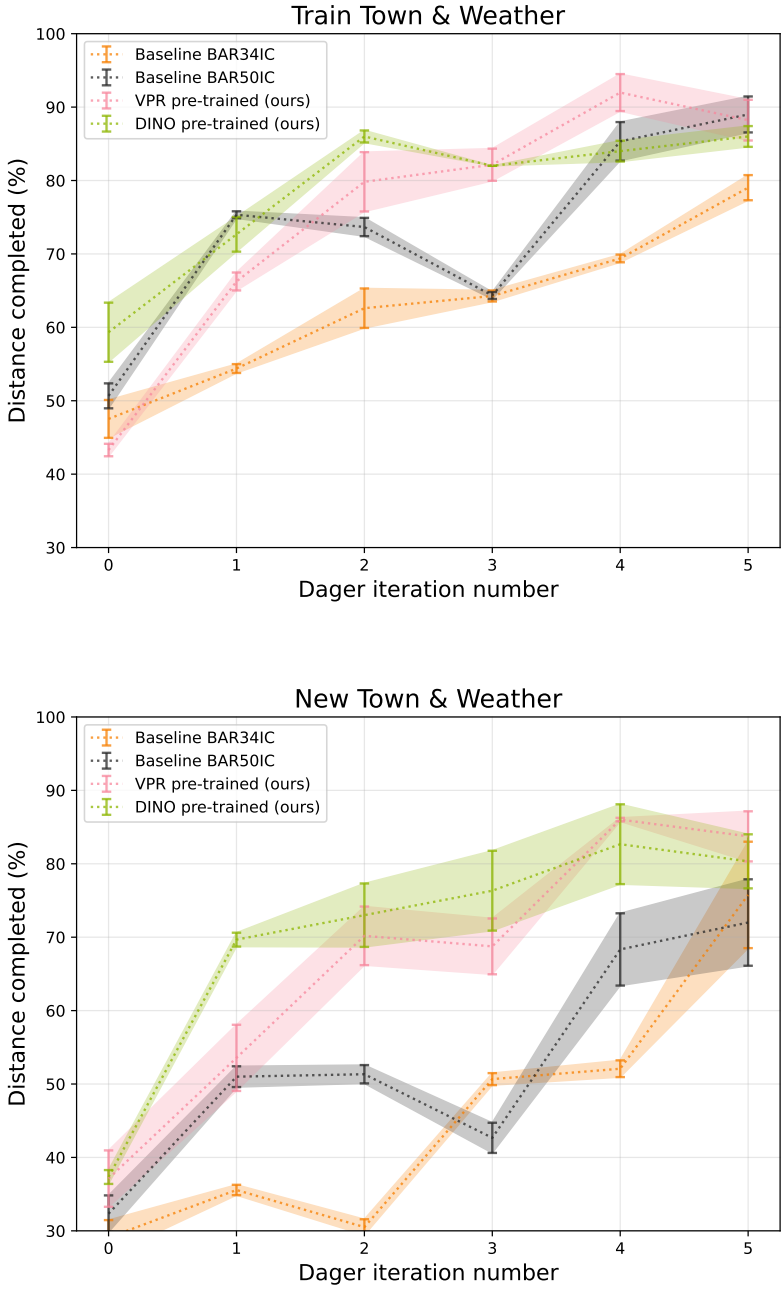


Figure 3.4: Distance completion (%) of agents on the Leaderboard benchmark under training conditions (top) and testing conditions (bottom), evaluated three times over different seeds and plotted along with the average of performance.

Table 3.7: Highest route completion (%) of driving agents under training and new (testing) conditions, across all DAgger iterations reported.

Pre-training method	**Train town & weather**	**New town & weather**
BAR34IC	64.67 ± 2	49.35 ± 9
BAR50IC	77.33 ± 4	53.20 ± 1
VPR pre-trained	**81.33** ± 4	60.25 ± 2
DINO pre-trained (ours)	72.67 ± 3	**62.18** ± 7

Table 3.8: Highest distance completion (%) of driving agents under training and new (testing) conditions, across all DAgger iterations reported.

Pre-training method	**Train town & weather**	**New town & weather**
BAR34IC	79.02 ± 2	75.75 ± 7
BAR50IC	89.36 ± 2	72.23 ± 6
VPR pre-trained	**91.97** ± 3	**86.01** ± 0
DINO pre-trained (ours)	86.04 ± 1	82.67 ± 6

The DINO pre-trained encoder and the compared baseline pre-trained encoders hold a notable commonality, that is the pre-training dataset being ImageNet. Yet, the results show substantial improvement on the route completion and distance completion metrics. As per the empirical evaluation, this improvement can be credited to the training method and loss functions that are employed during pre-training with the DINO method. This distinction allows improved learning and consequently encoding of better visual features that benefit the task of autonomous driving, hence proving image classification as a pre-training method to be outdated.

Another issue that the results of DINO pre-training address is of over-fitting. The baseline methods show a much larger gap in performance going from train environment settings to new environment settings, in comparison to the results of the DINO pre-training method. This gap could be present due to a strong over-fit, as the DINO pre-training method shows higher generalisation under new environment settings,

while scoring lesser route completion in train environment settings than the BAR50IC baseline.

Concluding this study, we point out that better pre-training leads to improved learning of the task of driving, especially over the conventional pre-training approaches.

3.3. EXTENDED ANALYSIS

In this section, we extend the analysis of the pre-training methods beyond the primary metrics. As the primary metrics validate the performances of the compared agents against the primary capabilities, i.e., reachability and driving without stopping, here we look at much more fine-grained behaviours. Therefore to quantify and analyse such behaviours of the agents while driving, we evaluate 4 additional metrics (described in Section 2.3.8). The fine-grained metrics we evaluate are collision static, collision pedestrian, collision vehicle and red light infractions. To bring the focus on how the compared methods are able to generalise, we calculate the aforementioned metrics based on how the agents drive in the new environment settings and not in train environment settings.

The DINO pre-trained agent shows consistent results across all four metrics, i.e., making the least collisions with static elements, pedestrians, vehicles and making the least red light infractions. This is reported in Figures 3.5, 3.6, 3.7 and 3.8. We also reveal the digit values of this evaluation in Table 3.9. The VPR pre-trained agent proves to be successful on the primary metrics by completing most routes and reaching farther distances. However, the VPR pre-trained agent fails to show consistency in showing lower error rates on metrics evaluating fine-grain behaviours, in contrast to the DINO pre-trained agent and the BAR50IC baseline agent. We suspect that this may be due to the previously identified over-fitting in the behaviour of the VPR pre-trained agent. Meanwhile, these results again affirm the improved generalisability of the DINO pre-trained agent when let to drive into new environments.

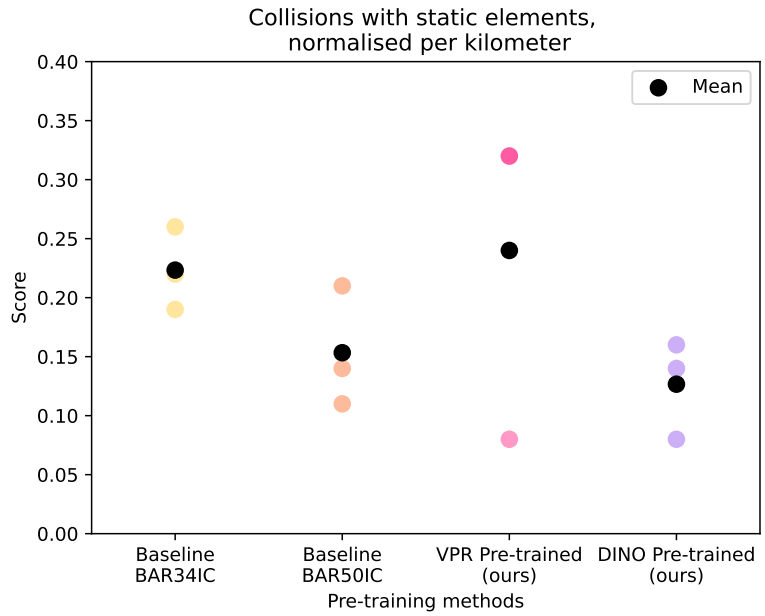


Figure 3.5: Number of collisions with static elements per kilometre over three random seed evaluations, along with the mean (lower is better).

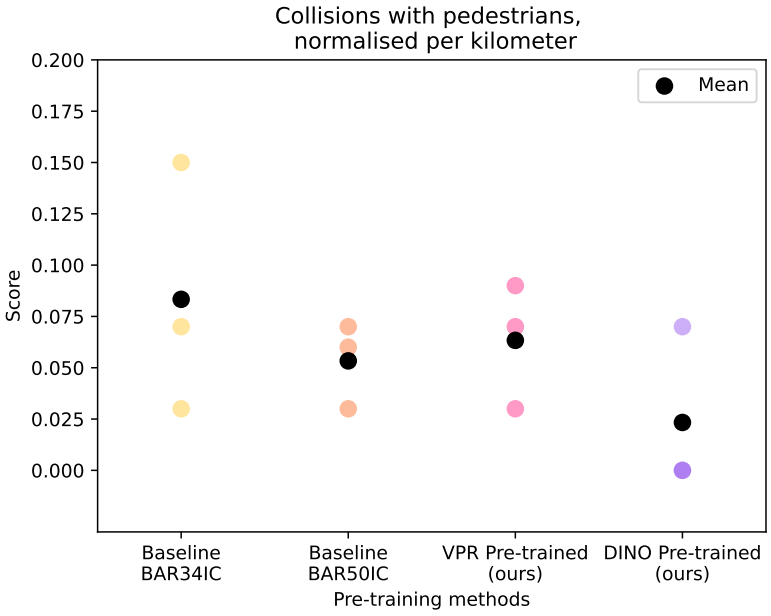


Figure 3.6: Number of collisions with pedestrians per kilometre over three random seed evaluations, along with the mean (lower is better).

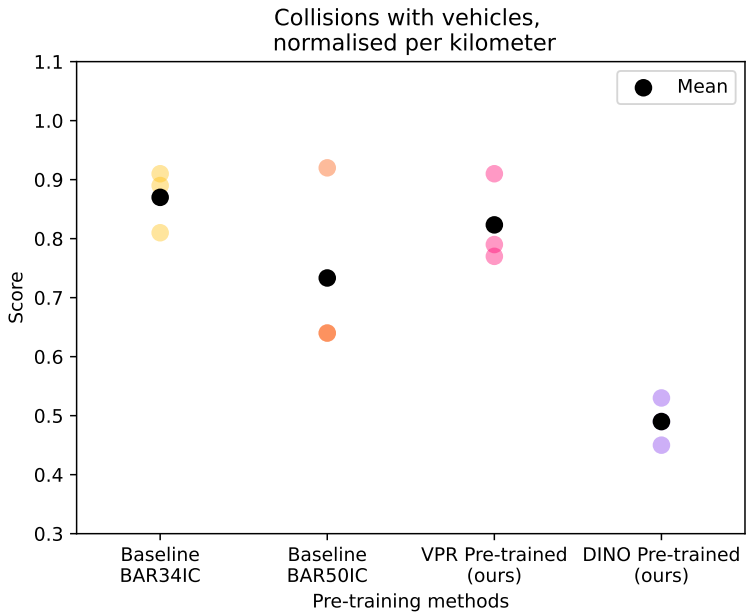


Figure 3.7: Number of collisions with vehicles per kilometre over three random seed evaluations, along with the mean (lower is better).

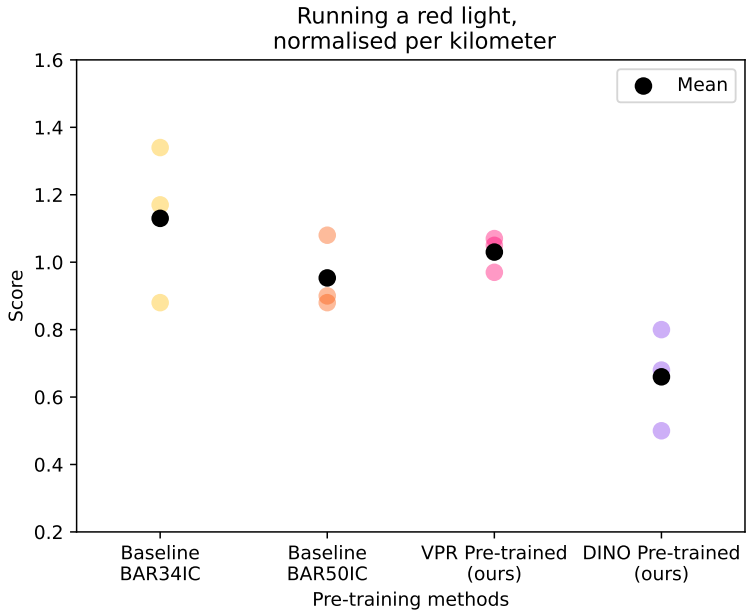


Figure 3.8: Number of red light infractions per kilometre over three random seed evaluations, along with the mean (lower is better).

Table 3.9: Number of collisions and infractions across the compared pre-training methods normalised by per distance travelled per kilometre (lower is better).

Pre-training method	**Collision static (↓)**	**Collision vehicle (↓)**	**Collision pedestrian (↓)**	**Red light infraction (↓)**
BAR34IC	0.22	0.87	0.08	1.13
BAR50IC	0.15	0.73	0.05	0.95
VPR Pre-trained	0.24	0.82	0.06	1.03
DINO Pre-trained	**0.13**	**0.49**	**0.02**	**0.66**

3.3.1. Comparison of the Proposed Methods

With references to the reported results in Sections 3.1, 3.4 and 3.3, we discuss the distinctions of the proposed methods, i.e., the VPR pre-trained and DINO pre-trained agents. We attribute the DINO pre-trained agent's superiority over VPR pre-trained agent, to the following points:

- **Pre-training dataset size:** Carrying out DINO pre-training leverages the ImageNet dataset with the size of 1 million images while VPR pre-trained agent's on-domain dataset of size $40,000$ falls short. Due to a smaller dataset size, the context to be learned may not be well-delivered in the pre-training phase.
- **Ability to reduce over-fitting:** In reference to other methods reported in Table 3.7, DINO pre-training shows better ability to complete routes in new environment settings. While this indicates improved generalisation, DINO pre-training also shows lower scores in completing seen routes in the seen weather conditions, additionally indicating over-fitting in the VPR pre-trained agent. This over-fit is also evident in the behaviour portrayed by the VPR pre-trained agent in the extended analysis as reported in Figures 3.5, 3.6, 3.7 and 3.8.
- **Open-ended learning capability:** The self-supervised learning ability to learn without image labels or annotations in the DINO

pre-training method enables generating features that can to outperform models trained with supervised learning. This is valid and evident especially in the fine-grained metrics that we evaluate on, as reported in Table 3.9. Such capability is quite a notable distinction since the provided images are of a very low resolution and yet only the DINO pre-trained agent shows substantial improvement.

3.4. CONCLUSIONS OF CHAPTER 3

This chapter presents the results of the experiments designed in Chapter 2 where the proposed VPR pre-trained agent and DINO pre-trained agents are compared against baselines. We formally list the overall conclusions of this chapter as follows:

- When the VPR pre-trained agent is deployed under new environment settings (i.e., unseen weather and towns) for the assessment of generalisability of the agents, it completes 7.05% and 10.90% higher number of routes than the BAR50IC and BAR34IC baselines respectively.
- This research empirically also shows that the VPR pre-trained agent achieves 13.78% and 10.26% higher distance completion than the BAR50IC and BAR34IC baselines respectively, in unseen environments. Therefore, pre-training the agent's visual encoder over the task of VPR improves the route completion and distance completion of the agent, in comparison to the pre-training of the same encoder on the ImageNet classification task. This is attributed to the distant relation of the task of the ImageNet classification to the task of autonomously driving in urban environments. The VPR task is centred around places which helps in understanding environments more than the ImageNet classification task.
- The VPR pre-trained agent achieves shows faster convergence to higher performance than both compared baseline agents, shown in Figure 3.1 and Figure 3.2.
- Whereas when the DINO pre-trained agent and the baseline agents are deployed in unseen conditions, the DINO pre-trained agent

completes 12.83% and 8.98% higher number of routes than the BAR34IC and BAR50IC baselines, respectively.

- The DINO pre-trained agent scores 6.92% and 10.44% higher on the distance completion metric compared to the BAR34IC and BAR50IC baselines respectively. The DINO pre-trained agent outshines against the baseline agents and shows how pre-training over the ImageNet dataset with another task instead of the over-simplified image classification task can generate better feature representations that help in improving the generalisation for the task of driving.

- On comparing the VPR pre-trained agent with the DINO pre-trained agent, the DINO pre-trained agent proves to be generalising better when deployed into unseen environments with a 1.92% higher number of completed routes as per the route completion metric.

- The DINO pre-trained agent also outperforms the VPR pre-trained agent on the behavioural metrics, i.e., by reporting 0.11 lower on the collision static metric, 0.33 lower on the collision vehicle metric, 0.04 less on the collision pedestrian metric and 0.37 less on the red light infraction metric, per distance travelled per kilometre. This shows that features generated by a visual encoder that is trained on self-supervision are better suited for the task of autonomous driving than features generated by a pre-trained process heavily guided by labels.

1. Our study demonstrates that pre-training the visual encoder on the VPR task using triplet loss improves autonomous driving performance compared to conventional ImageNet-based pre-training. Specifically, utilising triplet loss-based pre-training yields enhancements over the evaluated baselines, resulting in improvements of 7.05% in route completion and 13.78% in distance completion relative to the BAR50IC baseline, and improvements of 10.90% in route completion and 10.26% in distance completion over the BAR34IC baseline. These findings underline the effectiveness of task-specific pre-training in enhancing imitation learning outcomes for autonomous driving.

2. Our empirical findings demonstrate that pre-training with the DINO method enhances the agent's ability to generalise to previously unseen environments compared to baseline approaches. Specifically, the DINO pre-trained visual encoder achieves improvements of 8.98% in route completion and 10.44% in distance completion over the BAR50IC baseline, and improvements of 12.83% in route completion and 6.92% in distance completion compared to the BAR34IC baseline. These results highlight DINO's potential for significantly boosting robustness in imitation-based autonomous driving.

3. When comparing agents pre-trained with VPR and DINO methods, we conclude that the DINO-based agent exhibits superior generalisation performance. In particular, the DINO pre-trained agent achieves 1.93% higher route completion and demonstrates notably better performance in collision avoidance and adherence to traffic regulations. Specifically, per kilometre travelled, the DINO pre-trained agent records 0.11 fewer static collisions, 0.33 fewer vehicle collisions, 0.04 fewer pedestrian collisions, and 0.37 fewer red-light infractions compared to the VPR pre-trained agent.

[1] H. Abdelgawad and K. Othman. Multifaceted synthesis of autonomous vehicles' emerging landscape. In *Connected and autonomous vehicles in smart cities*, pages 67–113. CRC Press, 2020.

[2] P. A. Abrantes and M. R. Wardman. Meta-analysis of uk values of travel time: An update. *Transportation Research Part A: Policy and Practice*, 45(1):1–17, 2011.

[3] N. H. T. S. Administration. Critical reasons for crashes investigated in the national motor vehicle crash causation survey, 2015. `https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115`.

[4] T. Agarwal, H. Arora, and J. Schneider. Affordance-based reinforcement learning for urban driving. *arXiv preprint arXiv:2101.05970*, 2021.

[5] N. Akai, L. Y. Morales, T. Yamaguchi, E. Takeuchi, Y. Yoshihara, H. Okuda, T. Suzuki, and Y. Ninomiya. Autonomous driving based on accurate localization using multilayer lidar and dead reckoning. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2017.

[6] S. J. Anderson, S. B. Karumanchi, and K. Iagnemma. Constraint-based planning and control for safe, semi-autonomous operation of vehicles. In *2012 IEEE intelligent vehicles symposium*, pages 383–388. IEEE, 2012.

[7] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016.

[8] S. Atakishiyev, M. Salameh, H. Yao, and R. Goebel. Explainable artificial intelligence for autonomous driving: A comprehensive overview and field guide for future research directions. *IEEE Access*, 2024.

[9] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[10] J. A. Bagnell. An invitation to imitation. 2015.

[11] B. Baker, I. Akkaya, P. Zhokov, J. Huizinga, J. Tang, A. Ecoffet,

B. Houghton, R. Sampedro, and J. Clune. Video pretraining (VPT): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35:24639–24654, 2022.

[12] S. Behere and M. Törngren. A functional architecture for autonomous driving. In *Proceedings of the first international workshop on automotive software architecture*, pages 3–10, 2015.

[13] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[14] G. Berton, C. Masone, and B. Caputo. Rethinking visual geo-localization for large-scale applications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4878–4888, 2022.

[15] J. Bischoff and M. Maciejewski. Simulation of city-wide replacement of private cars with autonomous taxis in berlin. *Procedia computer science*, 83:237–244, 2016.

[16] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016.

[17] A. Broggi, M. Buzzoni, S. Debattisti, P. Grisleri, M. C. Laghi, P. Medici, and P. Versari. Extensive tests of autonomous driving technologies. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1403–1415, 2013.

[18] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

[19] L. Burns, W. Jordan, and B. Scarborough. Transforming personal

mobility, the earth institute, columbia university (2013). *The author declares competing financial interests: see go. nature. com/eb9bu3 for details*, 2016.

[20] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33: 9912–9924, 2020.

[21] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33: 9912–9924, 2020.

[22] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[23] C. Chen, A. Seff, A. Kornhauser, and J. Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision*, pages 2722–2730, 2015.

[24] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl. Learning by cheating. In *Conference on Robot Learning*, pages 66–75. PMLR, 2020.

[25] G. Chen, W. Choi, X. Yu, T. Han, and M. Chandraker. Learning efficient object detection models with knowledge distillation. *Advances in neural information processing systems*, 30, 2017.

[26] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li. End-to-end autonomous driving: Challenges and frontiers. *arXiv*, 2306.16927, 2023.

[27] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

[28] X. Chen and K. He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021.

[29] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF*

international conference on computer vision, pages 9640–9649, 2021.

[30] L. Chi and Y. Mu. Learning end-to-end autonomous steering model from spatial and temporal visual cues. In *Proceedings of the Workshop on Visual Analysis in Smart and Connected Communities*, VSCC '17, page 9–16, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450355063. doi: 10.1145/3132734.3132737. URL https://doi.org/10.1145/3132734.3132737.

[31] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, and A. Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[32] L. M. Clements and K. M. Kockelman. Economic effects of automated vehicles. *Transportation research record*, 2606(1):106–114, 2017.

[33] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4693–4700. IEEE, 2018.

[34] F. Codevilla, E. Santana, A. M. López, and A. Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9329–9338, 2019.

[35] M. . Company. Robotics and the factory of the future, 2017. https://www.mckinsey.com/capabilities/operations/our-insights/automation-robotics-and-the-factory-of-the-future.

[36] P. Daniušis, S. Juneja, L. Valatka, and L. Petkevičius. Topological navigation graph framework. *Autonomous Robots*, 45:633–646, May 2021. ISSN 1573-7527. doi: 10.1007/s10514-021-09980-x. URL https://doi.org/10.1007/s10514-021-09980-x.

[37] P. De Haan, D. Jayaraman, and S. Levine. Causal confusion in imitation learning. *Advances in neural information processing systems*, 32, 2019.

[38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE,

2009.

[39] J. Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[40] A. Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[41] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[42] M. Du. Overview of autonomous vehicle. In *Autonomous Vehicle Technology: Global Exploration and Chinese Practice*, pages 1–15. Springer, 2022.

[43] M. Du. *Autonomous Vehicle Technology: Global Exploration and Chinese Practice*. Springer Nature, 2022.

[44] D. J. Fagnant and K. Kockelman. Preparing a nation for autonomous vehicles: Opportunities, barriers, and policy recommendations. *Transportation Research Part A: Policy and Practice*, 77: 167–181, 2015. doi: 10.1016/j.tra.2015.04.003.

[45] D. J. Fagnant and K. M. Kockelman. The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios. *Transportation Research Part C: Emerging Technologies*, 40:1–13, 2014.

[46] Y. Fan, A. Guthrie, and D. Levinson. Waiting time perceptions at transit stops and stations: Effects of basic amenities, gender, and security. *Transportation Research Part A: Policy and Practice*, 88: 251–264, 2016.

[47] I. T. Forum. Urban mobility system upgrade. (6), 2015. doi: https://doi.org/https://doi.org/10.1787/5jlwvzdk29g5-en. URL `https://www.oecd-ilibrary.org/content/paper/5jlwvzdk29g5-en`.

[48] B. Friedrich. The effect of autonomous vehicles on traffic. *Autonomous driving: Technical, legal and social aspects*, pages 317–334, 2016.

[49] P. Gaussier and S. Zrehen. Perac: A neural architecture to control artificial animals. *Robotics and Autonomous Systems*, 16(2): 291–320, 1995. ISSN 0921-8890. doi: https://doi.org/10.1016/0921-8890(95)00052-6. URL `https://www.sciencedirect.com/science/article/pii/0921889095000526`. Moving

the Frontiers between Robotics and Biology.

[50] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[51] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[52] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[53] K. He, R. Girshick, and P. Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4918–4927, 2019.

[54] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[55] T. Hendershott, C. M. Jones, and A. J. Menkveld. Does algorithmic trading improve liquidity? *The Journal of Finance*, 66(1):1–33, 2011. doi: 10.1111/j.1540-6261.2010.01624.x.

[56] D. M. Herron. The past, present, and future of robotic surgical systems. *Surgical Endoscopy*, 30(10):3565–3578, 2016. doi: 10.1007/s00464-015-4510-7.

[57] A. J. Horowitz. Subjective value of time in bus transit travel. *Transportation*, 10(2):149–164, 1981.

[58] A. Hu, G. Corrado, N. Griffiths, Z. Murez, C. Gurau, H. Yeo, A. Kendall, R. Cipolla, and J. Shotton. Model-based imitation learning for urban driving. *Advances in Neural Information Processing Systems*, 35:20703–20716, 2022.

[59] H. Hu, Z. Qiao, M. Cheng, Z. Liu, and H. Wang. Dasgil: Domain adaptation for semantic and geometric-aware image-based localization. *IEEE Transactions on Image Processing*, 30:1342–1353, 2020.

[60] S. Hu, L. Chen, P. Wu, H. Li, J. Yan, and D. Tao. St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature

learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVIII*, pages 533–549. Springer, 2022.

[61] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang, L. Lu, X. Jia, Q. Liu, J. Dai, Y. Qiao, and H. Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[62] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7310–7311, 2017.

[63] A. Jaegle et al. Perceiver: General perception with iterative attention. *CoRR*, abs/2103.03206, 2021. URL `https://arxiv.org/abs/2103.03206`.

[64] A. Jaegle et al. Perceiver: General perception with iterative attention, 2021.

[65] J. Janai, F. Güney, A. Behl, and A. Geiger. Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*, 12(1–3): 1–308, 2020. ISSN 1572-2740. doi: 10.1561/0600000079. URL `http://dx.doi.org/10.1561/0600000079`.

[66] X. Jia, P. Wu, L. Chen, J. Xie, C. He, J. Yan, and H. Li. Think twice before driving: Towards scalable decoders for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21983–21994, 2023.

[67] S. Juneja, P. Daniušis, and V. Marcinkevičius. Combining multiple modalities with perceiver in imitation-based urban driving. *ALLSENSORS 2021, The Sixth International Conference on Advances in Sensors, Actuators, Metering and Sensing*, 2021.

[68] S. Juneja, P. Daniušis, and V. Marcinkevičius. Visual place recognition pre-training for end-to-end trained autonomous driving agent. *IEEE Access*, 11:128421–128428, 2023. doi: 10.1109/ACCESS.2023.3331678.

[69] S. Juneja, P. Daniušis, and V. Marcinkevičius. Dino pre-training for vision-based end-to-end autonomous driving. *Baltic Journal of Modern Computing*, 12(4), 2024.

[70] N. Kalra and S. M. Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice*, 94: 182–193, 2016.

[71] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[72] A. Karpathy et al. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[73] J. Kim and J. Canny. Interpretable learning for self-driving cars by visualizing causal attention. In *Proceedings of the IEEE international conference on computer vision*, pages 2942–2950, 2017.

[74] S. Kornblith, J. Shlens, and Q. V. Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[75] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[76] R. Kumar, P. Kumbharkar, S. Vanam, and S. Sharma. Medical images classification using deep learning: a survey. *Multimedia Tools and Applications*, 83(7):19683–19728, 2024.

[77] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE intelligent vehicles symposium (IV)*, pages 163–168. IEEE, 2011.

[78] J. Li, H. Li, J. Liu, Z. Zou, X. Ye, F. Wang, J. Huang, H. Wu, and H. Wang. Exploring the causality of end-to-end autonomous driving. *arXiv preprint arXiv:2407.06546*, 2024.

[79] S. Li. *AI-Empowered Personalized Prediction and Decision-Making Systems for Driving Co-Pilot*. University of California, Riverside, 2024.

[80] Y. Li, W. Yuan, S. Zhang, W. Yan, Q. Shen, C. Wang, and M. Yang. Choose your simulator wisely: A review on open-source simulators for autonomous driving. *IEEE Transactions on Intelligent*

Vehicles, 9(5):4861–4876, 2024. doi: 10.1109/TIV.2024.3374044.

[81] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[82] M. Magazine. Us commuters wait approximately 40 mins. a day for public transit, 2014.

[83] C. Masone and B. Caputo. A survey on deep visual place recognition. *IEEE Access*, 9:19516–19547, 2021.

[84] R. T. McAllister, Y. Gal, A. Kendall, M. Van Der Wilk, A. Shah, R. Cipolla, and A. Weller. Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning. International Joint Conferences on Artificial Intelligence, Inc., 2017.

[85] J. Mei, Y. Ma, X. Yang, L. Wen, X. Cai, X. Li, D. Fu, B. Zhang, P. Cai, M. Dou, et al. Continuously learning, adapting, and improving: A dual-process approach to autonomous driving. *arXiv preprint arXiv:2405.15324*, 2024.

[86] D. Metz. Developing policy for urban autonomous vehicles: Impact on congestion. *Urban Science*, 2(2):33, 2018.

[87] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[88] S. A. Miller and B. R. Heard. The environmental impact of autonomous vehicles depends on adoption patterns, 2016.

[89] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain, and J. Gao. Large language models: A survey. *arXiv preprint arXiv:2402.06196*, 2024.

[90] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[91] A. T. Moreno, A. Michalski, C. Llorca, and R. Moeckel. Shared autonomous vehicles effect on vehicle-km traveled and average trip duration. *Journal of Advanced Transportation*, 2018(1):8969353, 2018.

[92] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y. Cun. Off-road

obstacle avoidance through end-to-end learning. *Advances in neural information processing systems*, 18, 2005.

[93] T. Nguyen Duc, C. M. Tran, P. X. Tan, and E. Kamioka. Domain adaptation for imitation learning using generative adversarial network. *Sensors*, 21(14):4718, 2021.

[94] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp. A deep learning-based radar and camera sensor fusion architecture for object detection. In *2019 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pages 1–7, 2019. doi: 10.1109/SDF.2019.8916629.

[95] E. Ohn-Bar, A. Prakash, A. Behl, K. Chitta, and A. Geiger. Learning situational driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11296–11305, 2020.

[96] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 7(1-2):1–179, 2018. ISSN 1935-8253. doi: 10.1561/2300000053. URL `http://dx.doi.org/10.1561/2300000053`.

[97] K. Othman. Exploring the implications of autonomous vehicles: A comprehensive review. *Innovative Infrastructure Solutions*, 7(2): 165, 2022.

[98] Y. A. Ozaibi, M. Dulva Hina, and A. Ramdane-Cherif. End-to-end autonomous driving in carla: A survey. *IEEE Access*, 12:146866–146900, 2024. doi: 10.1109/ACCESS.2024.3473611.

[99] V. Paolicelli, A. Tavera, C. Masone, G. Berton, and B. Caputo. Learning semantics for visual place recognition through multi-scale attention. In *Image Analysis and Processing–ICIAP 2022: 21st International Conference, Lecce, Italy, May 23–27, 2022, Proceedings, Part II*, pages 454–466. Springer, 2022.

[100] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[101] S. D. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. S. Eng, D. Rus, and M. H. Ang. Perception, planning, control, and coordination for autonomous vehicles. *Machines*, 5(1):6, 2017. doi: 10.3390/machines5010006.

[102] B. Petryshyn, S. Postupaiev, S. Ben Bari, and A. Ostreika. Deep reinforcement learning for autonomous driving in amazon web services deepracer. *Information*, 15(2):113, 2024.

[103] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Proceedings of the 1st International Conference on Neural Information Processing Systems*, NIPS'88, page 305–313, Cambridge, MA, USA, 1988. MIT Press.

[104] A. Prakash, A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger. Exploring data aggregation in policy learning for vision-based urban autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11763–11773, 2020.

[105] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.

[106] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL `http://jmlr.org/papers/v21/20-074.html`.

[107] S. Ross, G. J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *J. Mach. Learn. Res.*, 15:627–635, 11 2010.

[108] A. Sauer, N. Savinov, and A. Geiger. Conditional affordance learning for driving in urban environments. In *Conference on Robot Learning*, pages 237–252. PMLR, 2018.

[109] Y. Savid, R. Mahmoudi, R. Maskeliūnas, and R. Damaševičius. Simulated autonomous driving using reinforcement learning: A comparative study on unity's ml-agents framework. *Information*, 14(5):290, 2023.

[110] M. Schneider, R. Krug, N. Vaskevicius, L. Palmieri, and J. Boedecker. The surprising ineffectiveness of pre-trained visual representations for model-based reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

[111] R. S. Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

[112] C. T. Systems. Future proofing infrastructure for connected and automated vehicles, 2017.

[113] A. Tampuu, T. Matiisen, M. Semikin, D. Fishman, and N. Muhammad. A survey of end-to-end driving: Architectures and training methods. *IEEE Transactions on Neural Networks and Learning Systems*, 33(4):1364–1384, 2022. doi: 10.1109/TNNLS.2020.3043505.

[114] M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[115] Y. Tang, H. Gong, N. Dong, C. Wang, W.-N. Hsu, J. Gu, A. Baevski, X. Li, A. Mohamed, M. Auli, et al. Unified speech-text pre-training for speech translation and recognition. *arXiv preprint arXiv:2204.05409*, 2022.

[116] 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.

[117] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011.

[118] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of field Robotics*, 25(8):425–466, 2008.

[119] V. Ušinskis, M. Makulavičius, S. Petkevičius, A. Dzedzickis, and V. Bučinskas. Towards autonomous driving: Technologies and data for vehicles-to-everything communication. *Sensors*, 24(11): 3411, 2024.

[120] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[121] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

[122] M. Wardman. Public transport values of time. *Transport policy*, 11 (4):363–377, 2004.

[123] J. Wei, J. M. Snider, J. Kim, J. M. Dolan, R. Rajkumar, and B. Litk-

ouhi. Towards a viable autonomous driving research platform. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 763–770. IEEE, 2013.

[124] P. Wu, X. Jia, L. Chen, J. Yan, H. Li, and Y. Qiao. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. In *NeurIPS*, 2022.

[125] P. Wu, L. Chen, H. Li, X. Jia, J. Yan, and Y. Qiao. Policy pre-training for autonomous driving via self-supervised geometric modeling. In *International Conference on Learning Representations*, 2023.

[126] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. López. Multimodal end-to-end autonomous driving. *Trans. Intell. Transport. Sys.*, 23(1):537–547, jan 2022. ISSN 1524-9050. doi: 10.1109/TITS.2020.3013234. URL https://doi.org/10.1109/TITS.2020.3013234.

[127] Y. Xiao, F. Codevilla, D. Porres, and A. M. Lopez. Scaling vision-based end-to-end driving with multi-view attention learning, 2023.

[128] M. Xu, N. Snderhauf, and M. Milford. Probabilistic visual place recognition for hierarchical localization. *IEEE Robotics and Automation Letters*, 6(2):311–318, 2021. doi: 10.1109/LRA.2020.3040134.

[129] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *International Conference on Robotics and Automation (ICRA)*, 2024.

[130] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE access*, 8:58443–58469, 2020.

[131] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3712–3722, 2018.

[132] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun. End-to-end interpretable neural motion planner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8660–8669, 2019.

[133] J. Zhang and K. Cho. Query-efficient imitation learning for end-to-end simulated driving. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 2891–2897. AAAI

Press, 2017.

[134] J. Zhang, J. Huang, S. Jin, and S. Lu. Vision-language models for vision tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[135] Q. Zhang, Z. Peng, and B. Zhou. Learning to drive by watching youtube videos: Action-conditioned contrastive policy pretraining. *European Conference on Computer Vision (ECCV)*, 2022.

[136] W. Zhang, S. Guhathakurta, J. Fang, and G. Zhang. Exploring the impact of shared autonomous vehicles on urban parking demand: An agent-based simulation approach. *Sustainable cities and society*, 19:34–45, 2015.

[137] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. Van Gool. End-to-end urban driving by imitating a reinforcement learning coach. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15222–15232, 2021.

[138] J. Ziegler et al. Making bertha drive—an autonomous journey on a historic route. *IEEE Intelligent transportation systems magazine*, 6 (2):8–20, 2014.

[139] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. Le. Rethinking pre-training and self-training. *Advances in neural information processing systems*, 33:3833–3845, 2020.

LIST OF AUTHOR PUBLICATIONS

Articles in international research journals with a citation index in the Clarivate Analytics Web of Science (CA WoS) database:

[A.1] Juneja, S., Daniušis, P., & Marcinkevičius, V. (2023). Visual place recognition pre-training for end-to-end trained autonomous driving agent. IEEE access, 11, 128421-128428.

[A.2] Juneja, S., Daniušis, P., & Marcinkevičius, V. (2024). DINO Pre-training for Vision-based End-to-end Autonomous Driving. Baltic Journal of Modern Computing, Vol. 12 (2024), No. 4, pp. 374–386.

Papers (and work presented) in peer-reviewed scientific conference proceedings:

[B.1] Juneja, S., Marcinkevicius, V., & Daniušis, P. Combining Multiple Modalities with Perceiver in Imitation-based Urban Driving. All Sensors 2021. 18^{th} July, 2021. Nice, France.

[B.2] Juneja, S., Daniušis, P., & Marcinkevičius, V. (2024). Monocular Depth Estimation Pre-training for Autonomous Driving. AI Sys 2024. 30^{th} September, 2024. Venice, Italy.

Additional work published during the studies but not included:

[C.1] Daniušis, P., Juneja, Valatka, L., & Petkevičius, L. Topological Navigation Graph framework. Autonomous Robots, vol. 45, no. 5, pp. 633-646.

[C.2] Daniušis, P., Juneja, S., Kuzma, L., & Marcinkevičius, V. (2022). Measuring Statistical Dependencies via Maximum Norm and Characteristic Functions. arXiv preprint arXiv:2208.07934.

CURRICULUM VITAE

Shubham Juneja graduated from Mumbai University with a Bachelor's Degree in Computer Engineering in 2014. In 2016, he graduated from Kaunas University of Technology with a Master's Degree in Informatics. He worked on researching AI for various products at Neurotechnology in Vilnius, Lithuania, from 2017 to 2024. From 2020 to 2024 he studied in the Ph.D. program at Vilnius University.

SUMMARY IN LITHUANIAN

TYRIMŲ SRITIS

Autonominio vairavimo technologija yra transformuojantis pokytis transporto srityje, kuris turi potencialo reformuoti automatizavimo, kelių eismo saugumo, eismo efektyvumo ir prieinamumo užtikrinimo sritis. Šios technologijos srities mokslinių tyrimų poreikį lemia savaeigių transporto priemonių sudėtingumas ir galimas jų poveikis. Svarbiausias šių mokslinių tyrimų uždavinys – sukurti patikimas sistemas, galinčias pažinti ir suprasti įvairias ir nenuspėjamas aplinkas ir jose naviguoti. Pažangiausių algoritmų progresas, ypač kompiuterių mokslo ir robotikos srityse, sudaro pagrindą tam, kaip autonominės transporto priemonės interpretuoja jutiklių duomenis, kad galėtų priimti sprendimus realiuoju laiku. Saugumas išlieka svarbiausiu rūpesčiu, todėl moksliniai tyrimai yra nukreipti į patikimų algoritmų, galinčių susidoroti su įvairiais scenarijais – nuo intensyvaus eismo iki sudėtingų oro sąlygų – ir galinčių užtikrinti, kad autonominės transporto priemonės bet kokiomis sąlygomis veiktų saugiau nei vairuotojai (žmonės), paiešką.

Navigacijos gebėjimo automatizavimas autonominio vairavimo ir robotikos moksliniuose tyrimuose buvo sprendžiamas taikant įvairias koncepcines paradigmas – dvi įtakingiausios paradigmos buvo SLAM (vienalaikė lokalizacija ir kartografavimas; angl. Simultaneous Localisation and Mapping) ir mašininiu mokymusi pagrįsti metodai. SLAM pagrįsti algoritmai padeda transporto priemonėms sudaryti ir atnaujinti nežinomos aplinkos žemėlapį ir tuo pat metu sekti savo buvimo vietą joje – tai labai svarbu realaus laiko navigacijai sudėtingoje ir dinamiškoje aplinkoje. Kita vertus, dirbtinio intelekto galimybių proveržis paskatino dažnesnį rėmimąsi mokymusi pagrįstais metodais – ypač neuroniniais tinklais grindžiamų sistemų naudojimą. Taikant šiuos metodus yra naudojami dideli duomenų rinkiniai, siekiant išmokyti sistemas suvokti, priimti sprendimus ir veikti įvairiais vairavimo scenarijais, taip pagerinant jų gebėjimą priimti sprendimus per labai trumpą laiką. Kartu šie metodai yra reikšmingos pažangos autonominio vairavimo mokslinių tyrimų srityje pagrindas.

Remiantis anksčiau aptartais pasiekimais, šioje disertacijoje yra siekiama praplėsti tyrimus, pateikiant naujas perspektyvas ir būdus. Šis tyrimas pradedamas nuo naujausių ir pažangiausių imitacinio mokymo (angl. imitation learning) metodų apžvalgos, atkreipiant dėmesį į tai, kad paruošiamasis mokymas (angl. pre-training) autonominio vairavimo srityje yra nepakankamai ištirtas. Daugumoje šios srities tyrimų yra pasirenkami vaizdo enkoderiai (angl. visual encoders), paruošiamojo mokymo būdu apmokyti ImageNet klasifikavimo užduočiai atlikti, vietoje to, kad būtų ieškoma geresnių alternatyvių metodų. Todėl šiame tyrime yra siūloma naujovė – ištisinio (angl. end-to-end) autonominio vairavimo užduočiai atlikti taikyti siūlomus paruošiamojo mokymo metodus. Vėliau tyrime šie pasiūlyti metodai yra palyginami su įprastiniais metodais, siekiant pademonstruoti šių naujų metodų geresnį efektyvumą.

TYRIMO OBJEKTAS

Tyrimo objektas – imitaciniu mokymu pagrįsti autonominio vairavimo metodai, daugiausia dėmesio skiriant paruošiamojo mokymo metodų ir jų poveikio vairavimo agento gebėjimui naviguoti nematytos aplinkos sąlygomis tyrimui.

TYRIMO TIKSLAS IR UŽDAVINIAI

Tyrimo tikslas – įdiegti ir ištirti imitaciniu mokymu ir giliaisiais neuroniniais tinklais pagrįstus autonominio vairavimo algoritmus, skirtus autonominei navigacijai aplinkoje, imituojančioje realaus pasaulio sąlygas, siekiant ištirti paruošiamojo mokymo metodus ir pagerinti generalizaciją matytos ir nematytos aplinkos sąlygomis.

Tyrimo tikslui pasiekti buvo įgyvendinti šie uždaviniai:

1. Atlikti naujausių ir pažangiausių imitacinio mokymo metodų, taikomų autonominio vairavimo procese, tyrimą ir nustatyti dabartinę autonominio vairavimo agentų vaizdo enkoderių paruošiamajame mokyme būklę.

2. Nustatyti ir pasiūlyti užduotį, skirtą vairavimo agento vaizdo enkoderio paruošiamajam apmokymui, kuri būtų labiau susijusi su vairavimo užduotimi nei tradiciškai naudojama ImageNet klasifikacija.

3. Nustatyti ir pasiūlyti savarankiškos priežiūros (angl. self-supervised) autonominio vairavimo agento vaizdo enkoderio paruošiamojo mokymo užduotį, kurios tikslas – generalizuoti geriau nei tai atlieka tradiciškai naudojamas ImageNet klasifikacijos paruošiamojo mokymo metodas.

4. Empiriškai ištirti siūlomus metodus su atitinkamais baziniais autonominio vairavimo agentais ir įvertinti gautus rezultatus.

TYRIMO METODAI

Šioje disertacijoje atskleistas tyrimas buvo atliktas remiantis šiais moksliniais metodais:

1. Atlikta literatūros apžvalga, atskleidžianti imitaciniu mokymu pagrįstus autonominio vairavimo metodus.

2. Naudojant kiekybinius ir kokybinius metodus buvo surinkti duomenys, atsižvelgiant į įvairius parametrus.

3. Pasiūlyti metodai yra įvertinti atliekant kelis pakartotinius eksperimentus (angl. experiment reruns) su skirtingais atsitiktinės inicializacijos parametrais (angl. seeds).

4. Konstruktyvaus tyrimo metu yra pasiūlyti pagerinimai ir patobulinimai, susiję su realaus pasaulio problemomis, taip pat pasiūlyti nauji teoriniai patobulinimo metodai.

5. Pritaikyti programinės įrangos kūrimo metodai, siekiant įgyvendinti pasiūlytus metodus ir eksperimentinę šios disertacijos dalį, įgyvendinant paruošiamojo mokymo ir autonominio vairavimo algoritmus bei papildomas vertinimo sistemas.

MOKSLINIS DARBO NAUJUMAS

Ši disertacija prisideda prie imitaciniu mokymu pagrįstų mokymosi metodų, skirtų ištisiniam autonominiam vairavimui, vystymo. Pagrindinis disertacijos mokslinis indėlis:

1. Išplėsti nepakankamai ištirtų paruošiamojo mokymo metodų, skirtų ištisiniam autonominiam vairavimui, tyrimus, pasiūlant būdus, leidžiančius atsisakyti priklausomybės nuo prižiūrimo paruošiamuojo mokymo, paremto vaizdo enkoderiais, skirtais vaizdų klasifikavimui.
2. Šiame darbe pasiūlytas vizualinio vietos atpažinimo panaudojimo paruošiamajame mokyme būdas, skirtas autonominiam vairavimui. Empiriškai parodyta, kad toks paruošiamasis mokymas pranoksta įprastai naudojamus paruošiamojo mokymo metodus.
3. Šiame darbe pasiūlytas ir kitas paruošiamojo mokymo metodas – DINO (savidistiliacijos be žymių, angl. self-distillation with no labels) paruošiamasis mokymas, kuris, remiantis eksperimentais, pasirodė esąs efektyvus.

PRAKTINĖ DARBO VERTĖ

Ši disertacija pagerina autonominio vairavimo metodų ir mokymo efektyvumą. Svarbiausias disertacijos praktinis indėlis:

1. Eksperimentai, atlikti simuliacinėse aplinkose, panaudojant paruošiamojo mokymo metodus, t. y. vizualinį vietos atpažinimą ir DINO, parodė didesnį atsparumą aplinkos pokyčiams. Tai reiškia, kad tokie metodai gali leisti pasiekti patikimą vairavimą nematytose aplinkose, todėl sumažėja mokymo duomenų poreikis.
2. Eksperimentai parodė greitesnį ir efektyvesnį konvergavimą, kai naudojami siūlomi metodai mokymo metu. Tai leidžia sumažinti grafinių procesorių skaičiavimo valandų kiekį, todėl sumažinamas anglies dioksido pėdsakas.

3. Disertacijoje sukurtas autonominio vairavimo metodų mokymo programinis kodas yra viešai skelbiamas, bei nurodomos kitos svarbios saugyklos (angl. repositories).

4. Šioje disertacijoje pateikiami ginamamuosius teiginius pagrindžiantys įrodymai naudojant industrijos ir mokslinių tyrimų standartines priemones, tokias kaip simuliatorius (angl. simulator), mašininio mokymo sistemos ir kt., kas leidžia tyrimo rezultatus ir išvadas lengvai perkelti į industrijos ir akademinės bendruomenės atliekamus mokslinius tyrimus.

GINAMIEJI TEIGINIAI

Šioje disertacijoje yra ginami šie teiginiai:

1. Paruošiamasis vaizdo enkoderio apmokymas atlikti vizualinio vietos atpažinimo užduotį naudojant trejeto nuostolius (angl. triplet loss), o ne įprastai naudojamą klasifikavimo užduotį ResNet architektūroje, pagerina imitaciniu mokymu pagrįstos autonominio vairavimo sistemos vairavimo efektyvumą vertinant maršruto įveikimo (angl. route completion) ir atstumo įveikimo (angl. distance completion) metrikas.

2. Paruošiamasis vaizdo enkoderio apmokymas naudojant ImageNet duomenų rinkinį ir DINO metodą, o ne įprastai naudojamą prižiūrimą vaizdų klasifikavimo užduotį ResNet architektūroje, generuoja geresnius požymius imitaciniu mokymu pagrįstam autonominiam vairavimui, kas leidžia pasiekti geresnį vairavimo efektyvumą vertinant maršruto įveikimo ir atstumo įveikimo metrikas.

3. Lyginant vizualinio vietos atpažinimo paruošiamąjį mokymą su DINO paruošiamuoju mokymu, DINO paruošiamojo mokymo metodas rodo geresnius rezultatus ir didesnį efektyvumą nematytose aplinkose, nes įveikia daugiau maršrutų ir sukelia mažiau susidūrimų su statiniais elementais, pėsčiaisiais ir transporto priemonėmis, taip pat atlieka mažiau raudono šviesoforo signalo pažeidimų.

TYRIMO APROBAVIMAS IR PUBLIKAVIMAS

Disertacijoje gauti rezultatai buvo paskelbti keturiuose moksliniuose darbuose: du moksliniai straipsniai paskelbti recenzuojamuose periodiniuose mokslo žurnaluose, du moksliniai darbai paskelbti ir pristatyti mokslinėse konferencijose. Toliau pateikiamas publikacijų ir pranešimų konferencijose sąrašas.

Straipsniai periodiniuose mokslo žurnaluose:

- Juneja, S., Daniušis, P., & Marcinkevičius, V. (2023). Visual place recognition pre-training for end-to-end trained autonomous driving agent. IEEE access, 11, 128421-128428.
- Juneja, S., Daniušis, P., & Marcinkevičius, V. (2024). DINO Pretraining for Vision-based End-to-end Autonomous Driving. Baltic Journal of Modern Computing, Vol. 12 (2024), No. 4, pp. 374–386.

Straipsniai (ir pristatyti darbai) recenzuojamose mokslinėse konferencijose:

- Juneja, S., Marcinkevicius, V., & Daniušis, P. Combining Multiple Modalities with Perceiver in Imitation-based Urban Driving. All Sensors 2021. 18^{th} July, 2021. Nice, France.
- Juneja, S., Daniušis, P., & Marcinkevičius, V. (2024). Monocular Depth Estimation Pre-training for Autonomous Driving. AI Sys 2024. 30^{th} September, 2024. Venice, Italy.

DISERTACIJOS STRUKŪRA

Šią disertaciją sudaro įvadas, trys skyriai, išvados ir santrauka lietuvių kalba. Įvado dalyje yra pateikiamas įvadas į tyrimą ir disertacijos apžvalga. Pirmajame skyriuje yra pateikiama literatūros apžvalga, apimanti imitaciniu mokymu grindžiamus autonominio vairavimo metodus ir

susijusias pamatines temas, tokias kaip autonominis vairavimas, imitacinis mokymas ir išankstinio mokymo metodai. Antrajame skyriuje yra aprašomi siūlomi metodai ir atlikti eksperimentai. Trečiajame skyriuje yra pristatomi ir analizuojami rezultatai, gauti eksperimentų metu. Galiausiai bendrųjų išvadų skyriuje pateikiamos išvados, suformuotos remiantis pristatytu tyrimu. Disertacijos pabaigoje yra pateikiamas literatūros sąrašas. Disertaciją sudaro 156 puslapių, 24 paveikslėlių ir 12 lentelių.

MOKSLINIŲ TYRIMŲ APŽVALGA

Autonominės transporto priemonės (AV) – tai savaeigės išmaniosios transporto priemonės, valdomos vidinėmis (angl. onboard) kompiuterinėmis sistemomis ir skirtos automatizuotam transportavimui. AV žada transformuoti pasaulį ir atskleidžia daugybę potencialių aspektų, kurie gali būti naudingi žmonių gyvenimui ir bendrai visai visuomenei. Autonominis vairavimas gali būti organizuojamas dviem metodais: moduliniu (angl. modular) arba ištisiniu (angl. end-to-end). Pagal modulinį metodą kiekvienam sistemos moduliui yra priskiriamos atskiros užduotys ir daug dėmesio reikia skirti inžinerinei daliai, o pagal ištisinį metodą vairuoti mokomasi holistiniu ir duomenimis paremtu keliu, naudojant demonstracijas. Atsižvelgiant į ištisinio metodo pranašumus, šiame tyrime yra koncentruojamasi į pastarąjį metodą. Ištisinis autonominio vairavimo metodas gali būti mokomas naudojant imitacinį mokymą arba sustiprintą mokymą (angl. reinforcement learning, RL). Imitacinis mokymas yra supaprastintas metodas, kuris yra kildinamas iš prižiūrimo mokymosi (angl. supervised learning), o RL mokosi taisyklių sąveikaudamas su aplinka, todėl yra lėtesnis ir reikalauja daug išteklių. Imitacinis mokymas susiduria su kovariacinio poslinkio (angl. covariate shift) problema, kai dėl duomenų pasiskirstymo skirtumų mokymo metu ir bandymo metu atsiranda grandininių klaidų (angl. cascading errors). Ankstyvieji metodai rodo neuroninių tinklų ir DAgger naudojimą, kas sudaro dabartinės būklės autonominio vairavimo, kuris remiasi imitaciniu mokymu, pagrindą. Vėlesnieji metodai naudoja įvairius aspektus, pavyzdžiui, sąlyginę architektūrą, daugiamodalumo (angl. multi-modality) įvedimą ir tobulinimą, demonstracijų kokybės gerinimą

ir kt. Šios kryptys tiesiogiai ar netiesiogiai siekia išspręsti kovariacinio poslinkio ir kitas susijusias problemas. Paruošiamasis mokymas tapo standartu daugelyje gilaus mokymosi tyrimų ir taikymo sričių. Pastaraisiais metais paskelbtuose tyrimuose pastebima ryški tendencija remtis ImageNet grindžiamu paruošiamuoju mokymu autonominio vairavimo srityje. Nors ImageNet grindžiamas paruošiamasis mokymas yra naudingas, tačiau jis taip pat gali būti neoptimalus. Tik keli tyrimai nagrinėjo alternatyvius paruošiamojo mokymo metodus autonominio vairavimo srityje. Todėl yra pastebėtina, kad šioje srityje yra susidariusi mokslinių tyrimų spraga. Atsižvelgiant į tai, įvardijame potencialias paruošiamojo mokymo paradigmas, kurios gali būti vertingos jas taikant autonominio vairavimo agentams. Atitinkamai yra siūlomi šie du metodai:

1. Vizualinio vietos atpažinimo paruošiamasis mokymas, skirtas autonominiam vairavimui.
2. DINO paruošiamasis mokymas, skirtas autonominiam vairavimui.

VIZUALINIO VIETOS ATPAŽINIMO PARUOŠIAMASIS MOKYMAS

Šis metodas yra siūlomas siekiant išspręsti kovariacinio poslinkio problemą autonominio vairavimo imitaciniame mokyme, specifiškai koncentruojantis į oro ir apšvietimo sąlygų variacijas. Keliame hipotezę, kad autonominis vairavimas labai priklauso nuo specifinių vaizdinių požymių, kurių gali nepavykti efektyviai užfiksuoti taikant ImageNet paruošiamąjį mokymą, kuris grindžiamas vaizdų klasifikavimu – užduotimi, tolima vairavimui. Todėl siūlome agento vaizdo enkoderio paruošiamąjį apmokymą VPR būdu, nes VPR duomenų rinkiniai savyje apima orų ir apšvietimo sąlygų variacijas, leidžiančias surasti (atpažinti) vietą (angl. achieve place retrieval) kintančiomis sąlygomis. Perduodant taip paruošiamojo mokymo būdu apmokytą enkoderį vairavimo agentui, šiuo metodu siekiama pagerinti agento gebėjimą prisitaikyti prie nematytų oro ir apšvietimo sąlygų ir sušvelninti kovariacinio poslinkio poveikį (kadangi kovariacinis poslinkis yra viena iš pagrindinių problemų taikant imitaciniu mokymu pagrįstus metodus).

Įprastai imitaciniu mokymu pagrįstas autonominio vairavimo metodas yra laikomas agentu, kurį sudaro specialios architektūros neuroninis tinklas. Šiame skyriuje pasiūlytą metodą vadiname VPR paruošiamojo mokymo metodu, o šio metodo rezultatą – VPR paruošiamojo mokymo metodu apmokytu agentu. VPR paruošiamojo mokymo metodu apmokyto agento sukūrimas yra pristatomas dviem dalimis. Pirmiausia aprašomas siūlomas vaizdo enkoderio paruošiamasis mokymas. Po paruošiamojo mokymo aprašomas paruošiamojo mokymo metodu apmokyto vaizdo enkoderio integravimas į agentą ir mokymas atlikti autonominio vairavimo užduotį.

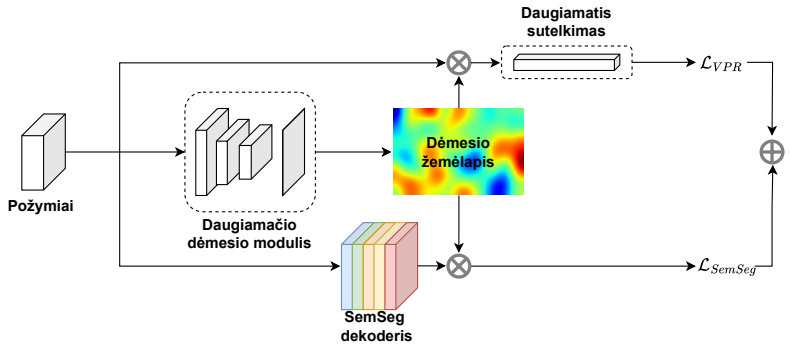
Vaizdo enkoderio paruošiamasis mokymas naudojant VPR

VPR yra fundamentali kompiuterinės regos užduotis, kuria siekiama nustatyti ir atpažinti anksčiau aplankytas vietas naudojant tik vaizdinę informaciją iš vaizdų ar vaizdo įrašų sekų. Pagrindinis VPR tikslas – nustatyti, ar pateiktas užklausos vaizdas (angl. query image) atitinka vietą, esančią referuojamoje vaizdų duomenų bazėje, ir taip atsakyti į klausimą „Ar aš čia jau buvau?“. VPR pagrįstos sistemos yra mokomos dirbti sudėtingomis realaus pasaulio sąlygomis, tokiomis kaip apšvietimo sąlygų variacijos (diena ir naktis), sezoniniai pokyčiai (vasara ir žiema), skirtingos oro sąlygos, varijuojantys žiūros taškai ir dinaminiai objektai veiksmo vietoje (angl. scene). Mokymo metu veikiami šių variacijų, neuroniniai tinklai išmoksta išskirti požymius (angl. feature representations), kurie išlieka atsparūs aplinkos pokyčiams.

Siūlomas VPR paruošiamojo mokymo metodas išplečia naujausio VPR metodo – SegVPR – taikymą autonominio vairavimo agento vaizdo enkoderio paruošiamajam mokymui. SegVPR atlieka mokymą naudodamas ImageNet iš anksto apmokytą ResNet50 pagrįstą vaizdo enkoderį. Mokymo metu šis enkoderis paskirstomas dviem užduotims:

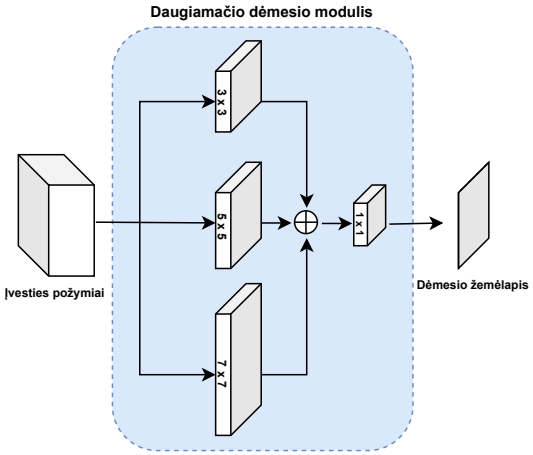
1. Vizualinio vietos atpažinimo (VPR) užduotis: ši užduotis yra pagrindinė.
2. Semantinio segmentavimo (angl. semantic segmentation) (SemSeg) užduotis: ši užduotis yra pagalbinė.

Šios dvi užduotys leidžia mokytis požymių, kurie sujungia konkrečios vietos informaciją (angl. place-specific information) ir semantinę veiksmo vietos informaciją (angl. semantic information from the scene). Abiejų užduočių derinys padeda orientuoti visą neuroninį tinklą, į kurią veiksmo vietos (aplinkos) sritį reikia žiūrėti. Tai padaryti papildomai padeda ir dekoderyje (angl. decoder), kuris pavaizduotas S.1 paveikslėlyje, esančių daugiamačio dėmesio (angl. multi-scale attention) ir daugiamačio sutelkimo (angl. pooling) mechanizmų naudojimas.



S.1 pav.: Paveikslėlyje pavaizduota SegVPR dekoderio struktūra, sudaryta iš segmentavimo dekoderio, daugiamačio dėmesio modulio ir daugiamačio sutelkimo modulio.

Daugiamačio dėmesio modulis yra naudojamas siekiant sutelkti dėmesį į svarbiausius įvesties vaizdo regionus ir papildomai tinkama kryptimi nukreipti semantinį segmentavimą mokymo metu. Bendrai, daugiamačio dėmesio modulis įvertina požymius, naudodamas skirtingą skiriamąją gebą, priskiria svorius (angl. weights) pagal jų svarbą ir sujungia požymius į vieną reprezentaciją . Šis modulis kaip įvestį priima enkoderio išvestį iš ketvirtojo konvoliucinio sluoksnio ir praleidžia per 3, 5 ir 7 dydžio branduolių filtrą, atvaizduotą S.2 paveikslėlyje. Atlikus modulio išvesčių padidinimą (angl. upsampling) ir sujungimą pagal kanalus (angl. channel-wise concatenation), sudaromas dėmesio žemėlapis. Dėmesio žemėlapyje esantys dėmesio parametrai rodo, kur yra sutelkiamas dėmesys. SegVPR naudoja daugiamatį sutelkimą, kad išgautų semantinę ir vaizdo informaciją skirtingais semantinės informacijos lygiais. Šis mechanizmas naudoja enkoderio ketvirtojo ir penktojo konvolicinių sluoksnių informaciją. Šiems požymiams yra priskiriami



S.2 pav.: Paveikslėlyje pavaizduotas SegVPR architektūroje naudojamas daugiamačio dėmesio modulis, kuris naudoja kelis erdvinius mastelius, kad būtų užfiksuoti skirtingo dydžio objektai, ir būtų sudarytas dėmesio žemėlapis.

parametrai pagal dėmesio žemėlapį ir yra gaunamas globalus deskriptorius (angl. global descriptor) (sinonimas reprezentacijai).

Ribinė trejeto nuostolių funkcija (angl. triplet margin loss) naudojama atliekant deskriptorių VPR mokymą. Todėl į SegVPR architektūrą pateikiant įvesties vaizdą, jos daugiamačio sutelkimo modulis išgauna deskriptorių (kaip pavaizduota S.1 pav.), kuriam priskiriame reikšmę F. Pagal klasikinio trejeto mokymo metodą užklausos (angl. query) arba pririšimo (angl. anchor), teigiamų (angl. positive) ir neigiamų (angl. negative) deskriptorių pavyzdžiai yra imami iš pavyzdžių archyvo. Teigiami ir užklausos pavyzdžiai priklauso artimoms GPS koordinatėms, o neigiami pavyzdžiai yra imami iš tolimos vietos. Kiekvieno trejeto pavyzdžio VPR nuostoliai yra nustatomi pagal formulę:

$$\mathcal{L}_{VPR} = h(d(F_{query}, F_{pos}) + m - d(F_{query}, F_{neg})), \tag{S.1}$$

kurioje h yra Hinge nuostoliai $h(x) = max(x, 0)$, d yra Euklido atstumas, $m > 0$ yra fiksuota reikšmė (angl. fixed margin), ir F_{query}, F_{pos} ir F_{neg} atitinkamai reiškia užklausos, teigiamų ir neigiamų trejeto pavyzdžių

reikšmes.

Semantinio segmentavimo nuostoliai (angl. semantic segmentation loss) $\mathcal{L}_{SemSeg}$ yra nustatomi pagal formulę:

$$\mathcal{L}_{SemSeg} = -\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} y_i \cdot \log p_i^{y_i}(M^i \cdot f_d^i), \tag{S.2}$$

kurie yra lygūs kryžminės entropijos nuostoliams (angl. cross-entropy loss), apskaičiuojamiems kiekvienai klasei y_i, pikselio i iš vaizdo $\mathcal{I}$, kur M_i yra dėmesio žemėlapis, susijęs su požymiu f_d^i, o f_d^i žymi klasės y_i tikimybę. Tiek f_d^i, tiek p_i yra segmentavimo dekoderio modulio išvestys, o M_i yra daugiamačio dėmesio modulio išvestis.

Suminių nuostolių funkcija (angl. overall loss function) yra VPR nuostolių ir semantinio segmentavimo nuostolių funkcijų suma:

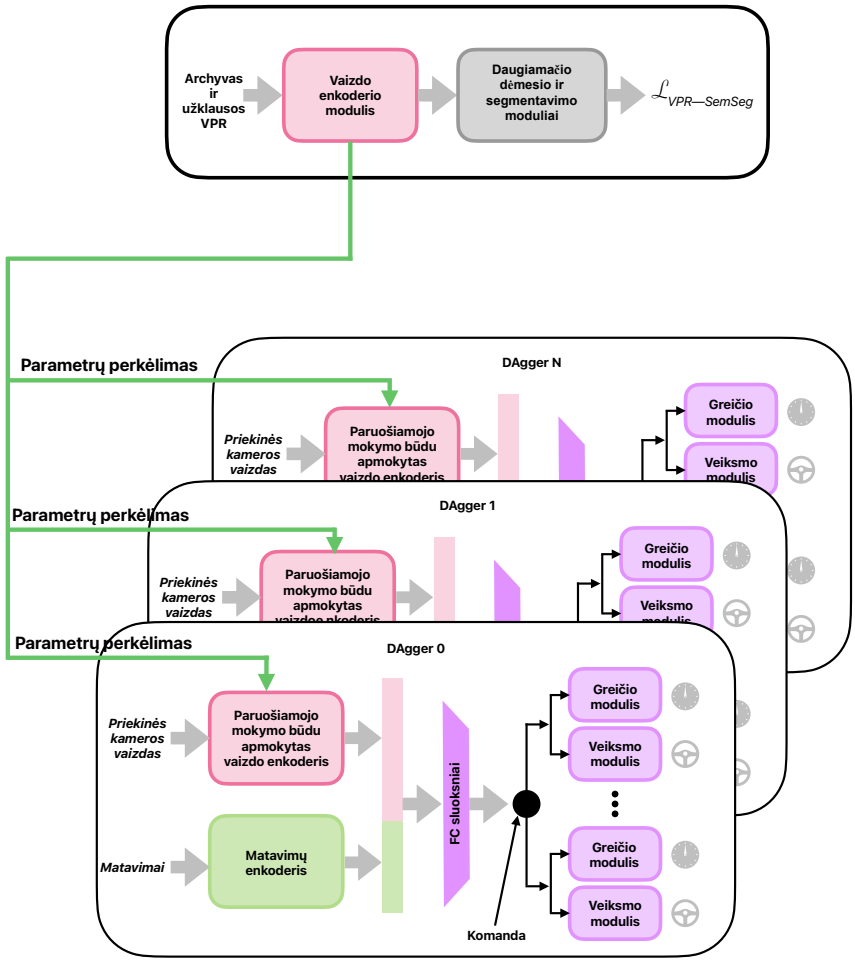
$$\mathcal{L}_{VPR-SemSeg} = \mathcal{L}_{VPR} + \alpha \cdot \mathcal{L}_{SemSeg}, \tag{S.3}$$

kur $\alpha > 0$ yra skaliarinis semantinio segmentavimo nuostolių dydis (angl. scalar weight for semantic segmentation loss).

SegVPR naudoja specialiai sukurtą duomenų rinkinį, užfiksuotą CARLA 0.9.10 simuliatoriuje. Duomenų rinkinys apima GPS informaciją ir 25 semantinių klasių pikselių semantinę anotaciją. Jame užfiksuota daugiau kaip 40000 vaizdų (10091 kiekvienam scenarijui), surinktų Town03 ir Town10 žemėlapiuose esant skirtingam orui – nuo giedro vidurdienio (angl. Clear Noon) iki smarkaus lietaus saulėlydžio (angl. Hard Rain Sunset).

VPR paruošiamojo mokymo metodu apmokyto agento mokymas

Siekdami išnaudoti patikimas vaizdines reprezentacijas, išmoktas kintančiomis oro sąlygomis, kurios buvo pateiktos paruošiamojo mokymo metu, į savo agento neuroninio tinklo sistemą integruojame vaizdo enkoderį iš SegVPR architektūros. Šiame poskyryje išsamiau aprašome VPR paruošiamojo mokymo metodu apmokyto agento mokymą ir architektūrą, į kurią yra įterptas paruošiamojo mokymo būdu apmokytas vaizdo enkoderis. Architektūra pavaizduota S.3 paveikslėlyje.



S.3 pav.: Paveikslėlyje pavaizduota bendra siūlomo vizualinio vietos atpažinimo paruošiamojo mokymo metodo blokinė schema, kurioje iš pradžių vaizdo enkoderis yra paruošiamojo mokymo būdu apmokomas VPR užduoties (viršuje), po kurio yra atliekamas parametrų perkėlimas, kad būtų galima mokyti ištisinio vairavimo užduočiai (apačioje).

Mūsų agento architektūra yra grindžiama CILRS, kur neuroninis tinklas yra sąlygotas aukšto lygio (angl. high-level) navigacinių komandų. Šiomis komandomis, kurias generuoja maršruto planuotojas (pateikiamas modeliavimo programinės įrangos) pagal kelionės tikslą, yra orientuojamas agento sprendimų priėmimo procesas. Pradinėms vairavimo demonstracijoms surinkti naudojame automatizuotą sistemą, pagrįstą

RL ekspertu, pasiūlytu Roach metodo, kuri suformuoja mokymo ir validavimo duomenų rinkinius. Ši metodika ne tik panaikina operatorių-žmonių poreikį, bet ir užtikrina nuoseklių ir kokybiškų duomenų surinkimą. Po paruošiamojo apmokymo etapo, atliekame savo pasiūlyto agento mokymą. Tada įgyvendiname DAgger procesą, kurio metu mūsų paruošiamojo mokymo būdu apmokytas agentas aktyviai generuoja vairavimo elgsenos įpročius, o ją tuo pačiu metu prižiūri Roach agentas. Kai atsiranda neatitikimų tarp mūsų agento siūlomų veiksmų ir Roach agento pasiūlytų veiksmų, šie atvejai yra registruojami ir kaupiami mokymo duomenų rinkinyje. Šis koreguojančiųjų demonstracijų agregavimas vykdomas pagal originalaus DAgger algoritmo nustatytą metodiką, todėl mūsų agentas gali mokytis iš eksperto koreguojančiųjų veiksmų.

Siūlomą VPR paruošiamojo mokymo būdu apmokyto agento architektūrą (S.3 pav.) sudaro du lygiagrečiai veikiantys kodavimo srautai: matavimų enkoderis, kuris apdoroja esamą greitį ir one-hot būdu koduotas aukšto lygio komandas, ir SegVPR enkoderis, kuris apdoroja vaizdo įvestį. Abiejų enkoderių išvestys yra sujungiamos (angl. concatenated) ir apdorojamos jungtiniame modulyje, sudarytame iš visiškai sujungtų sluoksnių, kuris sumažina kombinuotų požymių dimensiškumą (angl. dimensionality). Po to ši jungtinė reprezentacija yra perduodama į specializuotas veiksmų šakas, kur kiekviena šaka atitinka konkrečią aukšto lygio komandą pagal išsišakojusią architektūrą, nustatytą CILRS ir Roach metoduose. Vairavimo metu (angl. during execution) šaka, atitinkanti dabartinę aukšto lygio komandą, generuoja žemesnio lygio vairavimo komandas, o mokymo metu neaktyvios šakos užmaskuojamos ir prilyginamos nulinei reikšmei.

Tegul $X \in \mathbb{R}^{224\times224\times3}$ yra įvesties vaizdas iš priekinės kameros jutiklio. Agentas konvertuoja vaizdo įvesties X į $\mathbb{R}^2$ vektorių, kurį sudaro transporto priemonės akseleravimo vertė (angl. throttle value) ir posūkio kampo vertė (angl. steering value). Atitinkamai agentas yra reprezentuojamas toliau nurodyta lygtimi:

$$\widehat{\mathbf{a}}(X, u|\theta, \xi, \phi, \psi) := \sum_{i=0}^{n} c_i \phi_i(X, u|\theta, \xi, \phi, \psi), \tag{S.4}$$

kur $f_A(f_J(f_E(X|\theta), f_M(u|\xi)|\phi)|\psi)$ turi n veiksmo šakų ir $\phi_i(X, u|\theta, \xi, \phi, \psi)$ atitinka i^{th} veiksmo šakos išvestį. Kur,

- X yra įvesties vaizdas,
- f_E yra vaizdo enkoderis su parametrais θ , paruošiamojo mokymo būdu apmokytas atlikti VPR užduotį (t. y. SegVPR enkoderis),
- f_M yra matavimų enkoderio tinklas su parametrais ξ, o u yra matavimų vektorius (dabartinis greitis ir aukšto lygio komanda),
- f_J yra kitas neuroninio tinklo modulis su parametrais ϕ, kuris sujungia vaizdo ir matavimų reprezentacijas ir sumažina jų dydį,
- f_A yra veiksmų šakų modulis su parametrais ψ, kuris kiekvienai aukšto lygio komandai apskaičiuoja žemo lygio komandą,
- c_i yra vektorius, kuris išlaiko vieną iš n, veiksmų šakų, skirtų įvesties vaizdui X ir panaikina visas kitas veiksmų šakas.

Siekdami supaprastinti palyginimą su kitu agentu ir vadovaudamiesi kituose darbuose taikytu metodu, nuostolių funkciją naudojame kaip veiksmo nuostolių ir greičio prognozavimo reguliarizacijos sumą (angl. sum of action loss and a speed prediction regularisation),

$$\mathcal{L}_{Agent}(\theta, \xi, \phi, \psi) = \mathcal{L}_A(\theta, \xi, \phi, \psi) + \lambda_S \cdot \mathcal{L}_S, \tag{S.5}$$

kur veiksmo nuostoliai $\mathcal{L}_A$ yra lygūs L1 nuostoliams tarp eksperto veiksmo $\hat{\mathbf{a}}$ ir prognozuojamo veiksmo $\mathbf{a}$, kuris yra apskaičiuojamas pagal toliau nurodytą formulę:

$$\mathcal{L}_A = \|\hat{\mathbf{a}} - \mathbf{a}(X, u|\theta, \xi, \phi, \psi)\|_1, \tag{S.6}$$

o greičio prognozavimo reguliarizacija $\mathcal{L}_S$ tarp užfiksuoto greičio $\hat{s}$ ir prognozuojamo greičio s yra apskaičiuojama pagal toliau nurodytą formulę:

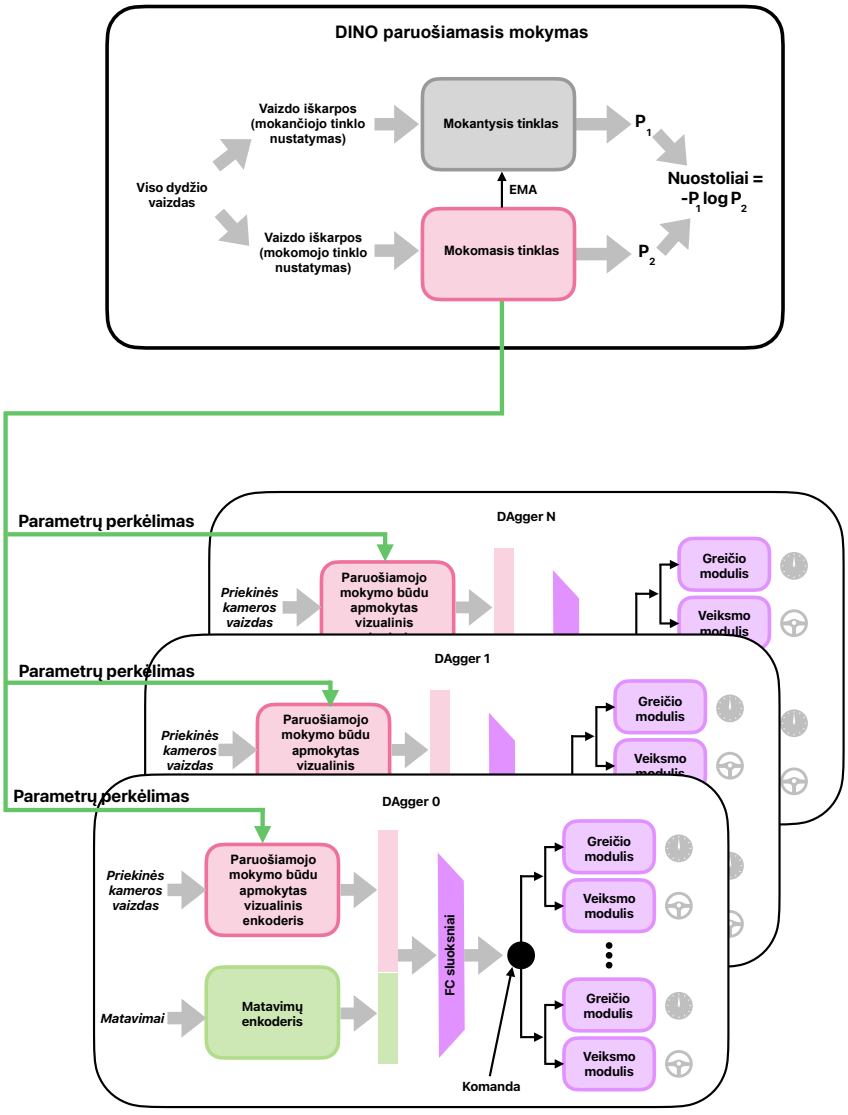
$$\mathcal{L}_S = |\hat{s} - s|. \tag{S.7}$$

Reguliarizacijos poveikis reguliuojamas skaliarine verte λ_s, kuri yra `1e-5`.

DINO PARUOŠIAMASIS MOKYMAS

Dauguma metodų yra pagrįsti prižiūrimu mokymu paremtu paruošiamuoju mokymu. Prižiūrimas mokymas daugiausia remiasi sužymėtais duomenimis (angl. labelled data), kai kiekvienam duomenų taškui yra priskiriama konkreti žymė (angl. label) arba kategorija. Šio metodo taikymas didesniu mastu gali būti brangus, nes anotaciją reikia atlikti rankiniu būdu. O savarankiškos priežiūros mokymas (angl. self-supervised learning) naudoja nežymėtus duomenis ir generuoja dirbtinius priežiūros signalus iš pačių duomenų. Siūlome naudoti savidistiliacijos be žymių (angl. self-distillation with no labels, DINO) metodą kaip paruošiamojo mokymo metodą, kad būtų pagerinamas agento gebėjimas prisitaikyti prie naujų, nematytų situacijų (kai susiduriama su kovariaciniu poslinkiu). Panašiai kaip ir VPR paruošiamojo mokymo metodu apmokyto agento atveju, siūlome atlikti agento vaizdo enkoderio paruošiamąjį mokymą naudojant DINO kaip paruošiamojo apmokymo metodą. Keliame hipotezę, kad gausus žymių naudojimas prižiūrimame paruošiamajame mokyme, pavyzdžiui, ImageNet klasifikacija, riboja modelio gebėjimą išmokti platų požymių spektrą, taigi ir generalizuoti naujose situacijose. Neseniai pristatytas DINO, savarankiškos priežiūros mokymo metodas, rodo gebėjimą mokytis platesnio spektro ir įvairesnių požymių nesiremiant konkrečiomis žymėmis. DINO taiko daugelio iškarpų (angl. multi-crop) mokymo metodą ir kontrastinių nuostolių funkciją (angl. contrastive loss), kad iš vaizdų be aiškių žymių išmoktų būdingą semantinę informaciją, ir taip parodo savarankiškos priežiūros mokymo efektyvumą užfiksuoti platesnį duomenų supratimą. DINO taip pat parodė, kad jai yra būdingas semantinės informacijos vaizde supratimas, kas yra naudinga įvairioms kompiuterinės regos užduotims, įskaitant autonominį vairavimą.

Remdamiesi VPR paruošiamojo mokymo būdu apmokyto agento struktūra, panašiai struktūruojame ir dabartinio pasiūlymo architektūrą. Šiame skyriuje pasiūlytą metodą vadiname DINO paruošiamojo mokymo metodu, o metodo rezultatą – DINO paruošiamojo mokymo metodu apmokytu agentu. DINO paruošiamojo mokymo metodu apmokyto agento kūrimas yra aprašomas toliau pateiktose dviejose dalyse. Pirmiausia aprašomas siūlomas vaizdo enkoderio paruošiamasis mokymas



S.4 pav.: Paveikslėlyje pavaizduota bendra DINO paruošiamojo mokymo metodo blokinė schema (viršuje), kurioje naudojama mokančiojo tinklo-mokomojo tinklo architektūra ir eksponentinis slenkantis vidurkis (angl. exponential moving average, EMA) mokančiojo tinklo parametrams atnaujinti iš mokomojo tinklo. Mokantysis tinklas ir mokomasis tinklas yra mokomi naudojant originalaus viso dydžio vaizdo iškarpas. Vėliau iliustruojamas parametrų perkėlimas, kad būtų galima mokyti atlikti ištisinio vairavimo užduotį (apačioje).

DINO metodu. Po paruošiamojo mokymo aprašomas paruošiamojo mokymo būdu apmokyto vaizdo enkoderio integravimo į agentą metodas ir mokymas atlikti autonominio vairavimo užduotį. Visas DINO paruošiamojo mokymo ir agento mokymo procesas yra pavaizduotas S.4 paveiksle.

Vaizdo enkoderio paruošiamasis mokymas naudojant DINO

Taikant inovatyvias mokymo paradigmas, savarankiškos priežiūros mokymas maksimaliai padidina turimų duomenų naudingumą. Tradicinio prižiūrimo mokymo metu daugiausiai dėmesio skiriama tiesioginiam konkrečios siekiamos atlikti užduoties mokymui, o savarankiškos priežiūros mokymo metodai yra optimizuoti atlikti pagalbines užduotis, kurios netiesiogiai padeda atlikti užsibrėžtą užduotį. Šio metodo efektyvumas priklauso nuo duomenų rinkinio dydžio, t. y. didesni paruošiamojo mokymo duomenų rinkiniai, derinami su tinkamomis savarankiškos priežiūros mokymosi paradigmomis, paprastai būna efektyvesni. Laikydamiesi šio principo, kaip paruošiamojo mokymo metodą taikome DINO. DINO taiko savarankiškos priežiūros mokymo sistemą, kuri mokosi iš ImageNet duomenų rinkinio, kuriame yra apie 1 milijonas vaizdų. Vietoje to, jog naudotų įprastą prižiūrimą klasifikavimą, DINO taiko du pagrindinius metodus: daugelio iškarpų mokymą (angl. multi-crop training) ir savidistiliavimą (angl. self-distillation).

Kaip ir kiti žinių distiliavimo metodai, DINO naudoja tinklus-dvynius (angl. twin networks), t. y. mokantįjį ir mokomąjį tinklus su vienodais parametrų kiekiais. Mokomasis tinklas g_{θ_s} su parametrais θ_s yra apmokytas imituoti jo mokančiojo tinklo analogo g_{θ_t} su parametrais θ_t išvestis. Gavę įvestį x, abu tinklai sugeneruoja K – dimensinius tikimybių pasiskirstymus (angl. dimensional probability distributions), atitinkamai žymimus g_{θ_t} ir θ_t. Tada šie pasiskirstymai yra apdorojami modifikuota softmax funkcija, kur temperatūros parametras kontroliuoja pasiskirstymo ryškumą. Mokomajame tinkle tikimybė P_s yra apskaičiuojama naudojant temperatūros parametrą τ_s, kaip parodyta S.8 lygtyje:

$$P_s(x)^{(i)} = \frac{\exp\left(g_{\theta_s}(x)^{(i)}/\tau_s\right)}{\sum_{k=1}^{K}\exp\left(g_{\theta_s}(x)^{(k)}/\tau_s\right)}. \tag{S.8}$$

Panašiai ir mokančiajame tinkle tikimybė P_t yra apskaičiuojama naudojant temperatūros parametrą τ_t, kaip parodyta S.9 lygtyje:

$$P_t(x)^{(i)} = \frac{\exp\left(g_{\theta_t}(x)^{(i)}/\tau_t\right)}{\sum_{k=1}^{K}\exp\left(g_{\theta_t}(x)^{(k)}/\tau_t\right)}. \tag{S.9}$$

Temperatūros valdymo parametrai yra sąlyginiai: $\tau_s > 0$, $\tau_s > 0$, ir iš pradžių yra nustatyti atitinkamai $0,1$ ir $0,04$.

Mokantysis tinklas yra mokomas kartu su mokomuoju tinklu, tačiau epochos (angl. epoch) metu jis yra sustabdomas. Vietoje to, naudojant pagreičio kodavimo metodą (angl. momentum encoder technique), eksponentinis slenkantis vidurkis yra kopijuojamas iš mokomojo tinklo į mokantįjį tinklą. Mokymo metu yra taikoma tokia atnaujinimo taisyklė:

$$\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s, \tag{S.10}$$

kurioje λ mokymo metu seka kosinuso grafiką nuo $0,996$ iki 1. Naudojant per epochą fiksuotą mokantįjį tinklą, mokymasis įvyksta minimizuojant kryžminę entropiją pagal mokomojo tinklo parametrus θ_s, kaip nurodyta toliau pateiktoje lygtyje:

$$\min_{\theta_s} H(P_t(x), P_s(x)), \tag{S.11}$$

kur $H(P_t, P_s) = -P_t \log P_s$.

Pasinaudodamas savarankiškos priežiūros privalumais, DINO naudoja daugelio iškarpų augmentacijos mokymą. Iš pradžių yra sudaromas kelių iškarpų V rinkinys, naudojant du nustatymus. Pirmuoju nustatymu sukuriamios dvi iškarpos, vadinamos visuotinėmis iškarpomis (angl. global views), pažymėtos x_1^g ir x_2^g, kurios yra 224×224 skiriamosios gebos, ir kurios apima daugiau kaip 50% vaizdo. Antruoju nustatymu sukuriamos kelios iškarpos, vadinamos lokaliomis iškarpomis, kurios yra 96×96 skiriamosios gebos ir kurios apima mažiau nei 50% vaizdo. Sukūrus iškarpas, visuotinės iškarpos yra perduodamos per mokantįjį tinklą, ir visos iškarpos, įskaitant visuotines ir lokalias iškarpas, yra perduodamos per mokomąjį tinklą. Atitinkamai yra naudojamas S.11 lygtyje minėtos nuostolių funkcijos modifikuotas variantas,

siekiant ją pritaikyti savarankiškos priežiūros aplinkoje tokiu būdu:

$$\min_{\theta_s} \sum_{x \in \{x_1^{g_1}, x_2^{g_2}\}} \sum_{x' \in V, x' \neq x} H(P_t(x), P_s(x')). \tag{S.12}$$

DINO moksliniuose tyrimuose pateikti darbai rodo jų metodų efektyvumą naudojant tiek konvoliucinius neuroninius tinklus, tiek transformerių architektūras. Mūsų įgyvendinimui ir eksperimentams, užuot mokę DINO modelį nuo nulio, naudojame jų konvoliucinio neuroninio tinklo variantą, pagrįstą ResNet50.

Duomenų rinkimas

Vadovaudamiesi Roach metodu, CARLA simuliatoriuje surinkome vairavimo duomenis, naudodami ekspertą demonstratorių, apmokytą sustiprinto mokymo (angl. reinforcement learning) būdu. Duomenų rinkinį sudaro 160 epizodų (12 valandų vairavimo duomenų) mokymo miestuose (angl. train towns) ir mokymo oro sąlygomis, laikantis Leaderboard lyginamojo standarto (angl. benchmark). Visi agentai buvo apmokyti pagal šį ir tą patį duomenų rinkinį.

Atskaitos metodai

Kadangi daugiausia dėmesio skiriame vaizdo enkoderio paruošiamajam mokymui, atskaitos agentai naudoja ResNet34 ir ResNet50 enkoderius, apmokytus paruošiamojo mokymo būdu naudojant ImageNet klasifikaciją. Dekoderis išlieka identiškas pasiūlytuose metoduose naudojamam dekoderiui. Sukūrėme du atskaitos agentus: BAR34IC (ResNet34) ir BAR50IC (ResNet50).

Metrikos

Norint kiekybiškai įvertinti ir palyginti skirtingų metodų efektyvumą, naudojame dvi pagrindines metrikas. Jos yra išvardytos ir apibrėžtos toliau:

- **Maršruto įveikimas:** Ši metrika parodo, kiek procentų maršrutų agentas sėkmingai įveikė, esant pasirinktam sąlygų deriniui.
- **Atstumo įveikimas:** Ši metrika parodo, kiek vidutiniškai procentų atstumo buvo įveikta per visus maršrutus, esant pasirinktam sąlygų deriniui.

Siekiant išplėsti palyginimą (kad jis apimtų ne tik supratimą, ar agentas įveikia maršrutus ir nuvažiuoja ilgesnius atstumus), mes įtraukiame papildomas smulkesnes metrikas. Šios metrikos kvestionuoja agentų elgsenos kokybę. Šios smulkesnės metrikos yra išvardytos toliau:

- **Susidūrimai su statiniais objektais (angl. Collision static):** susidūrimų su statiniais objektais, esančiais veiksmo vietoje (maršrute) (pvz., šviesoforų stulpeliais, medžiais, atitvarais, stulpais ir pan.), skaičius, normalizuotas vienam nuvažiuotam kilometrui.
- **Susidūrimai su pėsčiaisiais (angl. Collision pedestrian):** maršrute įvykusių susidūrimų su pėsčiaisiais skaičius, normalizuotas vienam nuvažiuotam kilometrui.
- **Susidūrimai su transporto priemonėmis (angl. Collision vehicle):** susidūrimų su transporto priemonėmis, kurios buvo sutiktos maršrute, skaičius, normalizuotas vienam nuvažiuotam kilometrui.
- **Raudono šviesoforo signalo pažeidimai (angl. Red light infraction):** atvejų, kai buvo pravažiuota per raudoną šviesoforo signalą, skaičius, normalizuotas vienam nuvažiuotam kilometrui.

EKSPERIMENTŲ REZULTATAI

Naudodami maršruto įveikimo ir atstumo įveikimo metrikas, atliekame VPR paruošiamojo mokymo metodu ir DINO paruošiamojo mokymo metodu apmokytų agentų vertinimą juos lyginant su BAR34IC ir BAR50IC atskaitos agentais. Papildomai atliekame pasiūlytų ir bazinių

atskaitos metodų palyginimą, naudodami smulkesnes metrikas – susidūrimus su statiniais objektais, susidūrimus su pėsčiaisiais, susidūrimus su transporto priemonėmis ir raudono šviesoforo signalo pažeidimus – siekdami įvertinti agentų vairavimo elgseną.

VIZUALINIO VIETOS ATPAŽINIMO PARUOŠIAMOJO MOKYMO EKSPERIMENTAI

Siekdami įvertinti siūlomo metodo, t. y. VPR paruošiamojo mokymo metodu apmokyto agento, privalumus, vertiname jo efektyvumą lyginant su mūsų baziniais atskaitos agentais. Nustatome kiekvieno agento pasiektus mūsų pagrindinių metrikų, t. y. maršruto įveikimo ir atstumo įveikimo, rezultatus. Stebint kiekvieno lyginamo metodo geriausias DAgger iteracijas, VPR paruošiamojo mokymo būdu apmokytas agentas pirmauja pagal maršrutų įveikimą lyginant su mūsų baziniais atskaitos agentais. Važiuodamas mokymo aplinkos sąlygomis, VPR paruošiamojo mokymo būdu apmokytas agentas įveikia atitinkamai 4% ir $16,66\%$ didesnį kiekį maršrutų, palyginti su BAR50IC ir BAR34IC atskaitos agentais. Važiuodamas aplinkose, kurių nėra mokymo duomenyse (t. y. naujos aplinkos sąlygomis), VPR paruošiamojo mokymo metodu apmokytas agentas įveikia atitinkamai $7,05\%$ ir $10,90\%$ didesnį kiekį maršrutų nei BAR50IC ir BAR34IC atskaitos agentai. Aprašytus rezultatus galima pamatyti S.1 lentelėje.

S.1 lentelė: Vairavimo agentų geriausi maršruto įveikimo rezultatai (%) mokymo ir naujomis (testavimo) sąlygomis visose DAgger iteracijose.

Paruošiamojo mokymo metodas	**Mokymo miestas ir oras**	**Naujas miestas ir oras**
BAR34IC	64.67 ± 2	49.35 ± 9
BAR50IC	77.33 ± 4	53.20 ± 1
VPR paruošiamasis mokymas (mūsų)	**81.33** ± 4	**60.25** ± 2

Norėdami įvertinti pasiektus rezultatus pagal antrąją pagrindinę metriką, t. y. atstumo įveikimą, taip pat vertiname tą pačią anksčiau stebėtą DAgger iteraciją. Mokymo aplinkos sąlygomis, VPR paruošiamojo mo-

S.2 lentelė: Vairavimo agentų geriausi atstumų įveikimo rezultatai (%) mokymo ir naujomis (testavimo) sąlygomis visose DAgger iteracijose.

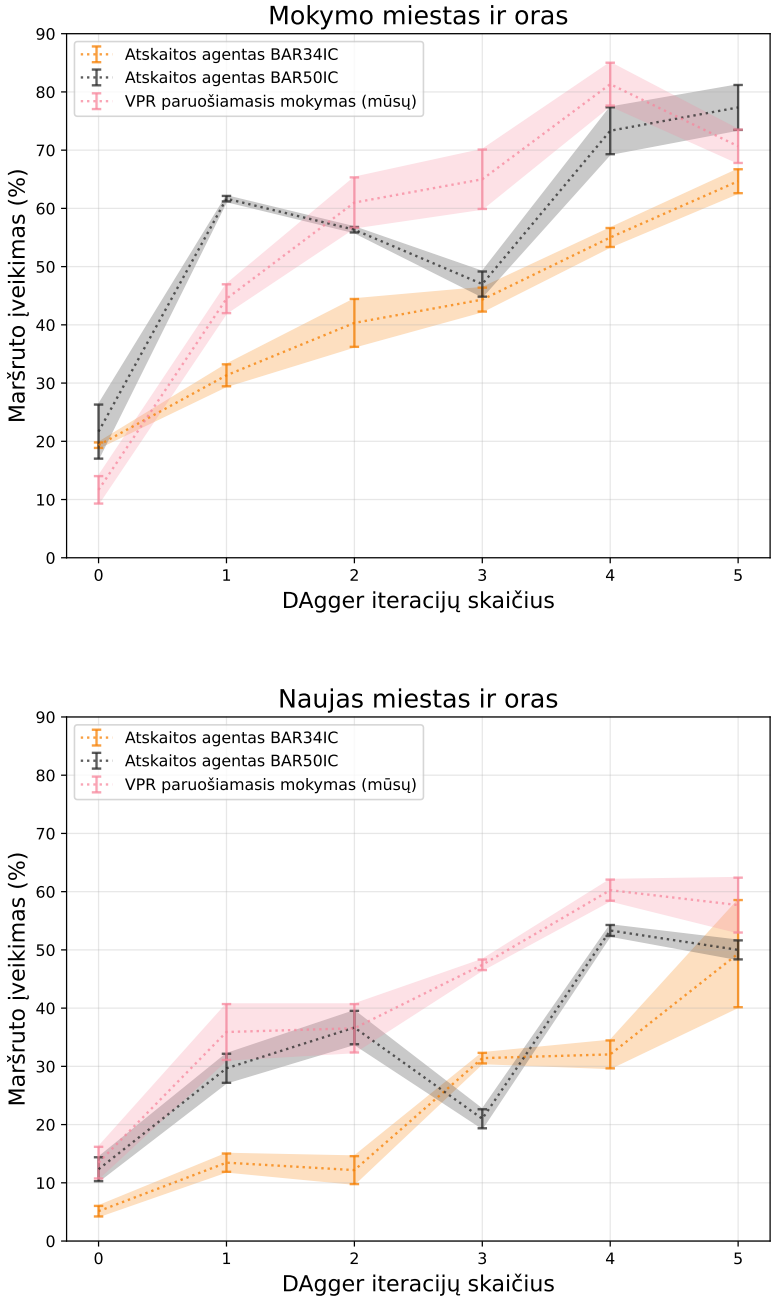
Paruošiamojo mokymo metodas	**Mokymo miestas ir oras**	**Naujas miestas ir oras**
BAR34IC	79.02 ± 2	75.75 ± 7
BAR50IC	89.36 ± 2	72.23 ± 6
VPR paruošiamasis mokymas (mūsų)	**91.97** ± 3	**86.01** ± 0

kymo metodu apmokytas agentas nuvažiuoja atitinkamai vidutiniškai $2,61\%$ ir $12,95\%$ toliau, o naujos aplinkos sąlygomis VPR paruošiamojo mokymo būdu apmokytas agentas nuvažiuoja atitinkamai vidutiniškai $13,78\%$ ir $10,26\%$ toliau, lyginant su BAR50IC ir BAR34IC atskaitos agentais. Šie rezultatai pateikti S.2 lentelėje.

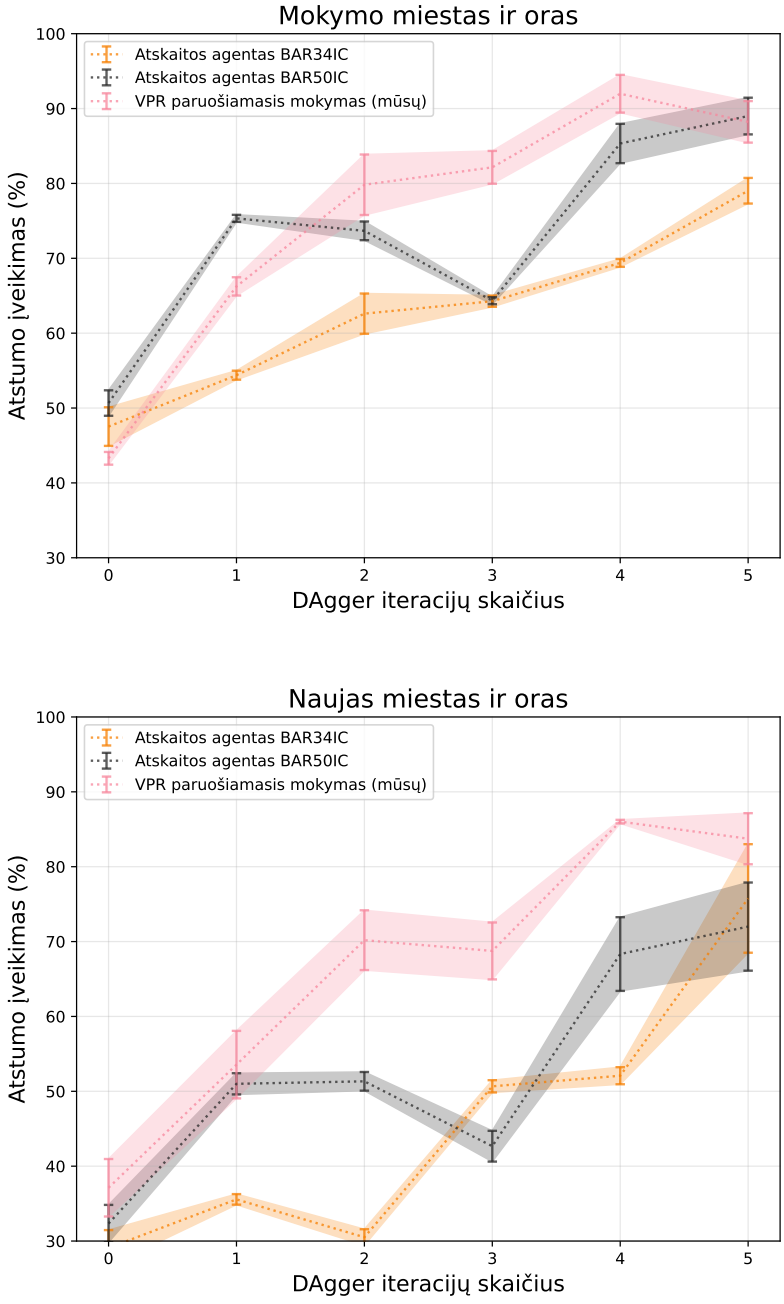
Papildomai atskleidžiame lyginamų metodų ir kiekvienos atsitiktinės inicializacijos efektyvumą kiekvienoje DAgger iteracijoje. Tiek mokymo, tiek naujų aplinkų sąlygomis VPR paruošiamojo mokymo būdu apmokytas agentas stabiliai konverguoja į aukščiausią efektyvumą jau ketvirtosios iteracijos metu, palyginti su atskaitos metodais. Tai galima pastebėti S.5 paveikslėlyje. Ta pati tendencija taip pat pastebėtina ir atstumo įveikimo metrikos atžvilgiu, kaip yra atskleidžiama S.6 paveikslėlyje.

Apskaičiavus eksperimentų, atliktų tiek mokymo aplinkoje, tiek naujoje aplinkoje, rezultatus, galima įvertinti kovariacinį poslinkį. Mūsų pateikti duomenys rodo, kad VPR paruošiamojo mokymo būdu apmokytas agentas ne tik pasiekia geresnius rezultatus maršrutų įveikimo ir ilgesnių atstumų įveikimo srityse, bet demonstruoja didesnį atsparumą, lyginant su atskaitos metodais, ypač nematytose sąlygose. Tai rodo geresnes generalizacijos galimybes, ypač kai vaizdo enkoderis yra apmokomas už įprastos užduoties (vaizdų klasifikavimas ImageNet duomenų rinkinyje) ribų. Taigi, šis metodas leidžia užtikrinti didesnį atsparumą kovariaciniui poslinkiui.

Vienas iš pastebimų VPR paruošiamojo mokymo metodo ir atskaitos metodų skirtumų, darančių įtaką rezultatams, yra mokymo užduotis.



S.5 pav.: Agentų maršruto įveikimas (%) naudojant Leaderboard lyginamąjį standartą mokymo sąlygomis (viršuje) ir testavimo sąlygomis (apačioje), įvertintas tris kartus su skirtingais atsitiktinės inicializacijos parametrais ir pavaizduotas kartu su efektyvumo vidurkiu.



S.6 pav.: Agentų atstumo įveikimas (%) naudojant Leaderboard lyginamąjį standartą mokymo sąlygomis (viršuje) ir testavimo sąlygomis (apačioje), įvertintas tris kartus su skirtingais atsitiktinės inicializacijos parametrais ir pavaizduotas kartu su efektyvumo vidurkiu.

VPR paruošiamojo mokymo metodas pasitelkia trejeto nuostolių funkciją regimajam vietos atpažinimui, kaip mokymo užduotį įtraukiant semantinį segmentavimą, priešingai nei atskaitos metodai, kurie remiasi klasifikavimo nuostolių funkcija. Toks nustatymas teikia pirmenybę vietų supratimui kintančiomis oro ir apšvietimo sąlygomis, o ne unikalių objektų klasių kodų formavimui. Remiantis rezultatais, toks pasirinkimas leidžia sukurti vaizdo enkoderį su pranašesne parametrų inicializacija (angl. weight initialisation), kuriuos galima perkelti į autonominio vairavimo užduotį.

Kitas VPR paruošiamojo mokymo metodo ir atskaitos metodų skirtumas, turintis įtakos pateiktiems rezultatams, yra duomenų rinkinys. VPR paruošiamojo mokymo metodo duomenų rinkinys yra sudaromas iš vaizdų, kurie yra susiję su vairavimo užduotimi. O atskaitos metodai remiasi ImageNet duomenų rinkiniu, kurį sudaro nereikšmingos vaizdų klasės, pavyzdžiui, katės, šunys, įvairūs objektai ir t. t. Šį skirtumą priskiriame prie vieno iš įtakingiausių, nes susidūrimas su susijusių duomenų pasiskirstymu yra esminis mašininio mokymo efektyvumui.

DINO PARUOŠIAMOJO MOKYMO AUTONOMINIAM VAIRAVIMUI EKSPERIMENTAI

Siekdami įvertinti siūlomo DINO paruošiamojo mokymo metodu apmokyto agento efektyvumą, atliekame lyginamąją analizę su dviem atskaitos metodais ir anksčiau pristatytu VPR paruošiamojo mokymo metodu apmokytu agentu. Šią analizę atliekame stebėdami įvardytų metodų efektyvumą pagal mūsų pagrindinius rodiklius – maršruto įveikimą ir atstumo įveikimą.

DINO paruošiamasis mokymas pranoksta visus metodus naujose aplinkose pagal didžiausio kiekio maršrutų įveikimo metriką. Mes palyginame kiekvieno metodo geriausias DAgger iteracijas ir rezultatai rodo, kad DINO paruošiamojo mokymo būdu apmokytas agentas įveikia vidutiniškai $1,93\%$ didesnį skaičių maršrutų nei anksčiau pasiūlytas VPR paruošiamojo mokymo būdu apmokytas agentas. Lyginant su BAR34IC ir BAR50IC atskaitos agentais, paruošiamasis mokymas naudojant DINO metodą pasiekia atitinkamai vidutiniškai $12,83\%$ ir $8,98\%$ didesnį

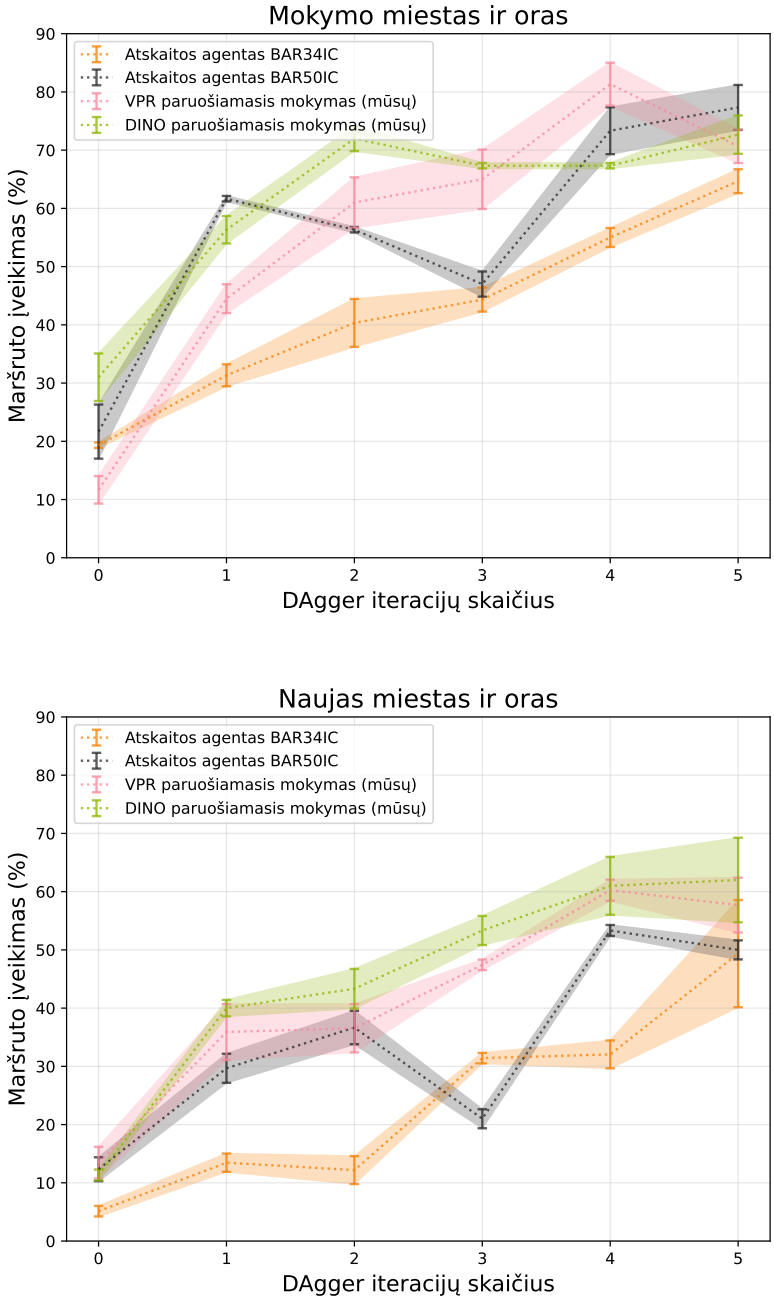
skaičių įveiktų maršrutų. Tačiau mokymo aplinkose pagal gebėjimą įveikti maršrutą DINO paruošiamojo mokymo būdu apmokytas agentas atsilieka $8,66\%$ nuo VPR paruošiamojo mokymo būdu apmokyto agento, o pagal efektyvumą priartėja prie BAR50IC atskaitos agento. Rezultatai pateikiami S.3 lentelėje. Pagal atstumo įveikimo metrikos rezultatus naujos aplinkos sąlygomis DINO paruošiamojo mokymo būdu apmokytas agentas efektyvumu lenkia BAR34IC ir BAR50IC atskaitos agentus atitinkamai vidutiniškai $6,92\%$ ir $10,44\%$. O mokymo aplinkos sąlygomis DINO paruošiamasis mokymas atsilieka $3,32\%$ lyginant jį su geresnį efektyvumą pasiekiančiu atskaitos metodu – t. y. BAR50IC. Lyginant su VPR paruošiamojo mokymo būdu apmokytu agentu, DINO paruošiamojo mokymo būdu apmokytas agentas įveikia trumpesnius atstumus. Rezultatai pateikiami S.4 lentelėje.

S.3 lentelė: Vairavimo agentų geriausi maršruto įveikimo rezultatai (%) mokymo ir naujomis (testavimo) sąlygomis visose DAgger iteracijose.

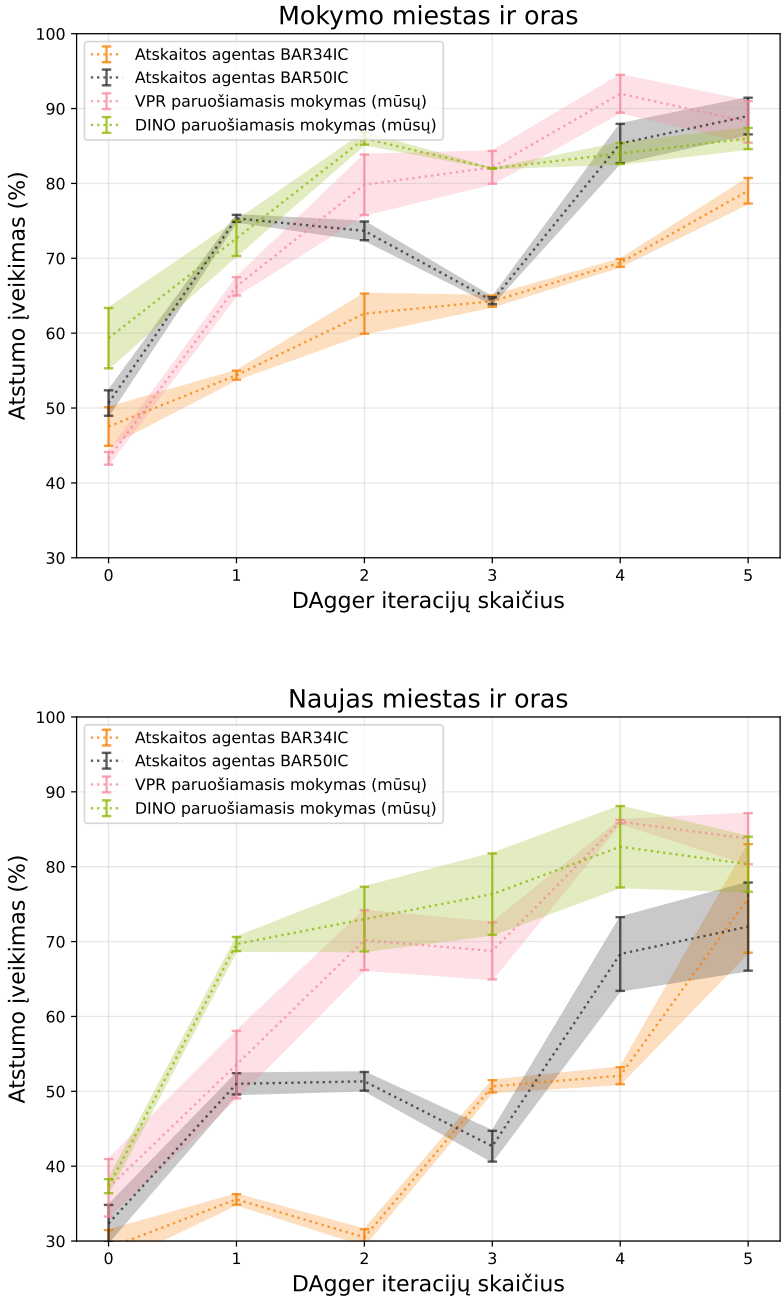
Paruošiamojo mokymo metodas	**Mokymo miestas ir oras**	**Naujas miestas ir oras**
BAR34IC	64.67 ± 2	49.35 ± 9
BAR50IC	77.33 ± 4	53.20 ± 1
VPR paruošiamasis mokymas	**81.33** ± 4	60.25 ± 2
DINO paruošiamasis mokymas (mūsų)	72.67 ± 3	**62.18** ± 7

S.7 ir S.8 paveikslėliuose taip pat pateikiame visų DAgger iteracijų abiejų pagrindinių metrikų rezultatus. Kaip matyti iš mūsų rezultatų, pažymėtina, kad DINO paruošiamojo mokymo metodas konverguoja į geresnius rezultatus anksčiau nei kiti metodai ir pirmauja maršruto įveikimo metrikoje naujos aplinkos sąlygomis. Palyginti su atskaitos metodais, DINO paruošiamojo mokymo metodas generuoja reikšmingai geresnius rezultatus.

DINO paruošiamojo mokymo būdu apmokytas vaizdo enkoderis ir lyginti atskaitos paruošiamojo mokymo būdu apmokyti enkoderiai turi reikšmingą bendrą bruožą – paruošiamojo mokymo duomenų rinkinys yra ImageNet. Nepaisant to, rezultatai rodo reikšmingą pagerėjimą



S.7 pav.: Agentų maršruto įveikimas (%) naudojant Leaderboard lyginamąjį standartą mokymo sąlygomis (viršuje) ir testavimo sąlygomis (apačioje), įvertintas tris kartus su skirtingais atsitiktinės inicializacijos parametrais ir pavaizduotas kartu su efektyvumo vidurkiu.



S.8 pav.: Agentų atstumo įveikimas (%) naudojant Leaderboard lyginamąjį standartą mokymo sąlygomis (viršuje) ir testavimo sąlygomis (apačioje), įvertintas tris kartus su skirtingais atsitiktinės inicializacijos parametrais ir pavaizduotas kartu su efektyvumo vidurkiu.

S.4 lentelė: Vairavimo agentų geriausi atstumų įveikimo rezultatai (%) mokymo ir naujomis (testavimo) sąlygomis visose DAgger iteracijose.

Paruošiamojo mokymo metodas	**Mokymo miestas ir oras**	**Naujas miestas ir oras**
BAR34IC	79.02 ± 2	75.75 ± 7
BAR50IC	89.36 ± 2	72.23 ± 6
VPR paruošiamasis mokymas	**91.97** ± 3	**86.01** ± 0
DINO paruošiamasis mokymas (mūsų)	86.04 ± 1	82.67 ± 6

pagal maršruto įveikimo ir atstumo įveikimo metrikas. Remiantis empiriniu vertinimu, šį pagerėjimą galima priskirti pasirinktam mokymo metodui ir nuostolių funkcijoms, kurios yra naudojamos paruošiamojo mokymo naudojant DINO metodą metu. Šis skirtumas leidžia patobulinti mokymo procesą ir pasiekti geresnių vaizdinių požymių kodavimą, kurie yra naudingi autonominio vairavimo užduočiai, taigi įrodo, kad vaizdų klasifikavimas (kaip paruošiamojo mokymo metodas) yra praradęs aktualumą.

Kita problema, kurią, kaip matyti iš rezultatų, sprendžia DINO paruošiamojo mokymo metodas – persimokymas (angl. over-fitting). Pagrindiniai metodai rodo daug didesnį efektyvumo atotrūkį pereinant nuo mokymo aplinkos sąlygų į naujos aplinkos sąlygas, lyginant su DINO paruošiamojo mokymo metodu ir jo rezultatais. Šis atotrūkis gali būti susidaręs dėl stipraus persimokymo (angl. over-fit), nes DINO paruošiamojo mokymo metodas rodo didesnę generalizaciją esant naujos aplinkos sąlygoms, bet tuo pačiu pasiekia prastesnus maršruto įveikimo rezultatus mokymo aplinkos sąlygose nei BAR50IC atskaitos metodas.

Apibendrinant šį metodą, pabrėžiame, kad geresnis paruošiamasis mokymas lemia geresnį vairavimo užduoties mokymą, ypač lyginant su įprastais paruošiamojo mokymo metodais.

Siekdami atkreipti dėmesį į tai, kaip lyginami metodai geba generalizuoti, minėtas metrikas apskaičiuojame remdamiesi tuo, kaip agentai važiuoja naujos aplinkos sąlygomis, o ne mokymo aplinkos sąlygomis. DINO paruošiamojo mokymo būdu apmokytas agentas pasiekia nuoseklius rezultatus pagal visas keturias metrikas – t. y. jis mažiausiai susiduria su statiniais objektais, pėsčiaisiais, transporto priemonėmis ir atlieka mažiausiai raudono šviesoforo signalo pažeidimų. S.5 lentelėje atskleidžiame šio vertinimo skaitmenines reikšmes. VPR paruošiamojo mokymo metodu apmokytas agentas įrodo esąs sėkmingas pagal pagrindinius rodiklius, nes įveikia daugiausia maršrutų ir pasiekia tolimesnius atstumus. Tačiau VPR paruošiamojo mokymo metodu apmokytam agentui nepavyksta pasiekti nuoseklumo atliekant mažesnį kiekį klaidų pagal smulkesnes metrikas, vertinančias agento elgesio kokybę, priešingai nei DINO paruošiamojo mokymo būdu apmokytas agentas ir BAR50IC atskaitos agentas. Manytina, kad tai gali būti susiję su vertinant VPR paruošiamojo mokymo būdu apmokyto agento elgseną anksčiau nustatytu persimokymu. Šie rezultatai dar kartą patvirtina DINO paruošiamojo mokymo būdu apmokyto agento geresnį generalizacijos gebėjimą, kai yra vairuojama naujose aplinkose.

S.5 lentelė: Lyginamų paruošiamojo mokymo metodų susidūrimų ir pažeidimų dažnis, normalizuotas pagal nuvažiuotą atstumą vienam kilometrui.

Paruošiamojo mokymo metodas	**Susidūrimai su statiniais objektais**	**Susidūrimai su transporto priemonėmis**	**Susidūrimai su pėsčiaisiais**	**Raudono šviesoforo signalo pažeidimai**
BAR34IC	0.22	0.87	0.08	1.13
BAR50IC	0.15	0.73	0.05	0.95
VPR paruošiam--asis mokymas	0.24	0.82	0.06	1.03
DINO paruošia--masis mokymas	**0.13**	**0.49**	**0.02**	**0.66**

1. Mūsų tyrimas rodo, kad vaizdo enkoderio paruošiamasis mokymas atlikti VPR užduotį naudojant trejeto nuostolių funkciją pagerina autonominio vairavimo efektyvumą, lyginant su įprastiniu ImageNet grindžiamu paruošiamuoju mokymu. Konkrečiai, pasitelkiant trejeto nuostolių funkcija pagrįstą paruošiamąjį mokymą yra pasiekiama geresnių rezultatų, palyginti su vertintais atskaitos agentais: lyginant su BAR50IC atskaitos agentu, maršruto įveikimas pagerėjo $7,05\%$, atstumo įveikimas – $13,78\%$, o lyginant su BAR34IC atskaitos agentu, maršruto įveikimas pagerėjo $10,90\%$, atstumo įveikimas – $10,26\%$. Šie rezultatai rodo, kad konkrečiai užduočiai skirtas paruošiamasis mokymas yra efektyvus siekiant pagerinti imitacinio mokymo rezultatus autonominio vairavimo srityje.

2. Mūsų eksperimentiniai rezultatai rodo, kad paruošiamasis mokymas taikant DINO metodą pagerina agento gebėjimą generalizuoti anksčiau nematytose aplinkose, palyginti su atskaitos metodais. Konkrečiai, DINO paruošiamojo mokymo metodu apmokytas vaizdo enkoderis, palyginti su BAR50IC atskaitos agentu, pasiekia $8,98\%$ geresnius maršruto įveikimo ir $10,44\%$ geresnius atstumo įveikimo rezultatus, o palyginti su BAR34IC atskaitos agentu, pasiekia $12,83\%$ geresnius maršruto įveikimo ir $6,92\%$ geresnius atstumo įveikimo rezultatus. Šie rezultatai išryškina DINO potencialą gerokai padidinti imitaciniu mokymu pagrįsto autonominio vairavimo patikimumą.

3. Lygindami agentus, apmokytus paruošiamojo mokymo būdu taikant VPR ir DINO metodus, formuojame išvadą, kad DINO metodu apmokytas agentas pasižymi geresnėmis generalizacijos savybėmis. DINO paruošiamojo mokymo metodu apmokytas agentas pasiekia $1,93\%$ geresnius maršruto įveikimo rezultatus ir demonstruoja pastebimai geresnius susidūrimų išvengimo ir eismo taisyklių laikymosi rezultatus. Konkrečiai, vienam nuvažiuotam kilometrui DINO metodu apmokytas agentas atlieka $0,11$ mažiau susidūrimų su statiniais objektais, $0,33$ mažiau susidūrimų su transporto priemonėmis, $0,04$ mažiau susidūrimų su pėsčiaisiais

ir $0,37$ mažiau raudono šviesoforo signalo pažeidimų, palyginti su VPR paruošiamojo mokymo metodu apmokytu agentu.

Shubham Anoop Juneja
Investigation of Pre-training in Imitation Learning-based Autonomous Driving
Doctoral Dissertation
Natural Sciences
Informatics (N 009)
Thesis Editor: Zuzana Šiušaitė

Shubham Anoop Juneja
Paruošiamojo mokymo tyrimas imitaciniu mokymu paremtame autonominiame vairavime
Daktaro disertacija
Gamtos mokslai
Informatika (N 009)
Santraukos redaktorius: Justas Rupšys

Vilnius University Press
9 Saulėtekio Ave., Building III, LT-10222 Vilnius
Email: info@leidykla.vu.lt, www.leidykla.vu.lt
bookshop.vu.lt, journals.vu.lt
Print run of 20 copies