UNIVERSITY OF TRENTO

Department of Information Engineering and Computer Science

Doctoral Program in Information and Communication Technology

PhD Dissertation

Cognitively Guided Modeling of Visual Perception in Intelligent Vehicles

Alice Plebe

Advisor: Prof. Mauro Da Lio

March 2021

Abstract

This work proposes a strategy for visual perception in the context of autonomous driving. Despite the growing research aiming to implement self-driving cars, no artificial system can claim to have reached the driving performance of a human, yet. Humans—when not distracted or drunk—are still the best drivers you can currently find. Hence, the theories about the human mind and its neural organization could reveal precious insights on how to design a better autonomous driving agent.

This dissertation focuses specifically on the perceptual aspect of driving, and it takes inspiration from four key theories on how the human brain achieves the cognitive capabilities required by the activity of driving. The first idea lies at the foundation of current cognitive science, and it argues that thinking nearly always involves some sort of mental simulation, which takes the form of imagery when dealing with visual perception. The second theory explains how the perceptual simulation takes place in neural circuits called convergence-divergence zones, which expand and compress information to extract abstract concepts from visual experience and code them into compact representations. The third theory highlights that perception—when specialized for a complex task as driving—is refined by experience in a process called perceptual learning. The fourth theory, namely the free-energy principle of predictive brains, corroborates the role of visual imagination as a fundamental mechanism of inference.

In order to implement these theoretical principles, it is necessary to identify the most appropriate computational tools currently available. Within the consolidated and successful field of deep learning, I select the artificial architectures and strategies that manifest a sounding resemblance with their cognitive counterparts. Specifically, convolutional autoencoders have a strong correspondence with the architecture of convergence-divergence zones and the process of perceptual abstraction. The free-energy principle of predictive brains is related to variational Bayesian inference and the use of recurrent neural networks. In fact, this principle can be translated into a training procedure that learns abstract representations predisposed to predicting how the current road scenario will change in the future.

The main contribution of this dissertation is a method to learn conceptual representations of the driving scenario from visual information. This approach forces a semantic internal organization, in the sense that distinct parts of the representation are explicitly

associated to specific concepts useful in the context of driving. Specifically, the model uses as few as 16 neurons for each of the two basic concepts here considered: vehicles and lanes. At the same time, the approach biases the internal representations towards the ability to predict the dynamics of objects in the scene. This property of temporal coherence allows the representations to be exploited to predict plausible future scenarios and to perform a simplified form of mental imagery.

In addition, this work includes a proposal to tackle the problem of opaqueness affecting deep neural networks. I present a method that aims to mitigate this issue, in the context of longitudinal control for automated vehicles. A further contribution of this dissertation experiments with higher-level spaces of prediction, such as occupancy grids, which could conciliate between the direct application to motor controls and the biological plausibility.

Acknowledgements

My sincerest gratitude goes to my advisor Mauro Da Lio, who believed in me and never wavered in his support throughout my whole PhD. I will always be thankful to Mauro for his relentless dedication, and for giving me the chance to travel the world and to grow both as a researcher and as a person.

I would also like to extend my deepest appreciation to Julian Kooij, who tutored me during the collaboration with the Intelligent Vehicles group from TU Delft. The time spent working with Julian has been invaluable. He made me feel welcome during the time I was in Delft, and he has provided me with insightful guidance even after the end of my visit, for which I am truly grateful.

Contents

Chapter 1

Introduction

1.1 Research Problem and Motivation

Modern society has always considered the development of fully autonomous vehicles a coveted achievement. The primary purpose of this research field is safety for all road users, as recommended by several governmental transportation institutions worldwide [63]. Road safety is anything but a minor problem: in 2018, the World Health Organization reported that road traffic injuries are the leading cause of death for people between 5 and 29 years old [165]. This suggests that the mitigation of traffic accidents could be one of the most beneficial outcomes expected from artificial intelligence and automation. A key aspect is that, in the United States, only 2% of vehicle crashes are due to technical failures; the rest is attributable to human drivers. Among the major causes of accidents are inattention, reckless driving, illegal maneuvers, the influence of alcohol or drugs, and tiredness [217]. Self-driving cars would be clearly immune to all the risky factors depending on human drivers.

The research on autonomous driving has a long history that dates back to the early 1950s [167], but it has become a reality—at a surprising fast pace—no longer than a decade ago [103]. While most of the components of a self-driving system (such as sensors) have improved at the typical rate of technological progress without any specific crucial innovations, the impressive advances have been mainly fueled by the emerging deep artificial neural networks [91, 207, 132].

Despite the remarkable technological progress, one of the main challenges the research has to face is how to demonstrate that self-driving vehicles are safer than human drivers. This is a non-trivial problem, for several reasons. Firstly, a prominent study [114] proved that a statistically significant evidence of the reliability of an autonomous driving agent would require billions of miles of test driving, which is not feasible in practice. As of February 2020, Tesla has collected a total of 3 billion miles[1] driven in autopilot, since the

[1] Video presentation by Andrej Karpathy, director of AI at Tesla (`https://youtu.be/hx7BXih7zx8`)

release of the first autopilot version in 2015. This mileage is clearly not enough, especially if each new software release requires to be tested from scratch. In addition, a proof of reliability is more challenging when key components of the system are implemented with artificial neural networks. It is well known neural networks suffer from "the black box problem", i.e., it is extremely difficult to explain how they work or why a particular input produces that specific output.

Above all, a crucial source of uncertainty and malfunctioning comes from the perception and understanding of the road environment [32]. In fact, this is one of the successful fields of application of deep neural models, which have quickly become the method of choice for perception of driving scenarios [18, 129]. Nonetheless, perception remains a major obstacle towards fully autonomous vehicles. The core of this issue could be identified in the narrow conception of "perception" usually assumed in autonomous driving, which lacks a fundamental aspect: gathering knowledge about objects and events in the environment oriented to the planning of future actions [145, 107]. Hence, perception is not a mere elucidation of objects in the world but the detection of action possibilities. My research deals precisely with the perception aspect of autonomous driving.

1.2 Cognitive Underpinning of the Research

My research relies, first and foremost, on one of the classical cornerstones of artificial intelligence: drawing inspiration from human cognition to design similar intelligent behaviors in artificial systems. This strategy fits well in the case of autonomous vehicles because, even considering the death toll from traffic accidents, humans are still the best drivers you can currently find. To date, no autonomous driving system can claim to reach the performance of an experienced—and sober—human driver. The human ability of driving is especially remarkable, given that vehicles are technological artifacts controlled by interfaces that are totally extraneous to the natural human motion control. Nonetheless, humans learn to drive quickly and excellently.

The idea of drawing inspiration from human cognition has been long ignored in the development of autonomous driving. In fact, the research has gradually consolidated a fixed approach over the years: a general structural and functional architecture closely derived from engineering practices. The typical architecture is made of many separate and basically independent modules organized in a hierarchical fashion. The modular decomposition is often applied recursively, so that perception—besides being independent from higher modules—is split into a hierarchy of modules, e.g. lower level preprocessing, segmentation, and object description. This approach has become so well established that the architectures of most autonomous systems have hardly changed through the years. Suffice it to say that the Ohio State University successfully competed in the 2007 DARPA Urban Challenge with an autonomous vehicle having almost the same overall architecture of a demo developed in 1996 [167]. Human cognition does not work this way, both in general and in the specific context of driving.

My research aims to develop intelligent behaviors for an artificial driving agent by taking inspiration from the neurocognition of human driving. My work originates from the European H2020 project *Dreams4Cars*[2] [183, 43, 44, 42]. The philosophy of Dreams4Cars is to take advantage of the computational solutions that the human brain has implemented for the complex task of driving. While Dreams4Cars applies this philosophy to the autonomous driving system as a whole, my research applies the idea specifically to visual perception. One of the main paradigm shift of Dreams4Cars is to conceive the artificial driving agent as a "co-driver", able to cooperate with the human driver. The interweaving between human and artificial in Dreams4Cars is pushed even further: the artificial agent should inherits the broad structure of the human motor control strategy, as known by the state-of-the-art in cognitive science. This means to move away from the classic *sense-think-act* paradigm adopted by most autonomous driving approaches. This paradigm implies a sharp separation among the perception system, the software that determines the agent's behavior, and the software that executes the selected action.

It is important to note, however, that vehicles are not biological bodies, and the hardware is not the brain. In fact, the engineering practice of modular decomposition can provide highly desirable features, like the reduction of complexity of every single module, or the decoupling of possible sources of failure. Similarly, there are algorithms that may be far different from brain computations but are very effective on silicon processors. Therefore, my research aims to find a compromise between the adoption of technologies that are well consolidated and the inspiration from human neurocognition.

1.3 Organization of the Dissertation

The presented dissertation is organized into nine chapters. It starts by illustrating what current cognitive science and neuroscience can reveal about how the human brain realizes the ability of driving. Chapter 2 addresses in detail the theories related to the processes involved in visual perception, and it focuses on a number of key points that seem well suited to be translated into a computational implementation.

The following chapter analyzes the role of deep learning in the development of artificial systems. Deep learning has turned computer vision from an insurmountable obstacle for autonomous driving to a challenging but feasible task. Therefore, it is almost mandatory to look at deep neural models when implementing solutions for artificial perception. Chapter 3 briefly reviews deep learning in general, then it points to some models that seem the best counterparts of the neurocognitive processes identified in Chapter 2 as fundamental for perception in the driving task.

Chapter 4 surveys the applications of deep learning for autonomous driving, especially in the context of visual perception. It also addresses the crucial aspect of the availability

[2] `www.dreams4cars.eu`

of appropriate datasets to train deep neural models for autonomous driving.

Chapters from 5 to 8 present the core of my work. Firstly, Chapter 5 presents an initial contribution in the direction of tackling the black box problem affecting any complex deep neural network. Chapter 6 describes the first group of models for visual perception, which work in a static context of single frames. Then, Chapter 7 presents the second group of perceptive models, this time taking into account the temporal dimension and processing temporal sequences of driving scenes. Furthermore, Chapter 8 presents a work in progress started during my research collaboration with the Intelligent Vehicles group from TU Delft: the work aims to extend perception to higher-level forms of representation oriented to motor actions. Lastly, Chapter 9 draws the conclusions of my research activity, analyzing the limitations and the potential future developments.

Chapter 2

Perception in Human Drivers

The presented work is motivated by the fact that humans are still the best driver one can currently find. Therefore, it is reasonable to expect that the functioning of the human brain could provide important cues on how to design autonomous driving systems. To fully commit to this idea, it is necessary to understand what kind of processes the brain executes when driving.

Driving is no different from any other high-level cognitive behavior. Unfortunately, the current understanding of how the brain enacts these behaviors is vague, often controversial, and short of detail. Nonetheless, there are countless theories trying to progress the understanding of the mind and the brain, and it is easy to lose track in this vast body of research. By taking advantage of the cooperation with the Dreams4Cars project, I have collected some suggestions to investigate and, then, I have identified the most promising cognitive principles and ideas that could help the design of an autonomous driving agent.

This chapter collects four key theories on how the human brain achieves the cognitive capabilities related to the activity of driving. Each section of the chapter analyzes one of the four proposals. The first idea lies at the foundation of modern cognitive science, and it argues that thinking nearly always involves some sort of *mental simulation*, which takes the form of imagery when dealing with visual perception. Then, moving from cognition to neuroscience, a second proposal could explain how the perceptual simulation takes place in neural circuits called *convergence-divergence zones*, which project external stimuli into high-level representations—but also the other way round, from top activations down to reconstructions of stimuli. In between cognition and neuroscience, the third idea highlights that perception, when specialized for a complex task as driving, is refined by experience with a process called *perceptual learning.* Lastly, I include a theory that in the last decade has received vast popularity in cognitive science: the *free-energy principle of predictive brains.* In the context of perception in the driving activity, this theory corroborates the role of visual imagination as a fundamental mechanism of inference.

2.1 Simulation Theory of Cognition

The ability to drive is one of the many specialized sensorimotor behaviors of humans, along with a variety of activities such as walking, playing the piano, handwriting, or snowboarding. Cognitive science has proposed several theories about how these complex behaviors are realized and which role has perception in them. Among these proposals, the *simulation theory of cognition* constitutes a major turning point of cognitive science of the last twenty years, and it is currently considered the most prominent position regarding the role of perception.

In a nutshell, the simulation theory argues that the process of thinking—most of the time, if not always—involves some sort of internal reconstruction of the external environment. The thinking process simulates how the environment would activate the perceptual system and how it would be affected by a potential action. Actually, the term "simulation theory" encompasses a number of different accounts of mental simulation [89, 90]. For this reason, it is useful to try to be more precise about what simulation means in the context of cognition. According to Fisher [69], a cognitive simulation is a kind of *mental process*, in the sense of a sequence of successive mental states each depending on the previous one. A mental process becomes a simulation when it meets the following conditions:

1. the process facilitates knowledge about the subject of simulation;
2. the process reflects significant aspects of what is simulated.

Hence, simulation can be considered an "epistemic device" aiming to produced knowledge about the process that is being simulated.

The definition of simulation holds for all the different facets of mental simulation proposed over the years. The first and still prevailing account of mental simulation is related to the so-called *theory of mind* [80]. This theory regards the ability of a person to guess the mental states of another interacting person. The observer constructs a sort of "theory" of how the mind works in general, and they use it to simulate what is going on in the mind of the target person. This account of simulation pertains to social cognition and is certainly relevant in the context of driving [209], but it is scarcely related with perception, so I am not discussing it any further. In the following sections, I will present in detail other forms of mental simulation more closely related to the topic of my work.

2.1.1 Simulation as Emulation

In the previous section, I have mentioned that mental simulation is a sequential process. More precisely, steps of the simulation should mimic the corresponding steps of the represented situation. Most of the time, the simulation is a rough and short imagination of the reality; every step of the original event does not usually have to correspond to a distinct step of the simulation. Conversely, every state of the simulation must correspond to an intermediate state of the real event. For example, when I imagine to drive from home to work, I do not actually simulate every single turn of the steering wheel but just a sequence of key turns on the overall route.

A crucial question regards how the brain is able to transit to successive states in the simulation. In general, this process seems to bear little resemblance to the way the states of the real event succeed each other. For example, I can imagine a car swerving on an icy road, but the mechanism my brain uses to imagine the swerving car has little or nothing to do with the the physical loss of grip of the tires on ice. However, there are cases where the brain seems to reproduce the states of the event by mimicking the real process that transform the successive states. This peculiar type of simulation is called *emulation*.

The *emulation theory of representation* is strongly linked to the field of control theory and signal processing. Not surprisingly, the developer of this theory is an American philosopher with a engineering background, Rick Grush [83]. The main influence comes from the *pseudo-closed-loop* control schematics: a controller receives the desired action and sends the corresponding control signal to the plant, which performs the action. In order to work correctly, the control needs feedback information about the effect of the signal on the plant. Since this feedback is difficult to obtain or may arrive too late to the control, the alternative in the pseudo-closed-loop system is to send a copy of the control signal to an *emulator*, a device that imitates the plant offline and produces information similar to the real feedback. Emulators exploit the classic technique of Kalman filters [113] to progressively estimate the behavior of any dynamic plant from a series of measurements.

The emulation theory finds the most appropriate cognitive counterpart in the domain of motor control. In this case, emulators have the role of emulating parts of the musculoskeletal system. This theory successfully solves the conundrum of how, during motor imagery, proprioception and kinesthesis—necessary for sustaining a dynamic plan—can exist in absence of limbs modifications. Today, there is evidence supporting the necessity for cognitive emulators in skeletal and eye movements [37]; a model based on Kalman filter is able to explain experimental data of hand movements [239]. Note that Kalman filters are just a mathematical framework suitable for modeling this kind of emulators. How the brain actually achieves this task is still uncertain. However, the emulation account of cognitive simulation for motor control—Kalman filters included—is certainly useful in the domain of autonomous vehicles [41], but it is less relevant for the perceptual aspects. The next section focuses on the account of mental simulation most close to visual perception.

2.1.2 Simulation as Imagery

The form of mental simulation most relevant for perception is *imagery*: the phenomenon when the brain reconstructs a percept in absence of external stimuli. Although this ability exists for all the human senses [1], visual mental imagery plays a dominant role in humans, also because of the large portion of cortical tissue assigned to the processing of visual information. One of the leading researchers on visual imagery is Stephen Kosslyn [121], who first proposed that visual perception and imagination share much of their neural processing. In fact, neural representations of imagined and perceived stimuli appear similar in the visual, parietal, and frontal cortex.

Visual imagery has been the center of a long-running controversy in cognitive science, known as the “imagery debate”. Supporters of the earlier cognitive science, which was strongly founded on language and symbols, have claimed that imagination—unlike direct vision—should have a propositional format in the mind. To consciously retrieve images from memory, the images have to be structured in a semantic form [185]. The opposite view, lead by Kosslyn [122], argued that the content of imagery is mostly depictive, exactly like visual perception, even though the subject is well aware of the semantic content of their imagination. Today, there is overwhelming evidence that imagery is essentially depictive—while still involving semantic information [173].

Just like generic imagination, visual imagery plays a variety of functions. With visual imagery, human can replay events from the past, conceive alternative and fictional realities, or picture a potential future. But one of the most fascinating functions of imagery—crucial in the context of driving—is the improvement of perception. There is ample evidence of this role: several experiments have demonstrated that when a person imagines a stimulus before the actual percept, imagery can facilitate the perception of the real stimulus [65, 105, 172]. Another function of visual imagery highly relevant to driving is to generate specific short-term predictions based upon past experience; this function links imagery with emulation. In fact, most bodily controls uses vision as a primary form of guidance: imagery mimics the perception of movements, both of the body and the surrounding objects, and generates predictions similar to Grush’s emulators [152]. Having clarified the importance of visual simulation in human cognition, the next section deals with the question of how the brain realizes simulation through the neural circuits.

2.2 Convergence-Divergence Zones in the Brain

A critical point for the theories on imagination is to explain how neural circuits can combine perception with imagery. What is the mechanism that allows neural circuits to recall entities from the memory and also recognize the content of a scene from perceptual stimuli? One of the more compelling answers to this question is due to the Portuguese-American neuroscientist Antonio Damasio [45, 46]. He introduced the concept of *convergence zones*: neural ensembles that link the representations of a same entity coming from different sensory and motor cortices. To better explain this, an external entity can be perceived under many aspects: visual, auditory, spatial, to mention a few. These fragments of representations are distributed across separate sensory and motor cortices. The convergence zones bind the distributed neural activity patterns that correspond to the same perceived stimulus. The binding is learned through experience on the basis of similarities, and spatial or temporal relations.

The convergence zones can account for the amodal representations of entities and events

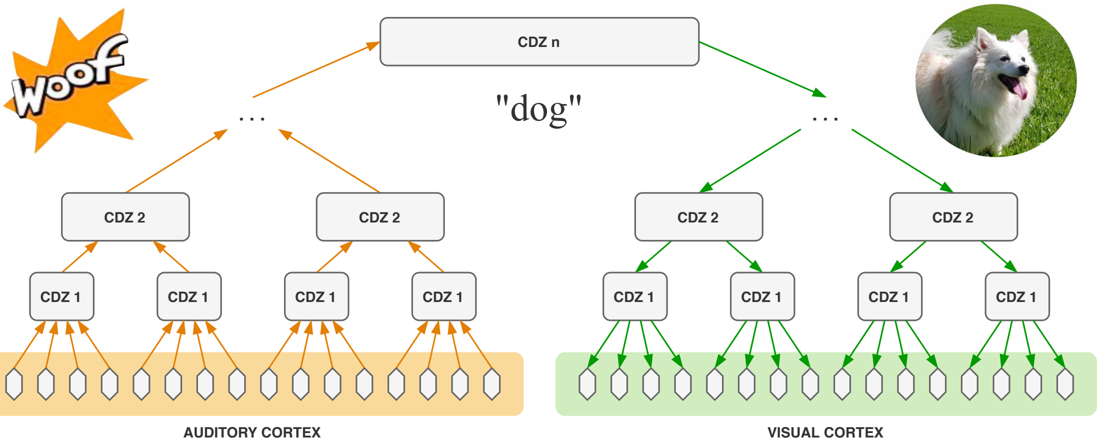


Figure 2.1: Schematic depiction of the convergence-divergence zones linking the representations of the entity "dog" in the auditory and visual modalities. When hearing the woof of a dog, I can recall from memory the visual appearance of my beautiful dog.

of the external world, i.e., conceptual representations that are abstract and have lost their perceptual properties. In fact, the convergence zones have the function of projecting in a many-to-one way from multimodal cortical regions into amodal regions. It is also possible to find convergence zones that bind features collected by different cortical areas of the same modality. This is the case of vision, in which there are tens of segregated cortical areas processing different visual features with different temporal and spatial scales.

In a recent refinement of his work, Damasio highlighted a reciprocal function of these neuron ensembles, introducing the more sophisticated *convergence-divergence zones* (CDZs) [147]. Besides projecting in a convergent way, the CDZs can produce divergent projections in a one-to-many way. The fascinating aspect is that, while feedforward projections characterize the mechanisms of recognition, the feedback projections are activated during memory recall. Moreover, CDZs can offer a powerful explanation of the more specific processes of visual perception and imagery. Fig. 2.1 gives a visual description of this mechanism.

Several neurophysiological and neuroimaging studies [141] now support this proposal, with evidence of hierarchies of CDZs at all levels in the brain. In a study from 2006 [223], subjects had to imagine one of the six possible domino-like patterns in either the left or right visual hemifield. By processing the fMRI data of the visual cortex of the subjects, it was possible to infer both the imagined pattern and its position in retinal space. A more recent study [40] analyzed the brain activity of the subjects when imagining a fruit, successfully decode the identity of the fruit, the shape, and the color. In conclusion, the CDZs represent the fundamental tool to support the formation of mental concepts and to reenact previous visual experiences with a simulation-like process [163].

2.3 Perceptual Learning

The next cognitive principle that could improve the design of an autonomous vehicle concerns the role of learning within the perceptual abilities. Nowadays, humans need to *learn* most of their skills through experience. In fact, humans lack the genetic instructions for modern-day tasks—such as driving a car—because the pace of technology development is orders of magnitude greater than the timescale of natural evolution. Among the learned skills, there are skills that are purely mental, like translating from a language to another or solving a mathematical problem. There are other skills, instead, that involve motor aspects and require *sensorimotor learning* [238], like playing tennis or (of course) driving a car. Sensorimotor learning involves the acquisition of a number of interacting competences, including the implementation of reactive control mechanisms, decision making, the selection of strategies, and the efficient gathering of task-relevant perceptual information. This last competence is acquired by *perceptual learning*.

Perceptual learning concerns non-declarative knowledge, i.e., knowledge that is expressed through performance rather than through recollection, like riding a bicycle—in contrast to declarative knowledge that refers to the capacity of recollecting facts and events, like the names of the planets of the solar system. Unlike the declarative forms of learning, perceptual learning does not require an intermediate consolidation storage such as the hippocampus, and it seems to affect directly the neural perceptual mechanisms. In fact, perceptual learning improves the actual perceptual performance: after training in a perceptual task, one is able to perceive something new that could not do before [64].

The study of perceptual learning is relatively recent with respect to other forms of traditional learning investigated in psychology and cognitive science. All perceptual modalities benefit from perceptual learning, but the most investigated is visual perceptual learning [206, 54]. Over the past two decades, the field of visual perceptual learning has evolved significantly, and there is abundant evidence of how experience is crucial in shaping human visual perception. In addition, there is now a better understanding of which visual areas are involved during perceptual learning. A recent work [211] have presented the paradigmatic case of visual perceptual learning performed by radiologists. After years of experience and training, radiologists learn to interpret X-ray images and discriminate between subtle differences of light and dark. This is a typical case of a professional perceptual skill, where there is a sharp difference in performance between experts and non-experts. The form of visual perceptual learning that involves driving is less evident, yet people learn to drive effortlessly after few years of experience. During the training period, the overall sensorimotor learning takes place, including the motor abilities of steering the wheel and using the pedals as well as the visual abilities concerning the main objects on the road: vehicles, traffic signs, lanes, pedestrians, obstacles. One of the skills derived by perceptual learning especially relevant when driving is the ability to detect impending collisions [134]. Hence, it is clear how learning by experience has a key role in visual perception in the context of driving.

2.4 Predictive Brains and Free Energy

The last cognitive theory I have investigated in relation to the activity of driving has gained large attention in the cognitive sciences during the last decade. This theory, proposed by Karl Friston [72, 73, 74], has become popular under the terms "Bayesian brain", "predictive brain", or "free-energy principle for the brain", and it is rapidly gaining consensus in various research fields such as cognitive science, psychology, neuroscience, and philosophy [36, 98].

According to Friston, the behavior of the brain—and of an organism as a whole—can be conceived as minimization of the *free energy.* The free energy has its origin in the field of thermodynamics, where it measures the portion of energy available in a system to perform thermodynamic work at constant temperature. The concept of free energy is mainly abstract and has only few precise forms, like the Helmholtz free energy or the Gibbs free energy. However, Friston does not exploit the thermodynamic meaning of the free energy, but its "free" usage in some statistical frameworks, especially variational Bayes methods [230]. In particular, he bases its theory on the relation between the thermodynamic entropy and the information entropy. The abstract formulation of free energy proposed by Friston is the following:

$$F_{\Phi}\left(\widetilde{\mathbf{x}}|\mathbf{a}\right) = \mathbb{E}_{\mathbf{c}\sim q_{\Phi}(\mathbf{c})}\left[-\log p\left(\widetilde{\mathbf{x}},\mathbf{c}|\mathbf{a}\right) + \log q_{\Phi}\left(\mathbf{c}\right)\right], \tag{2.1}$$

where $\widetilde{\mathbf{x}}$ is the sensorial input of the organism, $\mathbf{c}$ is the collection of the environmental causes producing $\widetilde{\mathbf{x}}$, $\mathbf{a}$ are the actions that may modify the environment and (as a consequence) the sensorial input in the future. The tilde in $\widetilde{\mathbf{x}}$ stands for an arbitrary number of time derivatives of the instantaneous vector $\mathbf{x}$ of sensorial input. The distribution $q_{\Phi}(\mathbf{c})$ is an internal model of the organism, and it refers to the probability density that the causes $\mathbf{c}$ would take place in the environment. The subscript Φ indicates the status of the brain, which is defined by a set of brain variables, for example, the strength of synaptic connections or the neuromodulator densities.

The free-energy principle for the brain implies that, by minimizing the free energy, the brain minimizes the exchanges with the environment that are considered unlikely or unexpected (often called *surprise* or *surprisal*), which are represented by the term $-\log p\left(\widetilde{\mathbf{x}},\mathbf{c}|\mathbf{a}\right)$ of equation (2.1). Although this equation may appear too abstract, it is possible to derive more precise formulations by specifying the brain function of interest. For example, the free energy corresponding to perception does not care about the potential actions $\mathbf{a}$, and it can be expressed as follows [76, p.427]:

$$F_{\Phi} = \Delta_{\mathrm{KL}}\Big(q_{\Phi}(\mathbf{c})\|p\left(\mathbf{c}|\widetilde{\mathbf{x}},\mathbf{a}\right)\Big) - \log p\left(\widetilde{\mathbf{x}}|\mathbf{a}\right), \tag{2.2}$$

where first term is the Kullback-Leibler divergence, which measures the distance between two distributions. In this case, minimizing F_{Φ} means to find a brain status Φ such that the internal expectation of the environmental causes $q_{\Phi}\left(\mathbf{c}\right)$ matches the actual density distribution $p\left(\mathbf{c}|\widetilde{\mathbf{x}}\right)$. On the other hand, the definition of the free energy corresponding to actions [76, p.428] is the following:

$$F_\Phi = \Delta_{\mathrm{KL}}\Big(q_\Phi(\mathbf{c})\|p(\mathbf{c})\Big) - \mathbb{E}_{\mathbf{c}\sim q_\Phi(\mathbf{c})}\Big[\log p\left(\widetilde{\mathbf{x}}|\mathbf{c},\mathbf{a}\right)\Big]. \tag{2.3}$$

In this case, the brain tries to choose the action that could expose the organism to causes in the environment that are likely expected. Note that the formalization of equations (2.2) and (2.3) still do not explain how to make the variables explicit or how to find the solutions to the equations. However, Friston [73, p.130] speculated on putative roles of neurons in the free energy minimization. Moreover, in support to the theory, he developed a computational model for the categorization of birdsong based on free energy minimization [75].

2.4.1 Imagination and Prediction

The free-energy principle is considered the new frontier of cognitive science because it attempts to unify all the aspects of cognition. However, it is wrong to consider this theory as opposed to the simulative account of cognition described in §2.1. In fact, Friston's theory can explain why the human visual system incorporates a generative capacity. The reason for this is to perform *perceptual inference*, i.e., to always be prepared to perceive what is mostly expected from the environment. When engaged in a demanding perceptual task like the analysis of the road scenario during driving, there is a deeply unified cooperation between perception and imagination. In turn, this perception/imagination alliance cooperates with the driving action selection to minimize "surprises" [120]. In this context, imagination does not correspond to an offline recollection of visual memories. It is, instead, a way to focus, refine, clarify, or concentrate on the current experience based on the previous events. Quoting the words of the philosopher Wiltsher [235], imagination can be interpreted "as a set of lenses rather than an imitating mirror".

In this chapter, I have illustrated four well-established theories related to how the human brain achieves the cognitive capabilities required to drive. In the next chapter I will try to identify some possible counterparts to these theories within the field of deep neual networks.

Chapter 3

Deep Learning Methods

In the Introduction, I have pointed out the predominant role the methods based on deep learning have in the context of artificial perception. During the last decade, deep neural networks have become so successful to make most of the existing methods in computer vision obsolete, including the applications to autonomous driving. Nowadays, deep learning represents the most prominent choice to build competitive solutions for artificial perception.

In addition to this, deep neural networks conform—to some extent—to the mechanisms humans adopt to perceive the external environment during complex sensorimotor tasks (like driving), which I have analyzed in Chapter 2. It is possible to identify some artificial architectures and strategies that could be compatible with related features and processes of human perception. In this chapter, I will briefly introduce the history of artificial neural networks and the recent transformation into deep architectures. Then, I will describe the specific artificial models I adopt in my work and the resemblance with their cognitive counterparts. Specifically, I will analyze the properties of convolutional networks, autoencoders, and variational Bayesian inference.

3.1 Evolution of Artificial Neural Networks

When first introduced back in the middle of the last century, the artificial neural networks (ANNs)—as can be deduced from the name—aimed to mimic the functioning of neurons in the brain. The key inspiration is the concept of *synaptic plasticity*, i.e., the fact that neural circuits are initially amorphous and gradually learn purposeful functions through experience. During the evolution towards deep learning, however, artificial neural networks have gradually lost their original strong neurocognitive commitment.

3.1.1 Perceptron and Backpropagation

One of the earliest implementations of artificial neural network is the *perceptron*, designed by Frank Rosenblatt [196, 197]. In the original proposal, the perceptron is an electronic device composed of three parts: an input matrix of sensorial units called “S-points”, a vector of associative units called “A-units” that receive connections from the S-points, and an output vector of “R-units” one for each class of objects to be recognized in the input. Each A-unit receives excitatory connections from several S-units, and it is triggered when the sum of the signals from the input connections reaches a fixed threshold. The R-units are connected with the A-units in a similar way, with an important difference: the connections can change dynamically, using motor-driven potentiometers that enact the mechanism of learning. The learning of the perceptron observes the following rule:

$$w_{i,j}^{(t+1)} = w_{i,j}^{(t)} + \eta\, a_i \left(\hat{r}_j^{(t)} - r_j \right), \tag{3.1}$$

where a_i is an A-unit connected with a R-unit r_j through a synapse with efficiency $w_{i,j}$, which is a real number in the range [0..1]. For an input sample t, the known correct level of activation of the unit r_j is $\hat{r}_j^{(t)}$. By virtue of this learning rule, if a R-unit is wrongly triggered, the weights of the connections between this unit and all the currently active A-units are reduced with a factor η. Conversely, if the R-unit is not active when it should be, the connection weights are increased. The A-units that are inactive for the sample t do not change.

When first proposed, the perceptron was met with harsh criticism [149] and considered disruptive for the progress of artificial intelligence of that time. The main criticism was that the learning rule (3.1) works for one plastic layer only. In fact, Rosenblatt himself acknowledged this issue before [197, p.579]. Ultimately, the problem concerning the number of learnable layers has affected artificial neural networks since their creation. The solution to this issue came twenty years later, from the research group lead by Geoffrey Hinton, under the name of *backpropagation* [199].

Backpropagation is a learning algorithm for neural networks that calculates the gradient of the error function with respect to the neural network's weights. Backpropagation applies specifically to *feedforward* neural networks, i.e., organized into layers with unidirectional connections. Being $\mathbf{w}$ the vector of all learnable parameters in the network, and $\mathcal{L}(\mathbf{x}, \mathbf{w})$ a measure of the error of the network with parameters $\mathbf{w}^{(t)}$ when applied to the sample $\mathbf{x}^{(t)}$, the backpropagation updates the parameters iteratively according to the following formula:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla_w \mathcal{L} \left(\mathbf{x}^{(t)}, \mathbf{w}^{(t)} \right), \tag{3.2}$$

where η is the *learning rate*, and ∇_w is the gradient of the weights. Note that, in order to compute the error $\mathcal{L}(\mathbf{x}, \mathbf{w})$, it is necessary to known a priori the correct responses to all samples $\mathbf{x}$ used during learning; this learning process is known as *supervised learning.* The invention of backpropagation has paved the road for a highly successful period for artificial neural networks [200].

3.1.2 Vanishing Gradient Problem

In the 1990s, it seemed obvious that the most effective feedforward models should have no more than a single hidden layer. This was, in fact, a limitation due to the intrinsic formulation of backpropagation: networks with multiple hidden layers suffer from the local minima problem during learning. The main difficulty lies in the chain rule necessary to compute the gradient of weights with respect to the errors in equation (3.2). If many weights across multiple layers are less than 1, when they are multiplied many times, the gradient could start to vanish into the smallest machine number—this is the well-known problem of the *vanishing gradient*. Also because of this limitation, at the beginning of this century, artificial neural networks seemed to have exhausted their potential, and the research field was stagnating.

The resurgence of artificial neural networks is due—again—to Hinton [92], who proposed a method to train models with four and five hidden layers, usign a solution based on one of his earlier ideas: the *restricted Boltzmann machines* (RBMs) [94], which can learn in an unsupervised mode. The idea is simple: take two adjacent layers in a feedforward network and train them as a RBM. The procedure starts creating the first RBM with the input layer and the first hidden layer; this first machine generates a new set of data by processing the original inputs. Then, the new set of data is used to train the next couple of layers, and so on for all the layers in the network. This procedure is a sort of pretraining that gives a first shape to all the connections in the network, which will be further refined by the ordinary supervised learning with backpropagation. The research community soon recognized the layer-wised pretraining with RBMs to be an elegant solution to the problems afflicting backpropagation for all sort of neural networks with multiple layers [91, 12, 203].

The layer-wise pretraining with restricted Boltzmann machines has the great merit of having acted as a catalyst for the revival of ANNs. However, from a technical point of view, it is not essential for dealing with more than three layers. In fact, it turns out that backpropagation can easily train deep feedforward networks with just few improvements; this new algorithm takes the name of *stochastic gradient descent* (SGD) [20]:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla_w \frac{1}{M} \sum_{i}^{M} \mathcal{L}\left(\mathbf{x}^{(i)}, \mathbf{w}^{(t)}\right). \tag{3.3}$$

It is immediate to see the similarity with the standard backpropagation equation (3.2). In this case, instead of computing the gradients over a single sample t, the algorithm makes a stochastic estimation over a random subset of size M of the entire dataset and, at each iteration step t, it samples a different subset of the same size. The growth of the research on deep learning has progressively improved and refined the SGD algorithm. A first crucial innovation is the *dropout* [95], a technique that randomly switches off a fraction of the neurons during training. This expedient prevents unwanted co-adaptations of feature detectors on a limited set of samples. Other improvements include various techniques such as adapting the learning rate η dynamically during training [56], or mixing the update given by equation (3.3) at step $t + 1$ with the update at step t [118].

3.1.3 Similarities with Biological Neural Networks

Given that my project aims to draw inspiration from brain mechanisms, it is important to discuss the relation between artificial neurons and biological neurons. In fact, the structure of an artificial neuron bears little resemblance with the highly complex cell of a biological neuron. Moreover, "learning" in artificial neural network is realized with algorithms that share nothing with the biological mechanisms of learning at neural level [16, 66, 27]. There are some computational models that simulate the actual behavior of biological neurons [184, 143], but they are very distant from deep learning and are hardly applicable to engineering problems. Even Hinton tried to develop a learning mechanism closer to that of biological neurons, with poor results [9].

It is possible, however, to identify some preliminary points in common between artificial neural networks and neurons in the brain:

- high-level functions of neural circuits derive from the interaction of a large number of units (neurons) that are very similar to each other;
- neural circuits are essentially learning devices that start from an amorphous state, and their mature function depends on experience;
- the activation of a neuron depends on the cumulative sum of the weighted activations of all neurons that are connected to it.

These are the first similarities that is possible to observe in general. When digging more deeply, there are additional—and more interesting—similitudes with specific kind of neural models. This is the topic of the rest of the chapter.

3.2 Convolutional Neural Networks

The operation of convolution is one of the oldest and well established technique adopted in image processing [198]. The first combination of convolutions with artificial neural networks is due to Kunihiko Fukushima, who proposed the architecture called *neocognitron* [77]. The neocognitron consists mainly of two types of cells called *S-cells* and *C-cells*—in accordance to the classification of "simple" and "complex" cells in the primary visual cortex [100]. The S-units act as convolution kernels, while the C-units downsample the resulting images by applying a spatial average.

After the introduction of backpropagation as the most effective learning method, Yann LeCun proposed to combine layers of neocognitrons with feedforward layers and train them with backpropagation [133]; this is the very first example of *convolutional neural networks* (CNNs). At the time, however, this solution could not compete with the established non-neural techniques in computer vision [4]. The great breakthrough arrived with the invention of *deep convolutional neural networks* (DCNNs), the first of which was an eight-layer network [125] proposed by the research group lead by Hinton. This network triumphed in the famous *ImageNet Large-Scale Visual Recognition Challenge* (ILSVRC), dropping the error rate from the previous 26.0% down to 16.4%. After this event, DCNNs have been at the core of the computer vision research, with a succession of successful developments that still lasts nowadays.

3.2.1 Convolutional Networks and Human Vision

The discussion on the relation between ANNs and biological neurons, started in Section §3.1.3, deserves further attention in the case of DCNNs. As mentioned above, Fukushima introduced the S-cells and C-cells to emulate the behavior of simple and complex cells in the primary visual cortex. Again, the biological inspiration implies nothing about the actual similarity in terms of behaviors or architectures. In fact, the contributions of simple and complex cells in the primary visual cortex are much more complex than the operations of convolutions and downsampling in the neocognitron [14]. Moreover, there are striking differences between the computations performed by DCNNs and the processes known so far in the visual cortical areas [193].

Despite these significant differences, recent studies show surprising patterns of similarities. Given an image, it is possible to note a resemblance between the layer activations of a DCNN and the response stages of the cortical visual system of a person looking at the image. This phenomenon was observed in several experiments comparing human fMRI data and DCNN models [84, 117, 241]. These studies confirm that, although there are still discrepancies, the patterns of activities in DCNN layers and in the human visual system are consistent and surprisingly similar; this suggests the existence of a common set of processes or features. Even if, at the moment, it is not possible to identify precisely which kind of process or feature is shared by the two systems, this analogy represents a strong ground for the adoption of DCNNs in my project.

3.3 Representation Learning

The scope of application of deep neural networks (DNNs) is not limited to achieving the desired mapping from input to output. On the contrary, DNNs are often used to learn effective "representations" of the input [11], especially when the input data are sensorial information in a perceptual context. This account of internal representations could be akin to the mental representations used by humans to conceptualize the world around them.

Truth to be told, there exists a long and debated controversy in cognitive science around the idea of mental representations. The so-called *embodied* and *enactive* cognitive science holds that there is no need to make use of representations altogether, as that cognition can be explained by directly relating perception with the acting in the environment [23, 201, 102]. However, unlike in current cognitive science, neuroscience commonly considers neurons as a representing device and employs a representational vocabulary to characterize various neural processes [35, 124].

In several engineering applications, the concept of representation learning is considered a key process. The efficiency of complex algorithms is often strongly dependent on the design of the representation used for the data. Instead of manually defining the representation from a prior analysis of the relevant features of the data, DNNs can be used to learn the best representation from the data themselves. This approach represents a radical improvement in the design of data representations.

3.3.1 Autoencoders

One of the first and most successful neural network developed for representation learning is the *autoencoder* [92]. It is a network trained to produce a reconstruction of the input; in doing so, the model develops in the inner layer a compact representation of the input data. The theoretical advantage of this architecture is that it learns the representation without any prior information on the data—this learning scheme is called *self-supervision*. The model can be described by two functions:

$$g_\Phi \quad : \quad \mathcal{X} \to \mathcal{Z}, \tag{3.4}$$

$$f_\Theta \quad : \quad \mathcal{Z} \to \mathcal{X}. \tag{3.5}$$

The first is called *encoder* (or *generator*), defined by a set of parameters Φ, and it computes the compact representation $\mathbf{z} \in \mathcal{Z}$ of a high-dimensional input $\mathbf{x} \in \mathcal{X}$. The second function is the *decoder*, defined by the parameters Θ, and it aims to reconstruct the high-dimensional data $\mathbf{x}$ from the low-dimensional representation $\mathbf{z}$. The autoencoder is trained to minimize the loss $\mathcal{L}_{\Theta,\Phi}\Big(\mathbf{x}_i,\ f_\Theta\big(g_\Phi(\mathbf{x}_i)\big)\Big)$ for each sample $\mathbf{x}_i$ of the dataset $\mathcal{X}$.

3.3.2 Autoencoders and Convergence-Divergence Zones

Autoencoders—specifically convolutional autoencoders—manifest a compelling similarity with the biological strictures of convergence-divergence zones, introduced in Section §2.2. They both consist of a convergent path and a divergent path: the convergent component compresses sensorial information into abstract high-level representations, in a distributed and hierarchical way; symmetrically, the divergent component reflects the activations back to a low-level perceptual space.

There is, however, an inevitable physical difference: the CDZs exploit the same cells to supports both the convergent and the divergent streams. This is impossible for the autoencoders, as they necessarily have to use two separate groups of layers with different weights. Nonetheless, in the abstraction of the computation, it makes no difference if the memory locations of the variables in the convergent path are different from those in the divergent path, as long as the decoder mirrors the encoder.

3.3.3 Variational Autoencoders

In the last few years, the probabilistic Bayesian inference has received a renewed interest in the context of learning high-dimensional models. The Bayesian framework—in particular variational inference—has found a fertile ground in combination with deep neural models. Two concurrent and unrelated developments [119, 190] made this theoretical advance possible, connecting autoencoders and variational inference; this new approach quickly became popular under the term *variational autoencoder*.

The variational inference framework takes up the issue of approximating the probability distribution $p(\mathbf{x})$ of a high-dimensional random variable $\mathbf{x} \in \mathcal{X}$. The output distribution can be computed through a neural network f of parameters Θ as follows:

$$p_\Theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}\left(\mathbf{x} \mid f_\Theta(\mathbf{z}), \boldsymbol{\sigma}^2\mathbf{I}\right), \tag{3.6}$$

where $\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\sigma})$ is the Gaussian function in $\mathbf{x}$, with mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$. The desired approximation of $p(\mathbf{x})$ is, therefore, the following:

$$p_\Theta(\mathbf{x}) = \int p_\Theta(\mathbf{x}, \mathbf{z})\, d\mathbf{z} = \int p_\Theta(\mathbf{x}|\mathbf{z})\, p(\mathbf{z})\, d\mathbf{z}. \tag{3.7}$$

In equation (3.7), there is clearly no clue on what the distribution $p(\mathbf{z})$ might be. The idea behind variational autoencoder is to introduce an auxiliary distribution $q_\Phi(\mathbf{z}|\mathbf{x})$ from which to sample $\mathbf{z}$. Ideally, this should provide the posterior probability $p_\Theta(\mathbf{z}|\mathbf{x})$, which is unknown. The distribution can be derived from an additional neural network g of parameters Φ:

$$q_\Phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}\left(\mathbf{z}|g_\Phi(\mathbf{x}), \boldsymbol{\sigma}^2\mathbf{I}\right). \tag{3.8}$$

The measure of how well $p_\Theta(\mathbf{x})$ approximates $p(\mathbf{x})$ for a set of $\mathbf{x}_i \in \mathcal{D}$ sampled in a dataset $\mathcal{D}$ is given by the log-likelihood:

$$\ell(\Theta|\mathcal{D}) = \sum_{\mathbf{x}_i \in \mathcal{D}} \log \int p_\Theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})\, d\mathbf{z}. \tag{3.9}$$

This equation cannot be solved because of the unknown $p(\mathbf{z})$—here comes the help of the auxiliary probability $q_\Phi(\mathbf{z}|\mathbf{x})$. Each term of the summation in equation (3.9) can be rewritten as follows:

$$\begin{aligned} \ell(\Theta|\mathbf{x}) &= \log \int p_\Theta(\mathbf{x}, \mathbf{z}) d\mathbf{z} \\ &= \log \int \frac{p_\Theta(\mathbf{x}, \mathbf{z}) q_\Phi(\mathbf{z}|\mathbf{x})}{q_\Phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\ &= \log \mathbb{E}_{\mathbf{z} \sim q_\Phi(\mathbf{z}|\mathbf{x})} \left[\frac{p_\Theta(\mathbf{x}, \mathbf{z})}{q_\Phi(\mathbf{z}|\mathbf{x})} \right], \end{aligned} \tag{3.10}$$

where the last passage uses the expectation operator $\mathbb{E}[\cdot]$. Being the log function concave, it is possible to apply the Jensen's inequality:

$$\begin{aligned}\ell(\Theta, \Phi|\mathbf{x}) &= \log \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x})}\left[\frac{p_\Theta(\mathbf{x},\mathbf{z})}{q_\Phi(\mathbf{z}|\mathbf{x})}\right] \\ &\geq \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x})}\left[\log p_\Theta(\mathbf{x},\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x})}\left[\log q_\Phi(\mathbf{z}|\mathbf{x})\right]. \qquad (3.11)\end{aligned}$$

Since the derivation in the last equation is smaller or at least equal to $\ell(\Theta|\mathbf{x})$, it is called *evidence lower bound* (ELBO). Note that in $\ell(\Theta, \Phi|\mathbf{x})$ there is also the dependency from the parameters Φ of the second neural network defined in (3.8). It is possible to further rearrange $\ell(\Theta, \Phi|\mathbf{x})$ to have $p_\Theta(\mathbf{x}|\mathbf{z})$ instead of $p_\Theta(\mathbf{x},\mathbf{z})$ in equation (3.11).

At this point, it is possible to introduce the loss function $\mathcal{L}(\Theta, \Phi|\mathbf{x})$ as the value to be minimized in order to maximize ELBO:

$$\begin{aligned}\mathcal{L}(\Theta, \Phi|\mathbf{x}) &= -\ell(\Theta, \Phi|\mathbf{x}) \\ &= -\int q_\Phi(\mathbf{z}|\mathbf{x}) \log \frac{p_\Theta(\mathbf{x},\mathbf{z})}{q_\Phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\ &= -\int q_\Phi(\mathbf{z}|\mathbf{x}) \log \frac{p_\Theta(\mathbf{x}|\mathbf{z}) p_\Theta(\mathbf{z})}{q_\Phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\ &= \Delta_{\mathrm{KL}}\big(q_\Phi(\mathbf{z}|\mathbf{x}) \| p_\Theta(\mathbf{z})\big) - \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x})}\left[\log p_\Theta(\mathbf{x}|\mathbf{z})\right]. \qquad (3.12)\end{aligned}$$

where the last step uses the Kullback-Leibler divergence Δ_{KL}. Still, this formulation seems to be intractable because it contains the term $p_\Theta(\mathbf{z})$. However, there is a simple analytical formulation of the Kullback-Leibler divergence in the Gaussian case [119, Appendix B]:

$$\Delta_{\mathrm{KL}}\Big(q_\Phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})\Big) = -\frac{1}{2}\sum_{i=1}^{Z}\left(1 + \log\left(\sigma_i^2\right)\right) - \mu_j^2 - \sigma_i^2\big), \qquad (3.13)$$

where μ_i and σ_i are the i-th components of the mean and variance of $\mathbf{z}$ given by $q_\Phi(\mathbf{z}|\mathbf{x})$. At this point, it is clear that the networks g_Φ and f_Θ play the roles of, respectively, the encoder and decoder components of an autoencoder.

3.3.4 Variational Autoencoders and Predictive Brains

The adoption of variational inference leads to a mathematical formulation of the variational autoencoder impressively similar to the concept of free energy proposed by Friston. It is easy to notice the resemblance when comparing equation (3.12) with the definition of free energy for actions in equation (2.3).

This close analogy has gone unnoticed by the protagonists of the research on variational autoencoders—even both Kingma & Welling and Rezende et al. seem to ignore the theories of Friston altogether. Reciprocally, Friston makes no mention of variational autoencoders or any application to deep neural networks in general, although he has very recently demonstrated interest in deep learning: he is currently collaborating with DeepMind to develop a very general framework for intelligent agents [86].

Ultimately, this reciprocal disregard is not so surprising, given that mainstream deep learning is mainly driven by engineering goals without any particular interest towards cognition, and that cognition has largely ignored deep learning until recently. There are just a couple of exceptions worth mentioning. A first work [162] uses variational autoencoders inspired by the Friston's predictive theory to model human electroencephalogram and physiological signals of subjects watching video excerpts. Another work [227] implements an agent based on variational autoencoders that performs active inference under the free-energy principle. The work evaluates the performance of the agent using a toy situation. However, even these very few exceptions do not address real engineering applications.

Chapter 4

Deep Learning for Autonomous Driving

The previous chapter has illustrated how deep learning has played a tremendous role in advancing a wide range of technological applications, autonomous driving included. Deep learning methods have shown great promises in several aspects of autonomous driving. Some of these aspects, like path planning or behavior arbitration, have limited relevance to my work, so I do not provide further details on them—thorough surveys can be found in [82, 129, 99]. Before deep learning, visual perception has been perhaps the most serious hindrance to the development of autonomous vehicles. In fact, perception is the task enjoying the greatest leap forward because of deep learning, and several surveys cover this aspect [213, 228, 108]. This chapter provides a brief overview of the popular deep learning approaches that aim to implement perception for automated driving, focusing also on the fundamental role played by datasets in training a deep neural network.

4.1 Approaches to Perception for Driving

The distinctive strategy pursued in my work—borrowing concepts from the functioning of the brain and the mind—is mostly overlooked in the field of autonomous vehicles, but it is certainly not novel. There are works adopting neural networks for intelligent vehicle perception that declare virtues of a neurocognitive inspiration [169, 31, 244]. However, these ideas often do not transfer the specific brain mechanisms into algorithms. To the best of my knowledge, the two main neurocognitive principles embraced by this work—Damasio's CDZs and Friston's predictive brain—have not been proposed in any work on perception for autonomous driving.

The majority of ongoing developments for autonomous vehicles adopts artificial neural models without caring for the biological plausibility. The lack of cognitive foundation is often linked to the employment of on-board sensors that collect information alien to the human perception system. A clear example is the common use of LIDARs to produce bird's-eye views of the environment—a representation format that is impossible for a human driver. In the context of driving, the human sensory system can be roughly reduced to a stereo camera setup, and this is more than sufficient for a person to carry out the task of driving a vehicle in complex or unknown scenarios. Vision, however, might not be the only human sense engaged in this context: a noteworthy work proposes to use auditory information to detect vehicles approaching behind blind corners before they enter in line-of-sight [208]. Since the focus of this work is visual perception, I will not further analyze the role of hearing in driving. The following sections will identify three general approaches to visual perception for intelligent vehicles.

4.1.1 Modular Perception

The conventional engineering approach decomposes the overall problem of perception into a set of sub-tasks to be solved with independent modules. The modules interact with each other only to exchange their input and output, typically in hierarchical pipelines. According to recent studies [99, 108], the most common modules for perception are 2D object detection, 2D object tracking, segmentation, depth estimation, ego-motion estimation, localization, 3D object detection, and 3D scene understanding. In earlier works within traditional computer vision, each of the listed modules comprised in turn its own modular pipeline. For example, 2D object detection used to be divided into preprocessing, extraction of regions of interest, object classification, verification, and refinement. With the adoption of deep learning, several modules are now replaced by standard pre-trained neural network, such as YOLO [187] for object detection, or FCN-8 [138] for segmentation.

The modular approach to perception is favored by the transportation industry because of the engineering advantages provided in terms of robustness, inspectability, and maintenance. However, this approach remains highly complex, inefficient, and—needlessly to say—it has nothing to do with the way humans drive. Therefore, I shall not dwell on this modular account of perception.

4.1.2 Perception without Representations

A different approach that departs radically from the conventional modular computer vision is the *end-to-end* strategy. Instead of decomposing the complex task in explicit sub-tasks, this approach learns the overall task just using examples of the input and corresponding output. This strategy is in line with the empiricist philosophy of artificial neural networks, and it has led to significant advances in complex structured prediction tasks also outside the field of computer vision, like machine translation [112] or speech recognition [5]. In the context of driving, the end-to-end approach can reunite perception with the actions of driving. In fact, a typical end-to-end algorithm learns low-level commands (like steering or braking) from camera images, using generally a stack of convolutions followed by feedforward layers. The first end-to-end attempt dates before the rise of deep learning [154], and it was the groundwork for the renowned NVIDIA's PilotNet [17, 18], which generates steering angles given input images of the road ahead.

Proposers of the end-to-end approach seem unaware of the underlying assumptions this strategy implies. One assumption is the sensorimotor account of vision [164, 160], in which perception is directly related with motor commands. A second assumption is the perspective that internal representations are superfluous, as famously claimed by Rodney Brooks in his *Intelligence without Representation* [23]. These implicit assumptions, together with the dismissal of the modular approach, are the appealing features of the end-to-end philosophy. However, a typical drawback of end-to-end systems based on static frame processing is the erratic variation of steering wheel angle within short time periods. A potential solution is to provide temporal context in the models, combining convolutions with recurrent networks [60]. Another drawback is that end-to-end systems mapping images to steering angles suffer the strict dependence on the specific calibrated actuation setup of the data on which the model is trained. A possible way out is to train from multiple uncalibrated sources, learning to predict future vehicle ego motion, instead of steering angles [240].

Above all, the most appealing feature of the end-to-end strategy—dispensing with internal representations—is also the major source of troubles. Without an internal representation, the entire range of road scenarios has to be learned from steering supervision alone. In practical settings, it is not possible to achieve a significant coverage that takes into account all possible appearances of objects relevant to the drive. For this reason, several more recent proposals suggest to include some form of intermediate representations. The DeepDriving model [28] implements a paradigm called *direct perception*, where the mapping is in terms of "affordances" instead of steering angles; these affordances are just 13 discrete descriptions of possible relations between the ego car and the state of road and traffic. In between the camera input and the affordances, DeepDriving uses as internal representations a set of key perception indicators that directly relate to the affordances. Another model, Waymo's ChauffeurNet [8], proposes the so-called *mid-to-mid* strategy. ChauffeurNet is essentially made of a convolutional network that consumes the input data to generate an intermediate representation with the format of a top-down view of the surrounding area and salient objects. In addition, ChauffeurNet has several higher-level networks which iteratively predict information useful for driving.

There is a further serious issue affecting end-to-end neural models for driving automation: the lack of information about what exactly makes a model arrive to a specific driving command. The refinement over the basic end-to-end approach, here reviewed, can slightly reduce the harshness of the issue on low level commands like steering angles. Still, the total lack of transparency of the models remains a critical problem for applications ruled by strict safety demands such as autonomous driving.

4.1.3 Perception with Representations

In order to overcome the object agnosticism of the end-to-end approach, several strategies have proposed a more dominant use of internal representations. An example is the *object-centric* strategy [231], which combines many neural networks together: a first convolutional network takes an image and produces an intermediate representation; then, a group of downstream networks convert the representation into discrete driving actions. The downstream networks are diversified according to the taxonomy of objects-related structures in the intermediate representation. Another system, by Valeo Vision [218], uses an internal representation constructed with a standard ResNet-50 network [87]. In this case, the representation is shared across a multitude of tasks relevant to vehicle perception, like object detection, semantic segmentation, and depth estimation. All the downstream tasks are realized using standard networks, such as YOLO and FCN-8.

None of the works reviewed so far builds the internal representations through the idea of autoencoder. I have found just few notable exceptions in the field of perception for autonomous driving. The first one is the model by the company *comma.ai* [205], a variational autoencoder learning a latent representation of 2048 neurons and producing images of 160 × 80 pixels. After generating a dataset of latent representations, a recurrent neural network uses the representations to predict successor frames in time. Another exception is a work by Toyota in collaboration with MIT [3], a variational autoencoder with latent space of 25 neurons. The internal representation is decoded to restore the input image of size 200 × 66, while one neuron of the representation is interpreted as steering angle. Therefore, this approach mixes an end-to-end supervision with the classical unsupervised loss of the autoencoder.

A notable work that combines the idea of autoencoder within a cognitive account of prediction is the “world model” proposed by Ha & Schmidhuber [85]. In fact, this work does not deal specifically with perception, it is rather a complete agent including a controller responsible for determining the course of actions. This model has the interesting feature of resembling the imaginative processes, which have a fundamental role in my research as I have already amply discussed in Chapter 2. The model is able to generate “by imagination” new scenarios on which it can train itself in a sort of dreaming mechanism [236]. The model of Ha & Schmidhuber is, however, far too distant from applications in autonomous driving, because of the very shallow perceptual capability. Much like complex neural networks of the past generation, this model is an interesting proof of concept that can only afford basic toy examples. The simple videogame-like scenario used to test the model has an extremely simplified visual appearance, no perspective and very low resolution.

4.2 Datasets for Autonomous Driving

There is a crucial aspect of deep learning—and machine learning in general—that is often considered as one of its most severe drawbacks: the learning is intrinsically linked to the

Dataset	Type	Scenarios	2D boxes	3D boxes	Segmentation	Lane
KITTI [78]	real world	urban traffic	✓	✓	✓	-
Cityscapes [38]	real world	urban traffic	-	-	✓	-
Berkeley DeepDrive [242]	real world	urban traffic	✓	-	✓	✓
Waymo Open Dataset [221]	real world	urban traffic	✓	✓	-	-
Lyft Level 5 Perception [116]	real world	urban traffic	-	✓	-	-
Mapillary Vistas Dataset [157]	real world	urban traffic	-	-	✓	-
nuScenes [26]	real world	urban traffic	-	✓	-	-
Tsinghua-Daimler Cyclist [135]	real world	VRUs	✓	-	-	-
EuroCity Persons [22]	real world	VRUs	✓	-	-	-
SYNTHIA [195]	simulated	urban traffic	-	-	✓	✓
GTA-V [191]	simulated	urban traffic	-	-	✓	-

Table 4.1: A selection of popular datasets for perception in autonomous driving. A dataset is either composed of real-world recordings or generated in computer graphics, and it can either capture generic traffic scenes or focus on vulnerable road users (VRUs). Moreover, a dataset can provide 2D bounding boxes, 3D bounding boxes coming from LiDAR data, semantic and/or instance segmentation, and annotation of lane markings.

data used during training. The performance of a deep learning algorithm depends as much on the neural architecture as it does on the dataset. The quality of a dataset constitutes a key component in the development of an algorithm, and it is mainly determined by the size of the dataset and how well the distribution of the data represents the case at hand.

The application of deep learning to the field of intelligent vehicles has resulted in a significant effort in producing datasets specific for the driving task. The last decade has seen many attempts at creating high-quality datasets of realistic driving scenarios, often curated by the most prominent research groups in the field, like Waymo [221], Toyota [78, 55], Daimler AG [135, 38, 22], just to name a few. Table 4.1 lists a selection of popular datasets, which I am going to describe briefly.

Designing a high-quality dataset for driving automation can be a tricky challenge. As I just mentioned, the quality of a dataset depends on the number of samples and their distribution. In the context of vehicles, these features can be interpreted as the amount of video sequences of driving scenarios and how well the scenarios capture the multitude of possible situations one can encounter while driving. Collecting enough data that meet these quality requirements is extremely time-consuming, and it can easily take several years even with a considerable fleet of vehicles. Note that most perception tasks for autonomous driving require the data to be manually annotated with bounding boxes or semantic descriptions. Moreover, it is far from easy to ensure the acquired videos contain enough different traffic situations and environmental conditions.

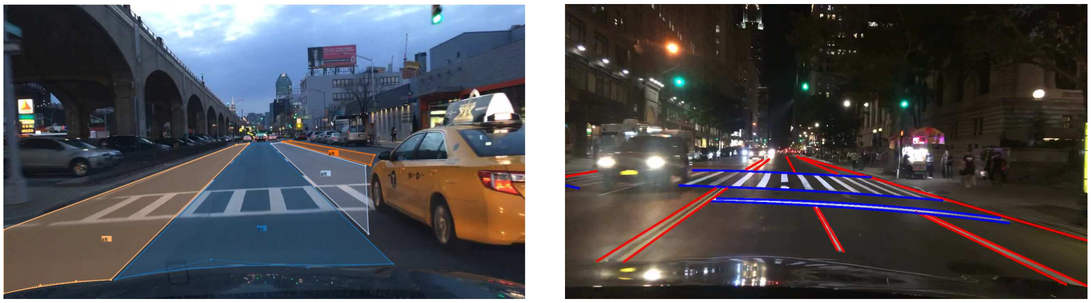


Figure 4.1: Examples of annotations of drivable areas (left) and lane markings (right) provided by the BDD100K dataset [242].

4.2.1 Real-world Datasets

The first notable effort in creating a comprehensive driving dataset is the KITTI Vision Benchmark Suite [78], which is the result of a collaboration between Toyota and the Karlsruhe Institute of Technology. This dataset has since become a pioneering benchmark for most perception tasks, and it has been enriched during the years with more sequences including data coming from multiple sensors. Currently, the benchmark includes different sets of annotated data for specific tasks: optical flow evaluation, depth prediction, 2D and 3D object detection, pixel-level and instance-level semantic segmentation, and multi-object tracking.

Another very popular benchmark, also collected in Germany like KITTI, is the Cityscapes dataset [38] produced by Daimler AG together with the Max Planck Institute for Informatics and the Technical Universities of Darmstadt and Dresden. This dataset focuses on semantic and instance segmentation, and it provides high-quality annotations of 30 different classes of objects. The class definitions adopted by Cityscapes has become a common standard for segmentation in the driving context. A further dataset developed by a research institute is the Berkeley DeepDrive (BDD100K) [242], a recent collaborative effort of UC Berkeley, UC San Diego, and Cornell University. Besides the classic annotations of 2D boxes and semantic/instance segmentation, the benchmark offers a vast set of annotations of lane markings and drivable areas for 100,000 images. Fig 4.1 shows an example of annotated images. This kind of labeled data is uncommon among other popular benchmarks and difficult to find in large amount, so in this sense the Berkeley DeepDrive is an appealing exception.

Several tech companies have joined universities and research institutes in developing new datasets that are increasingly large and rich in annotations. One of the most prominent examples is the Waymo Open Dataset [221], consisting of a set of well synchronized and calibrated high-quality LiDAR and camera data collected across Northern California and Arizona. Also Lyft, the famous California-based company offering vehicles for hire, published a dataset for perception [116] focusing on high-quality LiDAR data. Another noteworthy company is Mapillary, which has developed a service for crowdsourcing street-level images and map data. From this vast collection of images, the company released four labeled datasets, among which the Mapillary Vistas Dataset [157] provides annotations for semantic and instance segmentation, for a total of 25,000 high-resolution images globally spread among 6 continents. A further example is the company Motional (formerly known as nuTonomy), a joint venture between Hyundai and Aptiv. The company released an extensive dataset for 3D object detection and tracking called nuScenes [26], collected using a full sensor suite: five radars, one LiDAR, and six cameras covering a 360° field of view.

The datasets reviewed so far are designed for the task of perceiving and understanding generic urban and suburban traffic scenarios. However, it is worth to mention there are also datasets specialized in safety-critical tasks, like detecting *vulnerable road users* (VRUs). An early example is the Tsinghua-Daimler Cyclist Benchmark (TDCB) [135], which provides stereo images annotated with 2D bounding boxes for cyclist detection. Another comprehensive effort of Daimler AG together with TU Delft produced the EuroCity Persons dataset [22], a collection of accurate annotations of VRUs in urban traffic scenes coming from 12 European countries.

4.2.2 Artificial Datasets

Collecting and annotating such enormous amount of data typically requires years of work. This is one of the reasons why only tech companies or renowned research centers can afford the fleet of vehicles and the workforce needed to create a new dataset. An alternative that has been recently explored is to artificially generate the driving sequences in computer graphics. The advantage of this approach is clear: the number of possible variations in the driving scenarios that one can generate are ideally endless, and the annotation process in this case is totally automatic. Moreover, the recent developments in the field of 3D computer graphics make it possible to generate highly photorealistic images, which facilitates the later deployment of the autonomous system in a real driving context. The main drawback of generating an artificial dataset is that, beside the graphic appearance of the simulation, also the driving behaviors should be realistic. If the ego vehicle and the surrounding vehicles behave in an inconsistent or unnatural way, the quality of the dataset is undermined and the learning of the autonomous agent would be biased.

A prominent example of a dataset created artificially for driving automation is the SYNTHIA dataset [195] for semantic segmentation, proposed by a research group at the Autonomous University of Barcelona. In the case of real-world data for semantic segmentation, the annotation process is particularly cumbersome since pixel-level annotations are required. Therefore, using a virtual world to automatically generate realistic images with pixel-level annotations can really make a difference. However, there remains the problem of designing a full virtual environment and simulating realistic traffic. On this note, a research group from TU Darmstadt and Intel [191] proposed the inventive idea of using a video game as a ready-made generator of labeled data. The researchers adopted the game *Grand Theft Auto V* (GTA-V), which features an extensive and highly realistic world map, and they exploit the communication between the game and the graphics hardware to extract the semantic labels. Note that the realism of the game is not only in the high fidelity of material appearance and textures, it is also in the content of the game world: the layout of objects and environments, the motion of vehicles and autonomous characters, and the interaction between the player and the environment.

An even more sophisticated alternative to recording real driving scenarios is to develop a complete driving simulator. A noteworthy project is the one carried out by the same developers of SYNTHIA together with Toyota and Intel: a highly photorealistic open-source simulator for autonomous driving research called CARLA [55]. The simulator provides open-source code and protocols, and it supports flexible specification of sensor suites, environmental conditions, full control of static and dynamic actors, and maps generation.

4.2.3 Choice of Dataset

In the previous section, I have reviewed a selection of the most popular datasets for perception in driving automation, although recent years have seen lot more research efforts towards creating larger and richer datasets. In the development of my work, I have adopted two of the aforementioned datasets: SYNTHIA and nuScenes. The perception models I will present in Chapters 6 and 7 use SYNTHIA and take advantage of specific annotation of lane markings provided by the dataset, while the models working with higher-lever spaces of Chapter 8 use the nuScenes benchmark. I will further describe the datasets in Sections §6.1.2 and §8.2.1, respectively.

The choice of SYNTHIA as main dataset since the early stages of development of this work was motivated by the availability of lane marking annotations, which, as I have mentioned before, are very rare among the classical datasets for autonomous driving. However, the recent introduction of the BDD100K brings the possibility of choosing this new set of data that has the advantage of featuring real-world recordings. I consider the adoption of the Berkeley dataset a promising addition to my future work. In addition, during the very initial phase of my research, I have created from scratch an essential dataset in computer graphics to test the “primordial” version of my perception model. I will present this dataset in Section §6.1.1.

Chapter 5

Strategies for Observable Models

In the Introduction of this dissertation, I have mentioned the challenges in the research field of autonomous driving. A crucial issue is to demonstrate the reliability of autonomous vehicles when the key components of the system are implemented with artificial neural networks, which suffer from the *black box problem*: it is extremely difficult to explain how a neural network works or why a particular input produces that specific output. In this chapter, I propose a starting point for tackling the black box problem in the application of neural networks for the longitudinal control of an automated vehicle. Although the presented work does not solve the overall problem of intelligibility of deep neural networks, it could be considered a valid contribution in this direction. The next section analyzes the black box problem and the existing literature on the subject. In the remaining sections, I present two different approaches that aim to mitigate this issue.

5.1 Black Box Problem

The cause for the black box problem is the inherent opaque structure of ANNs. The massively distributed and entangled architecture of neural networks is, at the same time, the reason for their striking success and what makes it hardly possible to locate the source of malfunctioning. The literature offers many examples of neural networks of impressive performance behaving in unpredictable or inconsistent ways. A most popular case is the victory of the DeepMind's model AlphaGo over the world champion of the Go game [215]; during the match, AlphaGo played a totally unexpected move, which ultimately proved decisive for the victory. A Go expert attending the match made the following comment:

> "It's not a human move, I've never seen a human play this move" [146].

In that context, the opaqueness of the AlphaGo model turned out to be an advantage over the human opponent. However, the situation was extremely peculiar; in the case of autonomous vehicles, the impossibility of understanding and validating the computations of critical components becomes a crucial issue.

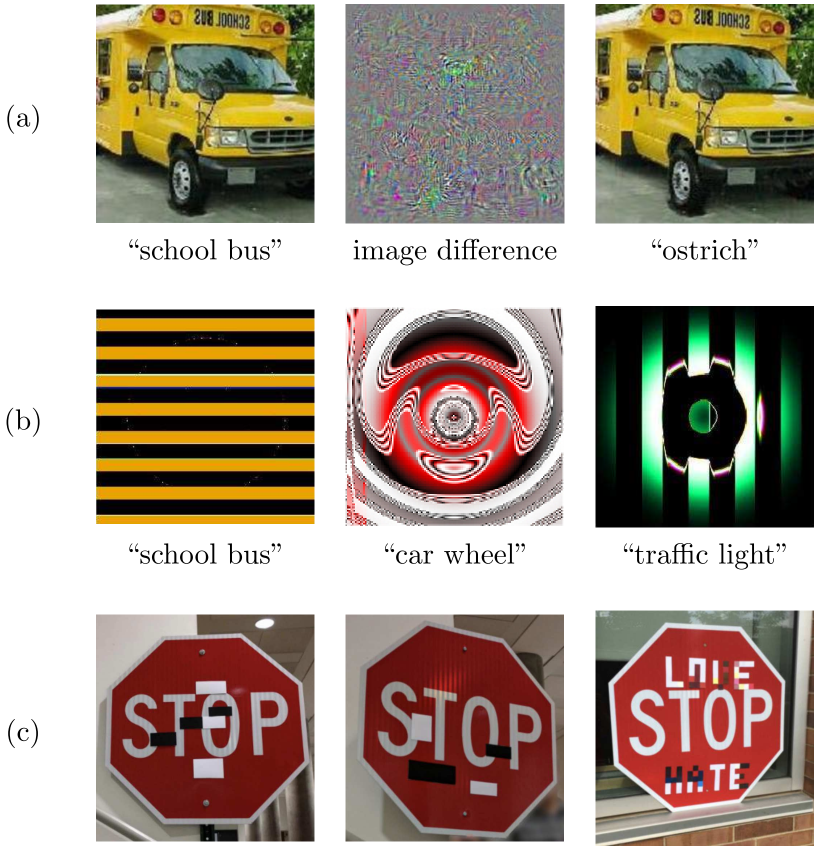


Figure 5.1: Different cases of adversarial examples: (a) an image of a school bus with an imperceptible perturbation makes the AlexNet model classify it as an ostrich [222]; (b) nonsensical images are classified as familiar objects with confidence greater than 99.6% [159]; (c) minimal physical perturbations of stop signs achieve a targeted-attack success rate of 84.8% [62].

In the context of computer vision, there are several notable examples of high-performing networks making glaring mistakes. The work of Szegedy et al. [222] was the first to discover that convolutional neural networks obtaining impressive performance on challenging datasets, such as the model AlexNet trained on ImageNet, can be completely confused in specific situations. By applying an imperceptible non-random perturbation to a test image, it is possible to arbitrarily change the prediction of the network. Similarly, Nguyen et al. [159] generated artificial images that are blatantly meaningless but get classified by the network as familiar objects with very high confidence. In addition, Eykholt et al. [62] experimented with the effect of physical perturbation on real objects; using just black and white stickers, they altered real stop signs causing targeted misclassification in 84.8% of the video frames captured from a moving vehicle. Fig. 5.1 displays some examples of these phenomena. The patterns that trigger this intriguing effect are called *adversarial examples*, and they are currently the object of intense studies. While significant research effort focuses on understanding and overcoming the problem of adversarial examples, there are also works aiming to take advantage of this phenomenon. The most prominent case is the invention of *generative adversarial networks* (GANs) [81], which exploit adversarial examples to carry out a more sophisticated training procedure.

Understanding why a neural network is sensitive to adversarial attacks falls within the larger research field on the explainability of deep neural networks [151, 243]. According to Ras et al. [186], explanation methods can be distinguished in three main classes: rule-extraction approaches, attribution approaches, and intrinsic approaches. The first two groups attempt to *solve* the black box problem by analyzing the neural network so as to render it transparent after it has been deployed. In particular, the rule-extraction methods aim to identify a set of rules that approximate the decisions of the network, while the attribution methods assess how specific components of the network affect the performance. In contrast, the intrinsic approaches attempt to *avoid* the black box problem altogether, by disentangling the internal representations so that the network does not become opaque in the first place. The most common strategy is the attribution methods, specifically the algorithms that automatically identify which cues in the input data lead the network to the corresponding output. In the context of visual perception, Samek et al. [204] proposed an attribution method that generates a heatmap visualizing the importance of each pixel for the classification of the input image.

The rule-extraction and attribution approaches present, however, a major drawback: even if the methods provide interesting insights on the network's behaviour, they are exposed to the very same problem—how to prove the faithfulness of the automatic generated explanations on neural networks [136, 233]. For this reason, the intrinsic methods are considered the most valuable approach, as they tries to make the neural networks inherently more interpretable. The strategies I propose in the following sections belong to the intrinsic class of approaches.

5.2 Towards Gray Boxes

The first strategy for explainable neural networks I propose falls within the context of motor control; in particular, I use a neural network to approximate the longitudinal control of a vehicle. In a nutshell, the idea is to gain intelligibility inside the network by substituting specific scalar values with groups of neurons coding the value of the scalars, restricted within their activation ranges. These groups of neurons work as "channels" encoding scalar values, and they follow a methodology developed outside of the domain of neural networks.

5.2.1 Channel Coding

The idea of "channel coding" was first proposed in 2002 [70, 68] and applied in the field of image enhancement [67]. The general method encodes a scalar x into a N-dimensional vector $\mathbf{c}$:

$$\mathbf{c}(x) = \left[F_1(x) \ \cdots \ F_N(x)\right]^{\mathrm{T}}, \tag{5.1}$$

where F_i represent a family of encoding functions, each indicating how close x is to the value represented by the i-th channel of the vector $\mathbf{c}$. In fact, each F_i reaches the maximum at $x = \frac{i}{N}\left(x_h - x_l\right)$, where x_l and x_h are the lowest and highest possible values of x. Felsberg et al. [67] demonstrated that it is possible to perform operations of image enhancement, such as smoothing and filtering, by encoding the pixels of an image using equation (5.1), manipulating the channels, and then decoding the vector back into pixel values. They formulated the encoding functions F_i as B-spline functions.

I adapt the concept of channel coding to the implementation of a neural network; in this case, the channels aim to expose the values of groups of neurons that represent scalar numbers. For this purpose, I define the encoding functions F_i as follows:

$$F_i(x) = \frac{1}{1 + e^{w(x-b_i)}}, \tag{5.2}$$

where the multiplier w and the offsets b_i are the following:

$$w = \frac{\omega(N-1)}{(x_h - x_l)}, \tag{5.3}$$

$$b_i = \frac{(i-1)x_h + (N-i)x_l}{N-1}, \tag{5.4}$$

where ω is a fixed parameter indicating the amount of overlap between the channels. In the final presented work, I have adopted $\omega = 2.7$ and $N = 11$. This definition of encoding function means that for every $x \in [x_l, x_h]$ it holds the following condition:

$$F_i(x) \geq F_{i+1}(x) \qquad i \in [1, N-1]. \tag{5.5}$$

The different formulation of channel coding produces a cumulative activation of the channels, rather than a local activation as in the original implementation. Fig. 5.2 illustrates the difference between the two formulations: in the classic channel coding, the scalars activate only few channels around the corresponding value; in my implementation, the activation is cumulative and the channels are progressively saturated towards the corresponding value of the scalar.

An important property of the new formulation is that F_i is expressed in terms compatible with the computation of a neural network. In fact, equation (5.2) applies the sigmoid function to the scalar number, which is modified with an offset and a weight. In this way, it is possible to apply channel coding to the inputs and outputs of a generic neural network, simply using an additional neural layer. A neural layer for channel coding has two

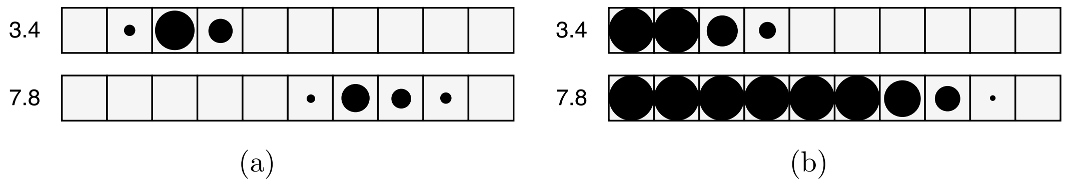


Figure 5.2: Difference between the original idea of channel coding (a) and my implementation (b). Both cases encode a scalar in the range $[0 \cdots 10]$ using 10 channels. The examples show a visual representation of the encoding of the numbers 3.4 and 7.8.

differences with respect to an ordinary layer: it is not fully connected, and its weights and biases remain fixed during learning. The interesting effect of channel coding on the neural network is that the channel layers are no longer opaque. The neurons now have a precise meaning: they directly encode the value of scalar numbers. Of course, this holds only for the neurons belonging to the channel layers, as the rest of the network remains not fully explainable. Therefore, I do not claim that channel coding solves the black box problem in its entirety, but at least it renders the black "less black" and produces partially intelligible networks, which I like to call *gray boxes* [178].

Beside the aspect of explainability, channel layers provide additional advantages depending on whether they are applied to the input or the output of the network. Applying channel coding to the input layer has attractive implications when the network takes multiple inputs. If there is prior knowledge on the role of each input in the objective of the neural model, it is possible to assign different relevance to the inputs by varying the number of channels in each case. It is possible to allocate a larger number of channels to the inputs that have more importance in the network computation, and a smaller number of channels to the less important inputs. This facilitates the training process because the network does not have to learn by itself which input to put more emphasis on. Hence, input channel layers prove to be a practical method for providing prior knowledge to the model.

On the other hand, using channel coding in the output layer offers an advantage from the point of view of reliability. Whether the network works as a function approximator or as a discrete classifier, there is always the possibility that it might commit a prediction error causing a spurious activation of output neurons. Even the failure of a single neuron can affect the entire output of the network, with potentially dangerous consequences. Note that a neural network typically applies an activation function to the last layer so as to constrain the output range. However, it can still happens that a network failure causes abrupt jumps between the boundaries of the accepted output range. Applying channel coding to the output layer can further mitigate the risk; in this case, the spurious activation of a single neuron is not possible because the channels are activated in a cumulative way, as in Fig. 5.2(b). Therefore, output channel layers ensure that prediction errors are less likely to have dangerous consequences.

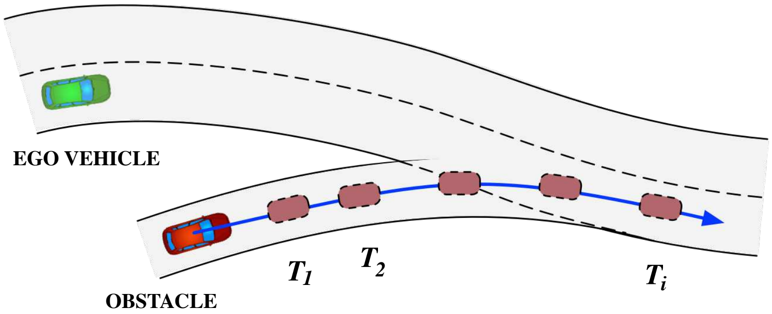


Figure 5.3: Schematic representation of the computation of collision trajectories from the prediction of the obstacle's path.

5.2.2 Application to Longitudinal Control

To illustrate an example of application to a safety-critical task, I present a neural network using channel coding for computing collision trajectories in the longitudinal dimension.

Computing Collision Trajectories

The computation of collision trajectories can be formulated as an inverse dynamics problem. In this way, the problem can be solved by finding the longitudinal controls that generate the trajectories leading to the collision. Let s_T be the longitudinal position of the ego vehicle at time T, displayed as the green car in Fig. 5.3. Similarly, $\hat{s}_T$ represents the position the obstacle vehicle at time T, corresponding to the red car in the figure. The trajectory leading to a collision at time T is determined by the longitudinal control that produces the condition $s_T = \hat{s}_T$. The optimal way to express the longitudinal control is the jerk j, i.e., the time derivative of acceleration. As I will better discuss in Section §8.4, the minimum square jerk criterion is known to be a valid approximation of optimal control in human sensorimotor strategies [137]. Therefore, it is possible to compute the longitudinal collision trajectory by minimizing the following equation:

$$J = w\, a_T^2 + \int_0^T j(t)^2 \, dt, \tag{5.6}$$

where T is the time necessary for the obstacle to reach the position $\hat{s}_T$, and a_T is the acceleration of the ego vehicle at time T. The parameter w characterizes the driving style, by weighting the final acceleration. Assuming the initial condition $s_0 = 0$ and imposing the final condition $s_T = \hat{s}_T$, the analytical solution to the minimization of J is the following:

$$j = \frac{90 s_T + (60 w s_T - 90 v_0)\, T - (45 a_0 + 60 w v_0)\, T^2 - 24 w a_0\, T^3}{(9 + 4wT)\, T^3}, \tag{5.7}$$

where v_0 and a_0 are the initial speed and acceleration of the ego car.

Note that the solution supposes that the prediction of the obstacle's trajectory is given. For the purpose of this work, it does not matter how the trajectory prediction is obtained. This approach has the advantage that separating the trajectory prediction from the collision computation facilitates a potential troubleshooting. In fact, in case of failure, it is possible to assess more easily which of the two processes is responsible. Conversely, if both processes are performed in a single step, the system would inevitably be less inspectable and likely prone to error. Moreover, in this work I am considering only the longitudinal dynamics, but the same approach can potentially be applied to the lateral dimension as well. By combining the two computations it is possible to determine the complete collision trajectory.

Neural Network Implementation

Having defined the problem, now I present a solution using neural networks. I compare three network implementations: an ordinary fully-connected network, a network using channel coding in input, and a network using channel coding in both input and output. In all the cases, the neural network aims to approximate the function (5.7), and its computation can be described as follows:

$$\mathcal{N}(a_0, v_0, s_T, T, w) = \tilde{\jmath}, \tag{5.8}$$

$$\mathcal{N} : \Omega \subset \mathbb{R}^5 \to \mathbb{R}, \tag{5.9}$$

where $\tilde{\jmath}$ is the predicted optimal jerk returned by the neural network $\mathcal{N}$. The inputs of the network are the current longitudinal speed v_0 and acceleration a_0 of the ego vehicle, the position s_T where the ego vehicle will collide with the obstacle at time T, and the parameter w setting the driving style. The domain Ω of the network function is a well-defined hypercube, because the ranges of input values can be determined precisely based on the vehicle specifications. In the current case, I have considered the following ranges:

$$\begin{aligned} a_0 &\in [-10, 10], \\ v_0 &\in [0, 50], \\ s_T &\in [0, 200], \\ T &\in [0, 20], \\ w &\in [0, 10]. \end{aligned}$$

Conversely, the co-domain of the function poses some challenges in relation to the wide range of values it can assume. In fact, there could be combinations of inputs in Ω that produce extreme values of jerk. Since the output might vary by several orders of magnitude, the co-domain must be considered within the range $[-\infty, +\infty]$. However, I am interested in designing a network that provides fine control in real-world conditions, i.e., when $\tilde{\jmath}$ is

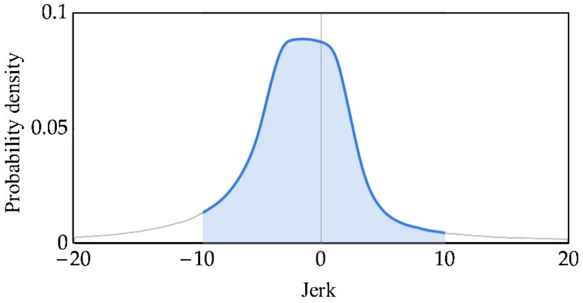


Figure 5.4: Distribution of jerk values generated for the training set.

relatively small. For this purpose, let $\tilde{\Omega}$ be a subset of the domain considering only the input combinations corresponding to ordinary jerk values:

$$\tilde{\Omega} = \left\{ [a_0, v_0, s_T, T, w]^{\mathrm{T}} : \; j \in [-10, 10] \right\}.$$

To train and test the neural networks, I have created two artificial datasets. For the training set, I have generated 750,000 input points uniformly distributed in $\tilde{\Omega}$, and I have computed the corresponding outputs using the analytical solution (5.7). Fig. 5.4 shows the distribution of the jerk values in the training set. Similarly, I have generated a test set of another 1 million points, but considering the entire Ω this time. By including the entire domain in the test set, it is possible to assess how the network operates also in the extreme conditions represented by $\Omega - \tilde{\Omega}$.

The first neural network I present does not employ channel coding, as it serves as comparison for the next implementations. The network is composed of an input vector of 5 neurons representing $\{a_0, v_0, s_T, T, w\}$, two fully-connected layers of 55 neurons each followed by ReLU activations, an output layer of 1 neuron representing $\tilde{\jmath}$. Fig. 5.5(a) shows the network prediction versus the target value in the test set. The blue interval refers to $\tilde{\Omega}$, while the orange part refers to $\Omega - \tilde{\Omega}$, in which there are no training examples. In $\Omega - \tilde{\Omega}$, the network tends to correctly extrapolate but with increasing errors.

The second neural network adopts channel coding in input. The overall number of layers and neurons in the network is the same as the previous implementation. However, the first fully-connected layer is transformed into a channel layer and uses a logistic sigmoid as activation function. In this case, the 55 neurons of the layer are divided into groups of 11 neurons, and each group encodes one of the 5 input scalar separately. The weights and biases of the channel layer are not learned during the training, but are derived using the encoding functions in equation (5.2). As example, Fig. 5.6 displays the encoding of the acceleration input into the 11 channels. Compared to the previous implementations, this network has less parameters to learn. Reducing the number of trainable parameters may seem counterintuitive; however, channel coding proves to be an effective way to regularize the network structure. The structured pattern imposed to the first layer makes the weights

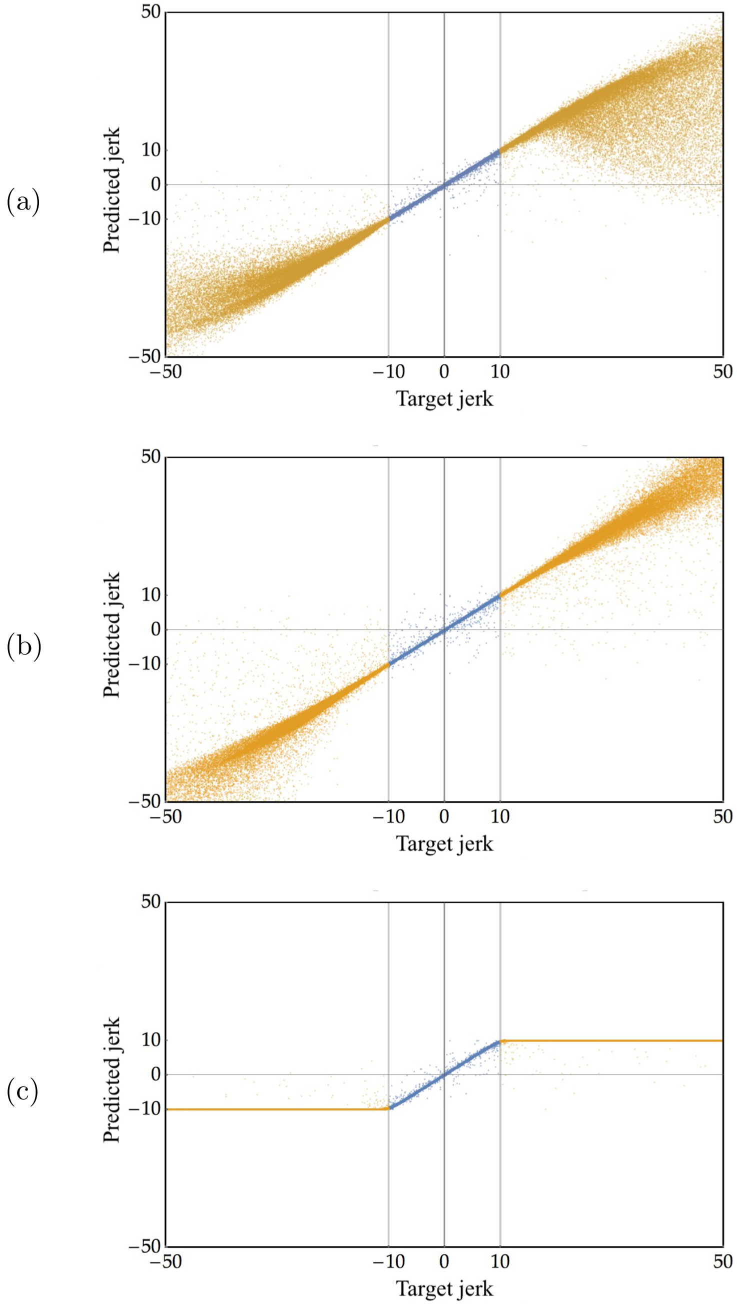


Figure 5.5: Test results of the three implementations of neural network for longitudinal control: (a) simple fully-connected, (b) channel coding in input, (c) channel coding in input and output. The blue points belong to $\tilde{\Omega}$, and the orange points refer to $\Omega - \tilde{\Omega}$.

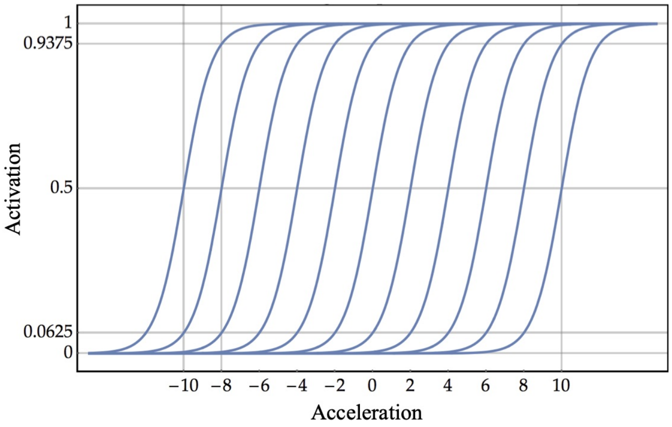


Figure 5.6: Example of channel coding for the acceleration input a_0. The scalar is encoded with 11 channels (neurons), each responding to a different interval of the input range.

	RMSE	% severe errors
Standard	0.088	0.036
Input channels	0.095	0.039
Input/output channels	0.071	0.018

Table 5.1: Prediction errors of the three network implementations. The metrics are the root mean square error and the percentage of test samples in $\tilde{\Omega}$ for which $|\tilde{\jmath} - j| > 1$ ms^{-3}.

of the second layer more interpretable, as they are directly associated to particular subintervals of the input vector. Fig. 5.5(b) shows the performance of the network. It is evident how in the domain $\Omega - \tilde{\Omega}$, where there are no training examples, the predictions improve significantly with respect to the standard network. This confirms that using channel coding even just in the input is beneficial to the network.

The third and final network for longitudinal control adopts channel coding in both input and output. Starting from the same architecture of the previous network, the learnable fully-connected layer is followed by a new channel layer of 11 neurons, which are decoded into the scalar value of jerk. Once again, the weights and the biases of this layer are designed *ad-hoc* and fixed during the training. Fig. 5.5(c) shows the prediction of the network. This time, the network responses in the domain $\Omega - \tilde{\Omega}$ are clipped at the maximum and minimum saturation values of jerk. To give a more precise evaluation of the performance inside the domain $\tilde{\Omega}$ representing the most probable conditions, Table 5.1 reports the prediction errors of the three networks, in terms of root mean square error and percentage of test samples for which the prediction error is greater than 1 ms^{-3}. It is immediate to notice

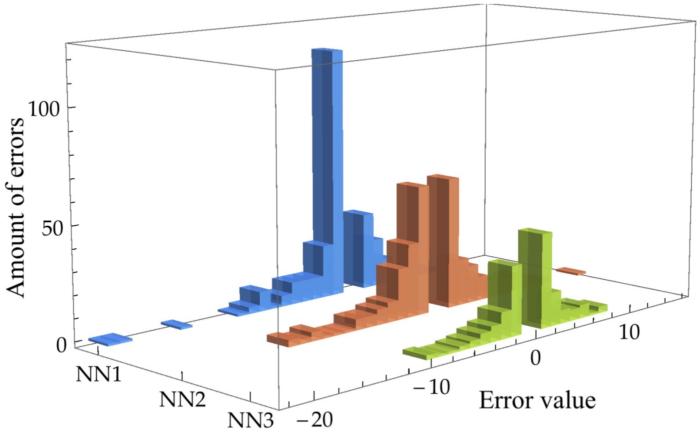


Figure 5.7: Distribution of the prediction errors larger than 1 ms^{-3} for the three networks: the standard network in blue, the network with input channels in orange, and the network with input/output channels in green. The histogram bins are 1 ms^{-3} wide.

that the scores improve significantly with the third implementation. In addition, Fig. 5.7 plots the distribution of the "severe errors" in $\tilde{\Omega}$ for the three implementations. Again, the network with channel layers in input and output shows the most compact histogram, which is limited in height and also less expanded towards the largest error values.

With a simple implementation example, this work demonstrates that it is possible to deploy a reliable neural network for a safety-critical application as the computation of collision trajectories. By applying the concept of channel coding to the output of the network, no spurious activations of neurons can possibly produce dangerous values of jerk. At the same time, using channel coding in the input layer has the double effect of regularizing the network—thus improving the overall performance—and exposing the internal layer, which becomes more interpretable.

5.3 Semantic Spaces

In the preceding section, I have illustrated an approach aiming to mitigate the black box problem exploiting the idea of channel coding. The approach successfully manages to render the external layers of the network intelligible, producing "gray boxes". However, this is clearly just a partial solution. A more sophisticated approach should focus on the explainability of the innermost layer of a network, the one capturing the most abstract representations of the data.

Considering in particular the architecture of autoencoders, I propose a second method aiming to learn intelligible representations of the driving scenario. I want the representations to bear a semantic explanation, in the sense that parts of the latent vector are associated with specific concepts related to the driving context. Albeit partitioning the representations may look as an expedient, this idea is related to the notion of topographic organization largely present in the brain, where similar concepts are encoded in close groups of neurons [219, 224, 170]. In contexts different from autonomous driving, the idea of assigning conceptual meaning to separate groups of neurons in the latent representation is not new. For example, a work [128] from the computer vision community proposed a method to generate head poses using a latent space with separate representations for viewpoints, lighting conditions, and shape variations. Similarly, another work [245] partitioned the latent vector into semantic content and geometric coding.

My second approach to mitigate the black box problem aims to partition the internal representations into distinct concepts related to driving. The network achieves the conceptual organization by adopting a multi-decoder structure learned using semantic segmentation as a supporting task. To demonstrate the effectiveness of this approach, in the next Chapters 6 and 7, I will present four different neural networks with increasingly sophisticated implementations oriented to improve the intelligibility of the internal representations for the driving context.

Chapter 6

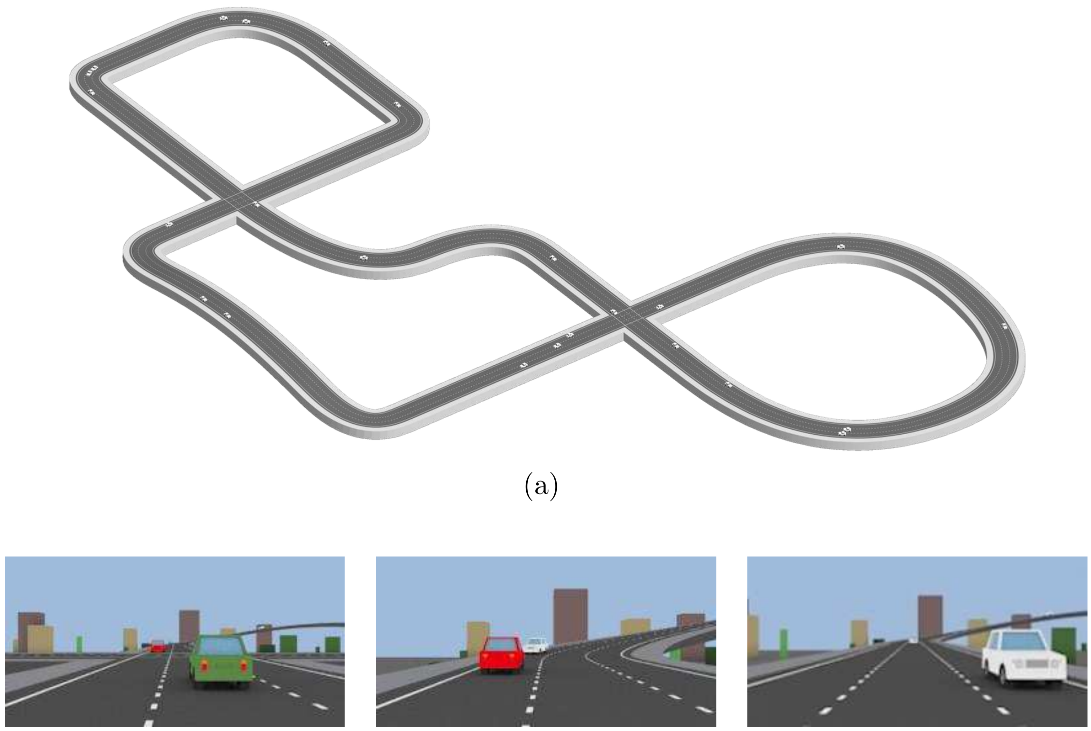
Static Models

Strictly speaking, there is no static perceptual processing in the brain [71, 160]. In the case of the visual stimulus, however, the ratio of the spatial resolution with respect to the temporal dimension is much higher than other stimuli, like the auditory one. In image processing, visual tasks are ordinarily approximated as series of static processing of still images. This approximation has been extremely fruitful in image processing, and it finds some support in natural vision, too [144]. There is also ample evidence that deep neural models for static image processing are particularly well suited to be extended for dynamic processing [15, 202]. Therefore, I find it convenient to decompose the development of my models into a first static phase and a second dynamic phase, aware that this is just a convenient approximation with respect to the brain neural process.

This chapter describes the first group of models developed for static image processing of driving scenes. The first section portrays the datasets of driving scenarios used by the static models. The second section presents the implementations of the two static models. Then, the third section clarifies the role of segmentation in my overall approach. The final section summarizes the obtained results.

6.1 Datasets

I have presented in Section §4.2 an overview of some of the most popular and recent datasets for perception in autonomous driving. Since I want my models to focus on learning simple and essential concepts of the driving context, namely vehicles and lanes, I choose the SYNTHIA dataset as the most suitable for this task. As I have mentioned in §4.2.2, SYNTHIA is one of the few large-scale datasets providing lane marking annotations, which are required for the task of learning the `lanes` concept. Besides this, at the beginning of my doctoral research, I have created a minimal dataset of video sequences to train an essential neural model and become familiar with the architecture of the autoencoder. In the following, I will give a detailed description of both datasets.



(a)

(b)

Figure 6.1: Driving environment of the Blender dataset: (a) orthographic view of the road track; (b) three samples from the video sequences in the dataset.

6.1.1 Blender Dataset

During the first stage of my work on autoencoders and neural networks in general, I have been interested in investigating how much the quality of a dataset can affect the performance of a neural network. I have found that the most effective way to study this aspect is to create a custom dataset from scratch using a computer graphics software. To date, Blender[1] is the most popular and comprehensive open-source 3D suite available. It features an extensive Python API and a strongly structured user interface [181], which makes it an excellent tool for scientific research involving 3D simulations. In the past, I have adopted Blender more than once for my research, for example, to simulate fire outbreaks in industrial plant using particle physics [180], and to optimize the design of interior lighting using genetic algorithms [182, 174].

Using Blender, I have created a virtual road track and generated video sequences of

[1] `www.blender.org`

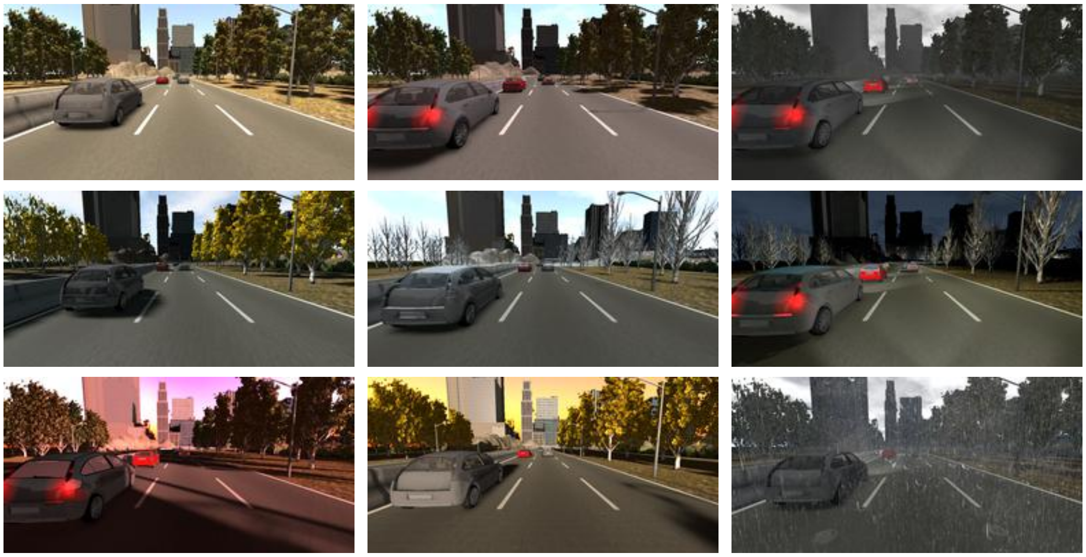


Figure 6.2: Samples from one of the tracks in the SYNTHIA dataset. The images show the results of rendering the same view using different environmental and lighting conditions.

driving scenarios. Fig. 6.1(a) shows an orthographic view of the road track, which is composed of three lanes where cars can drive in both directions with variable speed. To ensure enough variety, the track features a succession of curves of different radius alternated with straight segments, and it also includes slopes and intersections. The road track is placed in a minimal urban environment, with simple buildings surrounding the track. The virtual camera is placed on the windshield of the ego car, which drives along the track in a loop and performs random changes of lane. Fig. 6.1(b) gives an example of the recordings collected by the ego camera.

I have adopted this dataset in the development of the deterministic versions of the static models presented in Sections §6.2.1 and §6.2.2. I have incrementally added the aforementioned features of the track in parallel with the development of the first prototype of autoencoder. In this way, I have been able to determine how the variety of the dataset was impacting on the learning process of the network. I will detail the development steps in Section §6.2.1.

6.1.2 SYNTHIA Dataset

The SYNTHIA dataset [195] consists of a large collection of photo-realistic video sequences rendered using the game engine Unity. It comprises about 100, 000 images of urban scenarios recorded from a simulated camera placed on the windshield of the ego car. Each video sequence is acquired at 5 FPS and comes with semantic annotations or several classes—

including lane markings, which are not commonly found in other datasets.

Despite being artificially generated, this dataset offers a wide variety of reasonably realistic illumination and weather conditions, occasionally even resulting in very adverse driving conditions. The dataset features 5 sets of driving sequences; each set contains about 10 recordings of the same track rendered under different environmental conditions: traffic, weather, season, and time of the day. Fig. 6.2 gives an example of the variety of data coming from the same driving sequence with different conditions. Moreover, the tracks are very diverse as well, including freeways, tunnels, congestion, “NewYork-like cities”, and “European towns” (as the creators of the dataset describe it).

I have adopted SYNTHIA in the development of the non-deterministic versions of the static models presented in Sections §6.2.1 and §6.2.2, and the dynamic models in Sections §7.1.1 and §7.1.2. I randomly allocate 70% of the video sequences to the training set, 25% to the validation, and 5% to the test set, ensuring no overlap among the three sets. For a more interesting visualization of the results, I further organize the test set into four (overlapping) categories based on the driving scenarios: urban environments, freeways, sunny conditions, and conditions with darkness or adverse weather.

6.2 Models

This section presents the implementations of the static perceptual neural networks. The networks aim to generate compact representations of visual scenarios without taking into account the temporal dimension. I have experimented with two different architectures, both sharing the common feature of a hierarchical arrangement similar to the brain CDZs and in line with the strategies illustrated in Sections §3.3.2 and §3.3.4.

6.2.1 *Net1*: Simple Autoencoder

The first model I present here is the most essential. It is composed of two sub-networks: an encoder g_Φ and a decoder f_Θ, as in equations (3.4) and (3.5). The first version of the network was a deterministic autoencoder, trained without supervision using as loss function the *mean squared error* (MSE). Later, I have transformed the model into a variational autoencoder, adopting the loss function of equation (3.12). Fig. 6.3(a) depicts the architecture of the model, and Table 6.1 shows the numbers of layers and the parameters adopted in the final version of the model. The input of the network is a single RGB image of 256×128 pixels. The encoder is composed of a stack of 4 convolutions and 2 fully-connected layers, converging to a latent space of 128 neurons. The decoder has a structure symmetric to the encoder, mapping the 128 neurons back to an image of 256×128.

During an initial “prototyping stage”, I have trained the network on the Blender dataset, introduced in §6.1.1. I have improved the dataset in parallel with the development of the network. In this way, I have realized how the quality of the dataset affects the performance of the model as much as the definition of the hyperparameters of the

Encoder	convolution	$7 \times 7 \times 16$
	convolution	$7 \times 7 \times 32$
	convolution	$5 \times 5 \times 32$
	convolution	$5 \times 5 \times 32$
	dense	2048
	dense	512
Latent space		128
Decoder	dense	2048
	dense	4096
	deconvolution	$5 \times 5 \times 32$
	deconvolution	$5 \times 5 \times 32$
	deconvolution	$7 \times 7 \times 16$
	deconvolution	$7 \times 7 \times 3$
Total parameters		18 million

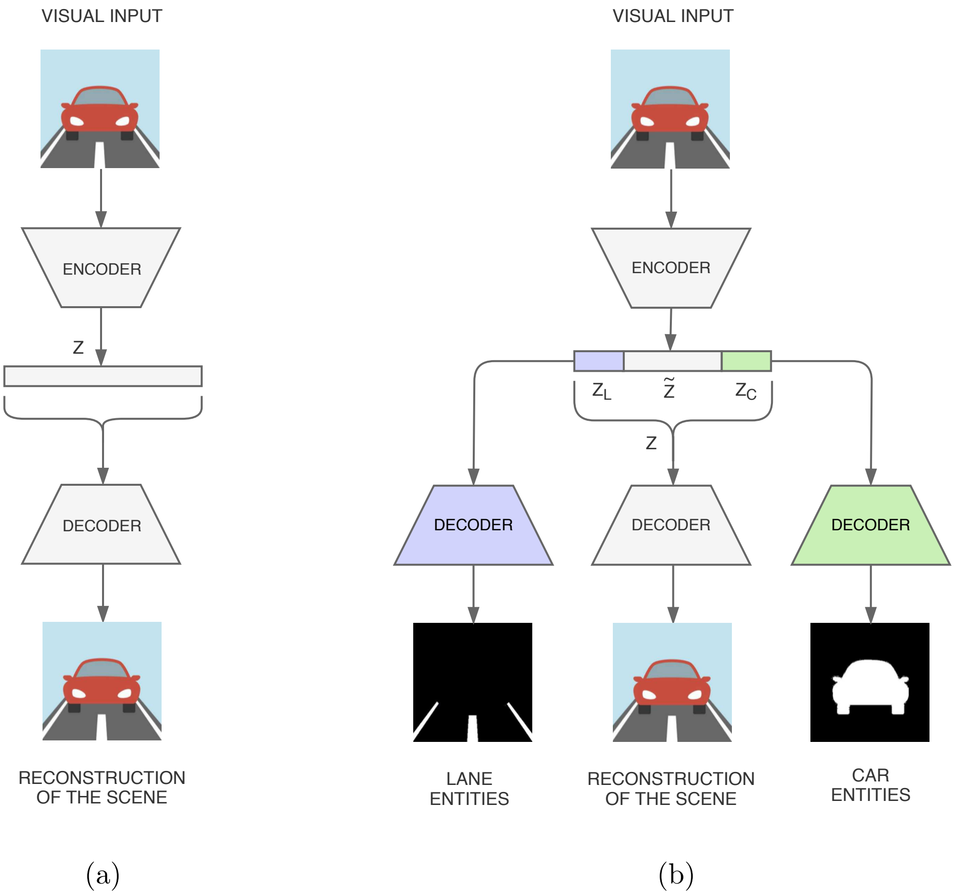
Table 6.1: Parameters describing the architecture of *Net1*.

network (e.g., number of layers, size of kernels, or learning rate). At first, the virtual road track had only two lanes and the vehicles were driving in a single direction. For this reason, the initial network was not able to capture the silhouette of a car driving in the opposite direction. Therefore, I have added a third lane and randomly assigned the driving direction of each vehicle. Another issue was that the network got used to the fixed point of view of the camera with respect to the road and the horizon. Hence, I have adopted two expedients: random changes of lane to ensure the camera observes the road from all the three lanes; slopes in the road, so that the horizon is not fixed at the same height of the camera frame during all the video sequences.

After I got acquainted with the functioning of the deterministic autoencoder and switched to the variational autoencoder, I have replaced the Blender dataset with SYNTHIA, described in §6.1.2. This dataset is undoubtedly more complex and accurate, so I have been able to fully test the capabilities of the neural network.

6.2.2 *Net2*: Conceptual Autoencoder

The following model shares most of its architecture with the previous model. The crucial improvement is the introduction of a semantic organization in the latent spaces. As discussed in Section §2.2, the human brain projects sensory information—especially visual—into compact representations through the CDZ structures. Some of these representations constitute the *conceptual space*, where neural activations encode the entities in the environment that produced the perceptual stimuli. It is possible to take inspiration from this



(a)

(b)

Figure 6.3: Comparison between the architectures of *Net1* (a) and *Net2* (b), where the green color denotes the `cars` concept, and violet the `lanes` concept.

theory and use the hierarchical architecture of the CDZs as a “blueprint” to design a more sophisticated neural network, which can learn representations that are not only in terms of visual features but also in terms of useful *concepts* [179, 175, 176].

Note that, in the driving context, the entire road scenario is informative. However, from a conceptual point of view, it is not immediately necessary to infer categories for every entity present in a scene. Within the aims and limits of this work, it is more effective to project in conceptual space the entities mostly relevant to the driving task. In this work, for the sake of simplicity, I have considered the two main concepts of `cars` and `lanes`.

Fig. 6.3(b) presents the architecture of this “conceptual” autoencoder, composed of one shared encoder and three independent decoders. The choice of parameters is similar to *Net1*, as Table 6.2 shows. The encoder and each of the three decoders maintain the same structure as in *Net1*, and the size of the latent space remains unchanged. Still, the

Encoder	convolution	$7 \times 7 \times 16$
	convolution	$7 \times 7 \times 32$
	convolution	$5 \times 5 \times 32$
	convolution	$5 \times 5 \times 32$
	dense	2048
	dense	512
Latent space		$[96, 16, 16]$
Each decoder	dense	2048
(conceptual | visual)	dense	4096
	deconvolution	$5 \times 5 \times 32$
	deconvolution	$5 \times 5 \times 32$
	deconvolution	$7 \times 7 \times 16$
	deconvolution	$7 \times 7 \times (1 \vert 3)$
Total parameters		35 million

Table 6.2: Parameters describing the architecture of *Net2*.

internal organization of the latent space is forcefully partitioned. The gray decoder of Fig. 6.3(b) works in the visual space—just like the decoder of *Net1*—mapping all the 128 neurons of the latent vector $\mathbf{z}$ altogether back into an RGB image. This decoder learns to reconstruct the input image and is trained in an unsupervised way. On the other hand, the decoder colored in green takes only a sub-vector $\mathbf{z}_{\mathrm{C}}$ of 16 neurons from the latent space and produces a matrix $\mathbf{x}_{\mathrm{C}}$ of 256×128 probability values. The sub-vector of 16 neurons is trained to represent the `cars` concept, and the output matrix can be interpreted as a semantic segmentation of the input image, where values indicate the probability of the presence of `cars` entities. Similarly, the violet decoder maps only a sub-vector $\mathbf{z}_{\mathrm{L}}$ of 16 neurons representing the `lanes` concepts into a probability matrix $\mathbf{x}_{\mathrm{L}}$ for `lanes` entities. These two decoders require supervised learning; their output is converted into binary images by applying a threshold, and they are trained to minimize the reconstruction error with semantic segmentation of the input images. Note that segmentation here can be considered a mere byproduct of the network, as the goal remains the meaningful latent representations—I will further discuss this aspect in Section §6.2.3. To give a mathematical description of the model, it is composed of four sub-networks:

$$g_{\Phi} \; : \; \mathcal{X} \to \mathcal{Z}, \tag{6.1}$$

$$f_{\Theta_{\mathrm{V}}} \; : \; \mathcal{Z} \to \mathcal{X}, \tag{6.2}$$

$$f_{\Theta_{\mathrm{C}}} \; : \; \mathcal{Z}_{\mathrm{C}} \to \mathcal{X}_{\mathrm{C}}, \tag{6.3}$$

$$f_{\Theta_{\mathrm{L}}} \; : \; \mathcal{Z}_{\mathrm{L}} \to \mathcal{X}_{\mathrm{L}}, \tag{6.4}$$

with $\mathcal{Z} = \mathbb{R}^{N_{\mathrm{V}}}$, $\mathcal{Z}_{\mathrm{C}} = \mathbb{R}^{N_{\mathrm{C}}}$ and $\mathcal{Z}_{\mathrm{L}} = \mathbb{R}^{N_{\mathrm{L}}}$. The subscript V denotes the visual space, and the subscripts C and L refer to the `cars` and `lanes` conceptual spaces respectively. For each latent vector $\mathbf{z}$ we have:

$$\mathbf{z} \in \mathcal{Z} = \left[\, \widetilde{\mathbf{z}}, \mathbf{z}_\mathrm{C}, \mathbf{z}_\mathrm{L} \,\right], \tag{6.5}$$

where $\mathbf{z}_\mathrm{C}$ and $\mathbf{z}_\mathrm{L}$ are the two sub-vectors representing the `cars` and `lanes` concepts. The remaining segment $\widetilde{\mathbf{z}}$ encodes the rest of the generic visual features, while the entire latent vector $\mathbf{z}$ is a representation in the visual space. The final version of the model has $N_\mathrm{V} = 128$ and $N_\mathrm{C} = N_\mathrm{L} = 16$; I will discuss this choice in Section §6.3.

The loss function of this model can be derived from the basic formulation of equation (3.12). As in the case of *Net1*, I have initially implemented *Net2* in the deterministic form, using the Blender dataset for training, However, the final version of the model adopts the variational architecture and is trained on SYNTHIA. By calling $\Theta = [\Theta_\mathrm{V}, \Theta_\mathrm{C}, \Theta_\mathrm{L}]$ the vector of parameters of all decoders, at each batch iteration b a random batch $\mathcal{B} \subset \mathcal{D}$ is presented, and the following loss is computed:

$$\mathcal{L}(\Theta, \Phi|\mathcal{B}) = E_\mathrm{K} + \lambda_\mathrm{V} E_\mathrm{V} + \lambda_\mathrm{C} E_\mathrm{C} + \lambda_\mathrm{L} E_\mathrm{L}, \tag{6.6}$$

where

$$E_\mathrm{K} = \left(1 - (1-k_0)\kappa^b\right) \sum_{\mathbf{x}}^{\mathcal{B}} \Delta_\mathrm{KL}\big(q_\Phi(\mathbf{z}|\mathbf{x}) \| p_{\Theta_\mathrm{V}}(\mathbf{z})\big), \tag{6.7}$$

$$E_\mathrm{V} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x})} \left[\log p_{\Theta_\mathrm{V}}(\mathbf{x}|\mathbf{z})\right], \tag{6.8}$$

$$E_\mathrm{C} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}_\mathrm{C}\sim \Pi_\mathrm{C}(q_\Phi(\mathbf{z}|\mathbf{x}))} \left[\log \widetilde{p}_{\Theta_\mathrm{C}}(\mathbf{x}_\mathrm{C}|\mathbf{z}_\mathrm{C})\right], \tag{6.9}$$

$$E_\mathrm{L} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}_\mathrm{L}\sim \Pi_\mathrm{L}(q_\Phi(\mathbf{z}|\mathbf{x}))} \left[\log \widetilde{p}_{\Theta_\mathrm{L}}(\mathbf{x}_\mathrm{L}|\mathbf{z}_\mathrm{L})\right]. \tag{6.10}$$

Few observations are due for the differences between this loss function (6.6) and the basic formulation (3.12). First of all, here I apply a delay in the contribution of the Kullback-Leibler divergence in the term E_K. This strategy is called *KL annealing* and was first introduced in the context of variational autoencoders for language modeling [21]. The motivation for this technique is that the encoder at the beginning of training is unlikely to provide any meaningful probability distribution $q_\Phi(\mathbf{z}|\mathbf{x})$. Hence, there is a cost factor for the KL component, which is set initially at a small value k_0 and gradually increased up to 1.0 with a time constant κ. A second difference is the terms $E_\mathrm{V}, E_\mathrm{C}, E_\mathrm{L}$: they represent the reconstruction errors of the visual scenario and the conceptual entities. The term E_V computes the error in the visual space using the entire latent vector $\mathbf{z}$, and it corresponds precisely to the second component in the basic loss (3.12). The other two terms E_C and E_L compute the error in the conceptual space and are slightly different; only the relevant portion of the latent vector is considered, as symbolized by the projection operators Π_C and Π_L.

Another difference of the loss function is the use of a variant of the cross entropy in equations (6.9) and (6.10), indicated with the symbols $\widetilde{p}_{\Theta_\mathrm{C}}$ and $\widetilde{p}_{\Theta_\mathrm{L}}$. This variant takes into account the large unbalance between the number of pixels belonging to one of the concepts and all the other pixels—a typical situation in ordinary driving scenes. Following a method first introduced in the context of medical image processing [220], I compensate this asymmetry by weighing the contribution of true and false pixels with the ratio P of true pixels over all the pixels in the dataset, computed as follows:

$$P = \sqrt[s]{\frac{1}{NM}\sum_{j}^{M}\sum_{i}^{N} y_{i,j}}, \tag{6.11}$$

where M is the number of images in the dataset, and N is the number of pixels in an image. The parameter s is used to smooth the effect of weighting by the probability of ground truth; a value evaluated empirically as valid is 4. The term $y_{i,j}$ is the value of the i-th pixel (in a flatten order) of the j-th target image of the dataset. I use a different set of target images for each semantic concept. Hence, I have a set of `car` labels composed of binary images where white pixels indicate the presence of cars in the scene, and a set of `lane` labels where white pixels correspond to lane markings. Lastly, in the loss equation (6.6) the contributions of the terms $E_\mathrm{V}, E_\mathrm{C}, E_\mathrm{L}$ are weighted by the parameters $\lambda_\mathrm{V}, \lambda_\mathrm{C}, \lambda_\mathrm{L}$. The purpose of these parameters is mainly to normalize the range of the errors, which varies widely from visual space to conceptual spaces. Hence, I set $\lambda_\mathrm{V} \neq \lambda_\mathrm{C} = \lambda_\mathrm{L}$.

I have mentioned in Section §5.3 that the idea of partitioning the latent vector into semantic components is not new. However, my approach is different: while I keep the two segments $\mathbf{z}_\mathrm{C}$ and $\mathbf{z}_\mathrm{L}$ disjointed, the entire $\mathbf{z}$ learns representations in the visual space. That is why the gray decoder of Fig. 6.3(b) takes as input the entire latent space. Another advantage of my approach concerns the well-known crucial issue of lack of transparency in deep neural networks, amply discussed in Chapter 5. This method mitigates the issue by explicitly assigning semantic meaning to the components of the inner representation.

6.2.3 Role of Segmentation

When looking at the results of the next section, for example Fig. 6.5, it may seem that the outcome of the proposed models is essentially image segmentation. Image segmentation is the process of partitioning an image into meaningful subsets. It has been one of the popular tasks in classical image processing [155, 212, 150] and continues to be a major topic in the era of deep learning for computer vision [6, 29, 148]. However, image segmentation has limited relevance to my research; even if the outputs of the networks here presented indeed include the segmentation of cars and lanes, this is not the objective of my work.

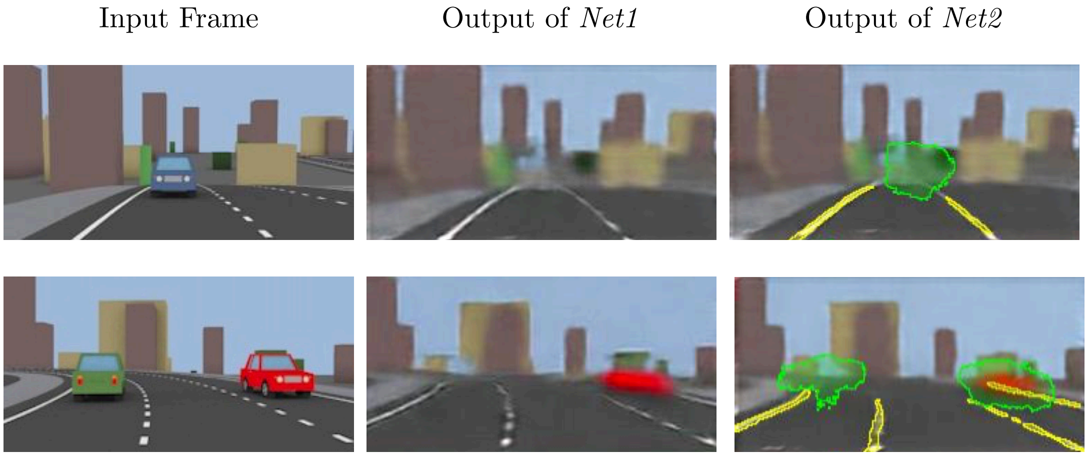


Figure 6.4: Comparison between the deterministic implementations of *Net1* and *Net2*, on the Blender dataset.

The models presented here and in the next chapter aim to learn representations of the driving scenario that can be exploited for imagination in the driving context. I want these representations to be, first of all, meaningful. The representations must bear a semantic explanation, i.e., parts of the latent space are associated with concepts useful in the context of driving—`cars` and `lanes` in this case. The models learn these meaningful representations by exploiting semantic segmentation as a supporting task, using the multi-decoder architecture described in Section §6.2.2, which forces the partitioning of the internal representations into distinct concepts. In this context, segmentation can be considered just a practical way to achieve the separation of the semantic concepts in the latent space. Hence, semantic segmentation is simply a byproduct of my overall approach and not its primary focus.

6.3 Results

Here I illustrate the results obtained by the two perceptual models *Net1* and *Net2*, which are both trained for 200 epochs. First, I present some qualitative comparisons between the networks. Fig. 6.4 compares the outputs of the networks produced from two input samples of the Blender dataset. For this comparison, I have used the deterministic implementations of both models. Note that the output of *Net2* is actually a vector of three images: one reconstructing the visual scene, and the other two segmenting the `cars` and `lanes` elements in the scene. For an easy visualization, I display the three outputs as a single image, where the background is the reconstruction in the visual space and the colored overlays are segmented entities—`cars` in green/cyan and `lanes` in yellow. The results of *Net1* (central column) decently reconstruct most of the scene, including the landscape in the background.

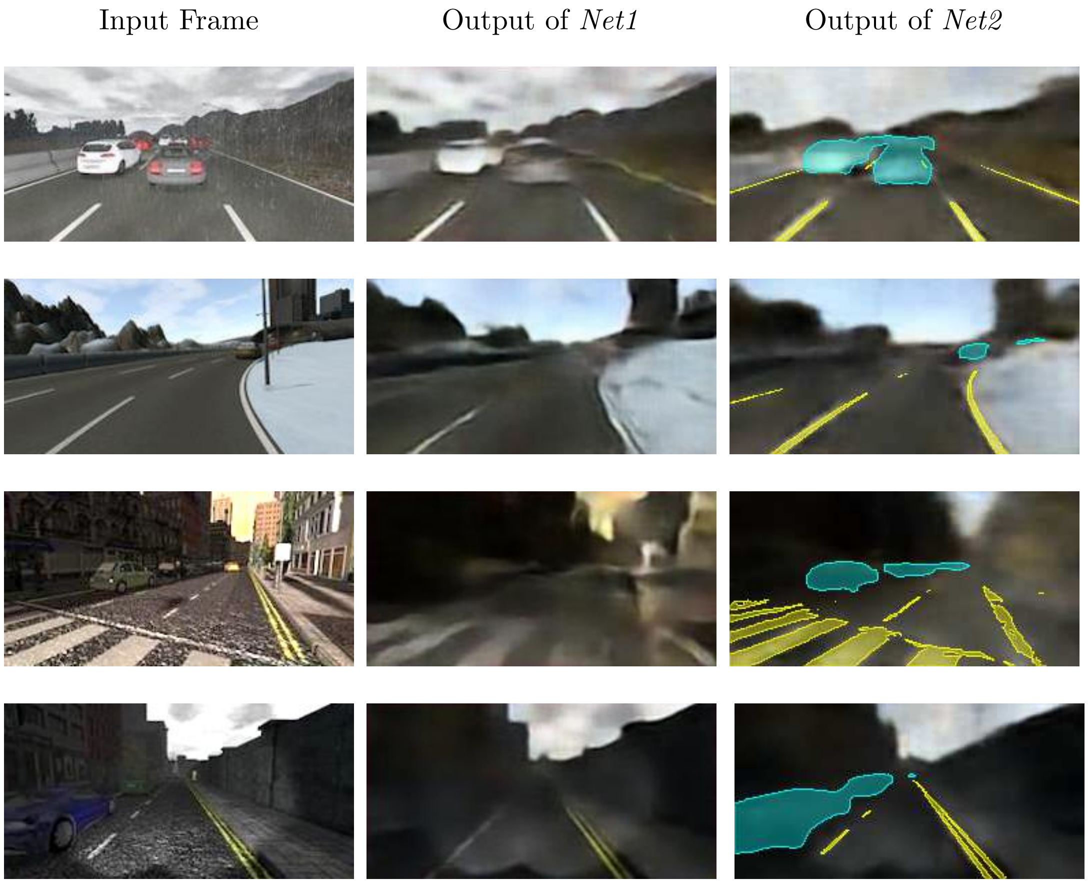


Figure 6.5: Comparison between the variational implementations of *Net1* and *Net2*, on the SYNTHIA dataset.

However, the network fails to capture the features that change faster than the surroundings and appear more rarely—the cars. This is what happens in the samples of Fig. 6.4, where the blue and green cars disappear almost completely. On the other hand, the results of *Net2* (right column) demonstrate the advantage of having a latent space semantically organized. Although the visual reconstruction of the scenario is still not sensible to some rapidly changing features, the specialized decoders do not fail to capture the conceptual entities. The segmented output correctly detect the lane markings and the cars, even if the segmented silhouettes are not very precise.

Fig. 6.5 presents a similar comparison, but it considers the variational implementations of the networks and is performed on the SYNTHIA dataset. It is evident the variational *Net1* performs better than the deterministic *Net1* in detecting the moving vehicles. Nonetheless, *Net2* prevails again in extracting with precision the conceptual entities. This

Learning rate	IoU car	IoU lane
1×10^{-2}	0.0000	0.0000
1×10^{-3}	0.7383	0.6368
5×10^{-4}	0.7391	0.6599
$\mathbf{1 \times 10^{-4}}$	**0.7702**	**0.6277**
5×10^{-5}	0.7584	0.6083
1×10^{-5}	0.7086	0.5502
1×10^{-6}	0.1187	0.1734

Table 6.3: Performance of *Net2* using different values of learning rate. The final choice adopted in the model is marked in bold.

$N_{\mathrm{C}} = N_{\mathrm{L}}$	IoU Cars	IoU Lanes
48	0.7814	0.6460
32	0.7768	0.6334
24	0.7709	0.6440
16	**0.7702**	**0.6277**
12	0.7539	0.6139
8	0.7194	0.5965
4	0.6162	0.5123

Table 6.4: Performance of *Net2* using different numbers of neurons for the `cars` and `lanes` concepts in the latent space, while keeping the overall size $N_{\mathrm{V}} = 128$. The final choice adopted in the model is marked in bold.

is especially true in the scenarios with adverse lighting conditions, like the last two rows of Fig. 6.5: *Net1* is not sensible to the vehicles in the shadows, while the specialized decoders of *Net2* are able to segment the `cars` entities with considerable accuracy.

Moving to a quantitative evaluation, I now discuss the choice of two important hyperparameters of *Net2*. Table 6.3 presents how the learning rate affects the reconstruction of the `cars` and `lanes` entities. I measure the goodness of the results using the *intersection over union* (IoU) metrics separately for each concept. In the final version of the network, I have chosen to favor the `cars` concept. The rationale for this decision is that an error in the detection of a vehicle can have much more serious consequences than a similar error for the lane markings. Moreover, in a sequence of frames, the lane markings change appearance more slowly than moving vehicles. Therefore, detecting an anomaly in the reconstruction of the `lanes` entities results easier than in the case of `cars`; that is why the `cars` concept requires more accuracy.

Table 6.4 shows the impact of the sizes N_{C} and N_{L} on the performance of *Net2*. N_{C} and N_{L} are the number of neurons in the latent space representing the `cars` and `lanes` concepts respectively, as defined in §6.2.2. In the final version of the model, I have set

$N_{\mathrm{C}} = N_{\mathrm{L}} = 16$, even if this does not correspond to the best IoU score. The reason I prefer having a latent representation of concepts as compact as possible is twofold: first, with a lower dimensionality, I force the model to capture the absolutely essential features from the data, discarding the non-relevant information; second, if the representation of a single concept occupies only a small fraction of the entire latent space, the model can learn several different concepts at the same time. Here, I have decided to assign 16 neurons to each concept with the idea that, in future works, I can use the same architecture to learn more than two concepts, adding for example `pedestrians` and `bikes`. Therefore, the final model adopts the most compact size not causing a severe drop in the performance, unlike in the cases of $N_{\mathrm{C}} = N_{\mathrm{L}} < 12$. I will present more results on *Net1* and *Net2* in Section §7.2 when I will compare them with *Net3*, a conceptual variational autoencoder that integrates the temporal dimension in the generation of the latent representations.

Chapter 7

Dynamic Models

After presenting the models tackling the static problem, it is now the turn of extending to the dynamic problem, where the input are temporal sequences of driving scenes. This chapter is organized in a first section describing the model implementations and illustrating the use of self-supervision in my approach. The second section concludes with the results achieved.

7.1 Models

This section presents the implementations of the two neural models taking into account the temporal dimension: a perceptual network and a predictive network. The perceptual network is the third step in the development of a model learning compact and informative representations of driving scenarios. The predictive network is an example of how the representations can be exploited for various downstream driving tasks: in this case, the prediction of long-term future frames in a video sequence.

7.1.1 *Net3*: Temporal Autoencoder

The following model is the final development of an autoencoder able to learn meaningful representations of the driving scenario. My work aims to learn representations oriented to the driving task from a static and a dynamic perspective [177]. In *Net2*, I have implemented the static perspective, i.e., a conceptual organization of the latent representations. *Net3* adds the dynamic perspective by forcing a temporal consistency in the representations.

The model learns how the concepts represented in the latent space will change in future driving scenarios. Note, however, that this model can predict only short-term windows, whereas longer-term predictions will be the subject of *Net4*. The model achieves representations consistent in the temporal dimension by including a recursive module in the architecture of *Net2* and using self-supervision—the use of self-supervision is detailed in

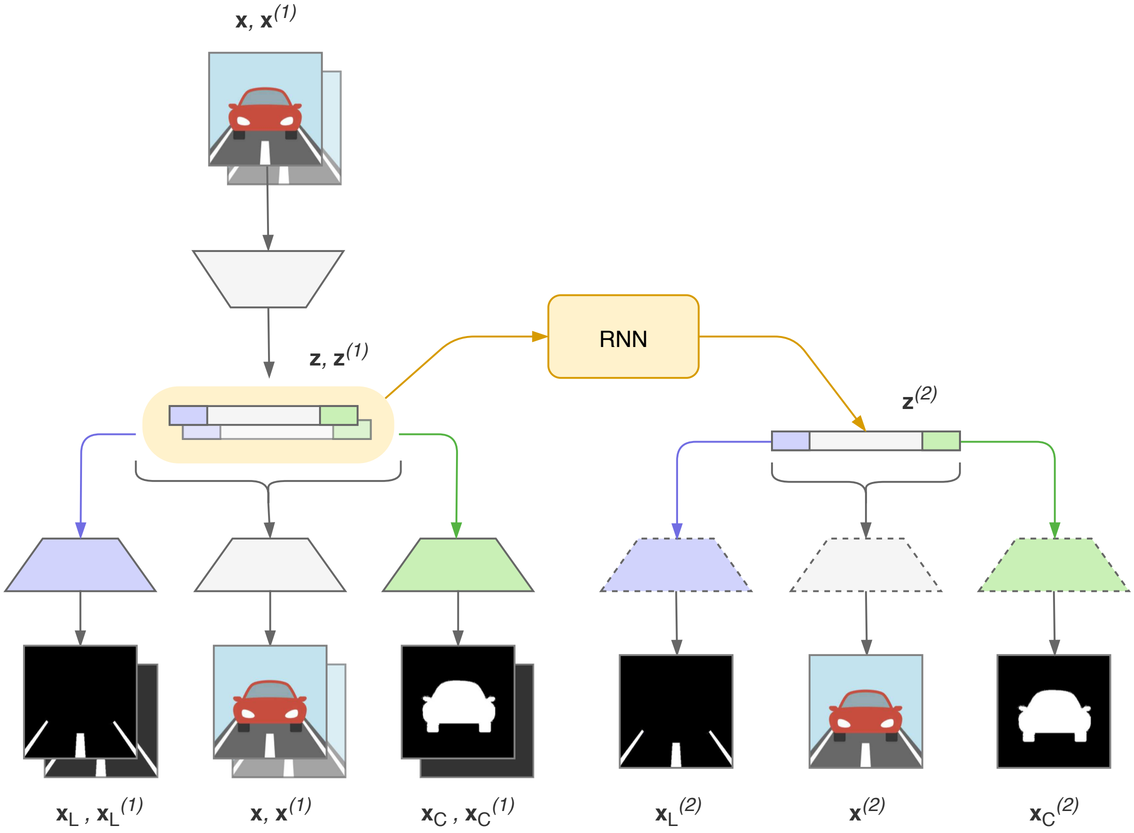


Figure 7.1: Architecture of *Net3*. The green color denotes the `cars` concept and violet the `lanes` concept. The decoders with dashed-line border represent the same instances of the decoders with solid-line border.

Section §7.1.3. Fig. 7.1 shows the architecture of *Net3*, and Table 7.1 describes the parameters used in the final implementation. The dataset of choice is again SYNTHIA. The model shares substantially the same architecture of *Net2*, except for an additional module based on a simple recursive neural network (RNN) [59], and a training procedure significantly different from the previous network.

To better explain how the training works, let me introduce the notation $\mathbf{x}^{(t)}$ to indicate the image frame t steps ahead of frame $\mathbf{x}$. Similarly, $\mathbf{z}^{(t)}$ refers to the latent representation of the frame t steps ahead of the frame represented by $\mathbf{z}$. At each iteration of the training, the model receives as input two consecutive frames $\mathbf{x}$ and $\mathbf{x}^{(1)}$. A common encoder processes the frames and computes two latent representations $\mathbf{z}$ and $\mathbf{z}^{(1)}$. Then, a RNN takes the latent vectors and predicts $\mathbf{z}^{(2)}$, which represents the successive frame in the sequence. Finally, a 3-decoders structure (the same of *Net2*) expands $\mathbf{z}$, $\mathbf{z}^{(1)}$, and $\mathbf{z}^{(2)}$ into conceptual and visual images. To sum up, at each iteration, the inputs of the model are $\mathbf{x}$ and $\mathbf{x}^{(1)}$, while the outputs are the visual and segmented images for $\mathbf{x}$, $\mathbf{x}^{(1)}$, and $\mathbf{x}^{(2)}$. The newly introduced recursive module is implemented using a basic RNN with a time window of 2 and a set of parameters Ψ, and it is described by the following function:

$$h_\Psi\left(\mathbf{z}, \mathbf{z}^{(1)}\right) \to \widetilde{\mathbf{z}} \approx \mathbf{z}^{(2)}. \tag{7.1}$$

The formulation of the loss function is similar to equation (6.6) except for two additional terms for the recursive prediction:

$$\mathcal{L}(\Theta, \Phi, \Psi|\mathcal{B}) = \ \mathcal{L}(\Theta, \Phi|\mathcal{B}) + E' + E'', \tag{7.2}$$

where the first term is the loss of eq. (6.6) and the additional terms are defined as follows:

$$E' = \lambda'_{\mathrm{V}}\, E'_{\mathrm{V}} + \lambda'_{\mathrm{C}}\, E'_{\mathrm{C}} + \lambda'_{\mathrm{L}}\, E'_{\mathrm{L}}, \tag{7.3}$$

$$E'' = \lambda''_{\mathrm{V}}\, E''_{\mathrm{V}} + \lambda''_{\mathrm{C}}\, E''_{\mathrm{C}} + \lambda''_{\mathrm{L}}\, E''_{\mathrm{L}}. \tag{7.4}$$

The expressions of the remaining terms are the following:

$$E'_{\mathrm{V}} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x}^{(1)})}\left[\log p_{\Theta_{\mathrm{V}}}\left(\mathbf{x}^{(1)} \,|\, \mathbf{z}\right)\right], \tag{7.5}$$

$$E'_{\mathrm{C}} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}_{\mathrm{C}}\sim \Pi_{\mathrm{C}}(q_\Phi(\mathbf{z}|\mathbf{x}^{(1)}))}\left[\log \widetilde{p}_{\Theta_{\mathrm{C}}}\left(\mathbf{x}_{\mathrm{C}}^{(1)} \,|\, \mathbf{z}_{\mathrm{C}}\right)\right], \tag{7.6}$$

$$E'_{\mathrm{L}} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}_{\mathrm{L}}\sim \Pi_{\mathrm{L}}(q_\Phi(\mathbf{z}|\mathbf{x}^{(1)}))}\left[\log \widetilde{p}_{\Theta_{\mathrm{L}}}\left(\mathbf{x}_{\mathrm{L}}^{(1)} \,|\, \mathbf{z}_{\mathrm{L}}\right)\right], \tag{7.7}$$

$$E''_{\mathrm{V}} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x})}\left[\log p_{\Theta_{\mathrm{V}}}\left(\mathbf{x}^{(2)} \,|\, h_\Psi\left(\mathbf{z}, g_\Phi\left(\mathbf{x}^{(1)}\right)\right)\right)\right], \tag{7.8}$$

$$E''_{\mathrm{C}} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x})}\left[\log \widetilde{p}_{\Theta_{\mathrm{C}}}\left(\mathbf{x}_{\mathrm{C}}^{(2)} \,|\, \Pi_{\mathrm{C}}\left(h_\Psi\left(\mathbf{z}, g_\Phi\left(\mathbf{x}^{(1)}\right)\right)\right)\right)\right], \tag{7.9}$$

$$E''_{\mathrm{L}} = -\sum_{\mathbf{x}}^{\mathcal{B}} \mathbb{E}_{\mathbf{z}\sim q_\Phi(\mathbf{z}|\mathbf{x})}\left[\log \widetilde{p}_{\Theta_{\mathrm{L}}}\left(\mathbf{x}_{\mathrm{L}}^{(2)} \,|\, \Pi_{\mathrm{L}}\left(h_\Psi\left(\mathbf{z}, g_\Phi\left(\mathbf{x}^{(1)}\right)\right)\right)\right)\right]. \tag{7.10}$$

The contributions of the terms $E'_{\mathrm{V}}, E'_{\mathrm{C}}, E'_{\mathrm{L}}$ is similar to that of $E_{\mathrm{V}}, E_{\mathrm{C}}, E_{\mathrm{L}}$, as they represent the errors in the reconstruction of the frame successor of $\mathbf{x}$. The temporal coherence is measured by the terms $E''_{\mathrm{V}}, E''_{\mathrm{C}}, E''_{\mathrm{L}}$, which represent the error between the images decoded from the latent vector predicted by h_Ψ and the targets relative to the frame 2 steps ahead of $\mathbf{x}$.

Once the training is completed, the network used for inference discards the recurrent module and reverts to the same architecture of *Net2*. Note that the purpose of the network

Encoder	convolution	$7 \times 7 \times 16$
	convolution	$7 \times 7 \times 32$
	convolution	$5 \times 5 \times 32$
	convolution	$5 \times 5 \times 32$
	dense	2048
	dense	512
Latent space		$[96, 16, 16]$
Recurrent layer		$128 \times 2 \rightarrow 128$
Each individual decoder (conceptual | visual)	dense	2048
	dense	4096
	deconvolution	$5 \times 5 \times 32$
	deconvolution	$5 \times 5 \times 32$
	deconvolution	$7 \times 7 \times 16$
	deconvolution	$7 \times 7 \times (1\|3)$
Total parameters		35 million

Table 7.1: Parameters describing the architecture of *Net3*.

is still to learn perceptual representations of driving scenarios and not to predict in the future, which is instead the aim of *Net4*. Therefore, there is no need for the recurrent module during inference, as the parameters of the encoder have already captured the information on the temporal aspect.

7.1.2 *Net4*: Recurrent Network

The last network I present is an example of how the results obtained by the previous model can be exploited to perform long-term prediction of driving scenarios. In the previous Sections §6.2.1, §6.2.2, and §7.1.1 I have described the three steps towards a perceptual model able to encode a visual scenario into representations that are conceptually organized and temporally consistent. *Net3* is the final result of this development.

The following network works exclusively with the latent representations. Once the final training of *Net3* was completed, I have deployed its encoder to generate a dataset of latent vectors from the frames in SYNTHIA—*Net4* is trained with this new dataset. In this way, the long-term prediction can be realized entirely within the latent space. Moreover, having a compact representation allows the recurrent network to have a complex architecture with a limited number of parameters.

Fig. 7.2 shows the architecture of the network. It is composed of two modules: the first module consists of multiple levels of stacked recurrent sub-networks; the second module comprises multiple parallel recurrent sub-networks predicting successive latent vectors in the sequence. In the first module, each stacked sub-network sends its entire output sequence to the next sub-network input. In the second module, instead, the parallel sub-networks

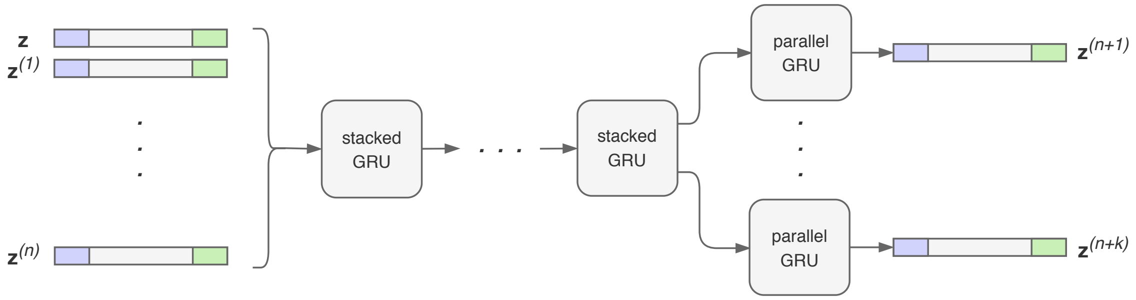


Figure 7.2: Architecture of *Net4*, where the green color denotes the `cars` concept and violet the `lanes` concept.

Stacked recurrency	GRU	$128 \times 8 \to 128 \times 8$
	GRU	$128 \times 8 \to 128 \times 8$
Parallel recurrency	GRU	$128 \times 8 \to 128$
	GRU	$128 \times 8 \to 128$
	GRU	$128 \times 8 \to 128$
	GRU	$128 \times 8 \to 128$
Total parameters		600,000

Table 7.2: Parameters describing the architecture of *Net4*.

yield only the last output in the time sequence. All the sub-networks of the model share the same core architecture implemented with *gated recurrent units* (GRUs) [34]—I will discuss this implementation choice in Section §7.2. The overall model is described by the following function:

$$r_\Xi : \mathcal{Z}^{N_\mathrm{I}} \to \mathcal{Z}^{N_\mathrm{O}}, \tag{7.11}$$

$$r_\Xi \left(\mathbf{z}, \mathbf{z}^{(1)}, \cdots, \mathbf{z}^{(N_\mathrm{I}-1)} \right) \to [\, \widetilde{\mathbf{z}}_1, \widetilde{\mathbf{z}}_2, \cdots, \widetilde{\mathbf{z}}_{N_\mathrm{O}}] \approx \left[\mathbf{z}^{(N_\mathrm{I})}, \mathbf{z}^{(N_\mathrm{I}+1)}, \cdots, \mathbf{z}^{(N_\mathrm{I}+N_\mathrm{O})} \right], \tag{7.12}$$

where N_I is the length of the input sequence, N_O is the length of the future sequence to be predicted, and Ξ is the set of parameters of the model. In the final implementation, I have set $N_\mathrm{I} = 8$ and $N_\mathrm{O} = 4$, and I have used 2 stacked GRUs and 4 parallel GRUs, as described in Table 7.2. Lastly, I want to highlight that this model does not use any odometry or other kind of information for the prediction, just the rich representations learned by the accompanying autoencoder.

7.1.3 Role of Self-supervision

Having clarified with Section §6.2.3 the role of segmentation in my work, now I discuss the connection with another important machine learning domain called *self-supervision*. Unlike unsupervised learning, self-supervision is not motivated by biological plausibility; it is instead a way around the ever-present issue of manual data labeling in large datasets of images [111, 158]. Usually, self-supervision is realized by designing pretext tasks without any particular relevance for the agent but useful for the automatic generation of pseudo-labels. While learning to solve the pretext tasks, the model is forced to capture certain visual features of images that are ideally useful for the core task of the agent.

The computer vision community has proposed several kinds of creative pretext tasks for self-supervision. A prevalent task is *colorization* [131], where a color image is first converted to graylevel, and the model learns to reconstruct the color version. Another kind of task is solving jigsaw puzzles made from patches of the input image [161]. There are also self-supervision tasks that are indeed useful to the overall objective of the model, but the labeling is assumed by analytical methods [33]: a common example is the exploitation of the epipolar constrains in the stereo image pair as supervision for training a monocular image depth estimation model [30].

On the other hand, a small number of approaches exploit *prediction* as a self-supervision task. My models adopts this idea, using prediction of future frames to bias the internal representation towards the ability to learn the dynamics of objects in the scene. In this sense, prediction for self-supervision shows a connection with the cognitive idea of predictive brain I have discussed before in Section §2.4. Moreover, besides having a well-structured internal organization, the representations learned by my models have a second important feature: they can be exploited for imagery, the mental process introduced in Section §2.1.2. Imagery can result from a latent representation of a scenario seen before, or it can be triggered by a prediction of a future scenario based on past ones. It can also results from manipulating a latent space, generating scenarios the model has never seen before.

Still, not all approaches maintain a sound cognitive account of self-supervising prediction in the context of vision. For example, a recent work [229] arranges images in overlapping blocks by rows and columns, scanned in sequence with recursive networks attempting to "predict" the next block. This account of prediction is clearly an artifact with no correspondence in a cognitive agent. Instead, my work aims to include effective forms of prediction: prediction as imagination, and prediction as the construction of a probable future scenario.

One of the few works based on a cognitive account of prediction is the model proposed by Ha and Schmidhuber [85], which I have already briefly discussed in Section §4.1.3. This model shares some fundamental components with my architectures: the use of variational autoencoders and recursive neural networks. There is, however, a significant difference in the objectives of the models. The work of Ha and Schmidhuber is a complete agent and includes other components not considered in my models, like a controller responsible

	Net2		**Net3**		**FCN-8**		**U-Net**	
	IoU car	IoU lane	IoU car	IoU lane	IoU car	IoU lane	IoU car	IoU lane
City	0.7834	0.6487	0.8305	0.7155	0.8033	0.6109	0.8552	0.7451
Freeway	0.7755	0.5840	0.7952	0.7490	0.7587	0.6959	0.7975	0.8666
Sunshine	0.7736	0.6283	0.8077	0.6970	0.7741	0.6652	0.8351	0.8128
Darkness	0.7682	0.6274	0.7943	0.7116	0.7450	0.6385	0.7914	0.7927
All	0.7702	0.6277	0.7992	0.7062	0.7558	0.6484	0.8076	0.8001

Table 7.3: Comparison of the advanced autoencoders *Net2* and *Net3* with other popular models for semantic segmentation. The scores are divided into `cars` and `lanes` classes, and they are organized into the four categories of driving conditions.

for determining the course of actions of the agent. Their wider architecture comes at the expense of a very shallow perceptual capability. Much like complex neural networks of the past generation, this model is an interesting proof of concept working in synthetic simplified examples. The simple game-like scenario on which the model was tested has an overly simplified visual appearance, not using perspective and with very low resolution. Conversely, my aim is not to train an agent but to learn the perceptual capability needed for visual imagery, including the projection of hypothetical driving scenarios in visual space.

7.2 Results

Here I illustrate the results of the two dynamic models together with some comparisons with the static models presented in Chapter 6. In the final implementations, both dynamic models use the SYNTHIA dataset, *Net3* is trained for 200 epochs, and *Net4* for 100 epochs.

First, I present the results achieved by *Net3*. Fig. 7.3 shows the images produced from four different frames of the test set, one for each of the driving categories I considered, as described in §6.1.2. Just like the results of the static models, I display the output of the three decoders as a single image to facilitate understanding of the results. In addition, for a practical reference, the right column of the figure displays the target images with the same colored overlays. Although *Net3* is capable of predicting a successive frame in the future, as I have mentioned in §7.1.1, the objective of the network is to encode visual scenarios into compact abstract representations. Therefore, Fig. 7.3 shows the result of *Net3* receiving a single input image, compressing it into a latent vector, and expanding it back into visual and conceptual images. The results are satisfying in all four conditions, even when the illumination is particularly adverse, like in Fig. 7.3(b) and (d). The network is always able to detect the vehicles with enough precision, including the very distant cars. Moreover, the network has learned to recognize complex patterns of lane marking and pedestrian crossing, as in Fig. 7.3(a).

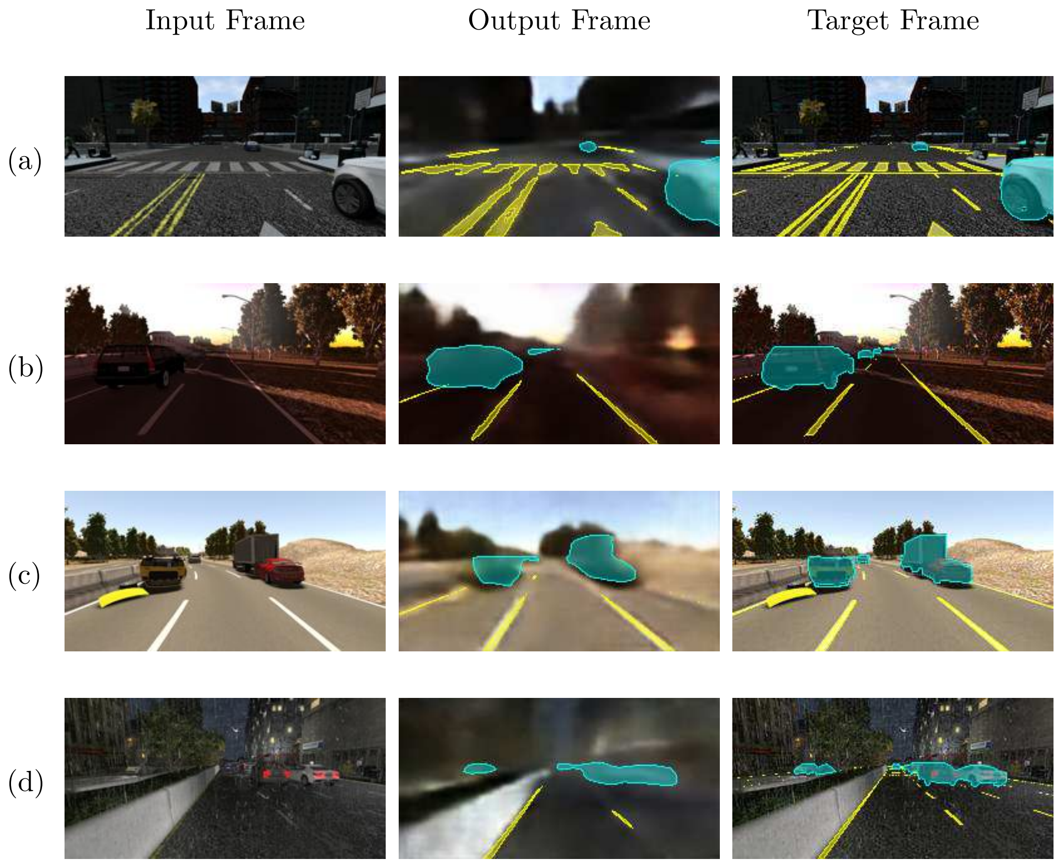


Figure 7.3: Results of *Net3* for four samples belonging to different categories of driving conditions: (a) city, (b) freeway, (c) sunshine, and (d) darkness. The cyan overlay indicates the `cars` entities, the yellow overlay the `lanes` entities.

Secondly, I illustrate a quantitative comparison between *Net2* and *Net3*. Table 7.3 reports the scores for the `cars` and `lanes` classes grouped into the four driving conditions mentioned before, together with the scores on the entire test set. In addition, I include in the comparison two other well-known models[1] for pure semantic segmentation, FCN-8 [138] and U-Net [194], both using VGG-16 [216] as base model. The scores demonstrates that *Net3* learns more consistent latent representations compared to *Net2* and the FCN-8 model, in all the categories of driving sequences. For both *Net2* and *Net3*, it is evident how the task of recognizing the `cars` concept achieves better scores compared to the `lanes` concept. An explanation of why the latter task is more difficult can be the very low ratio

[1] The Keras implementations I have used are available at the following repository: `https://github.com/divamgupta/image-segmentation-keras`

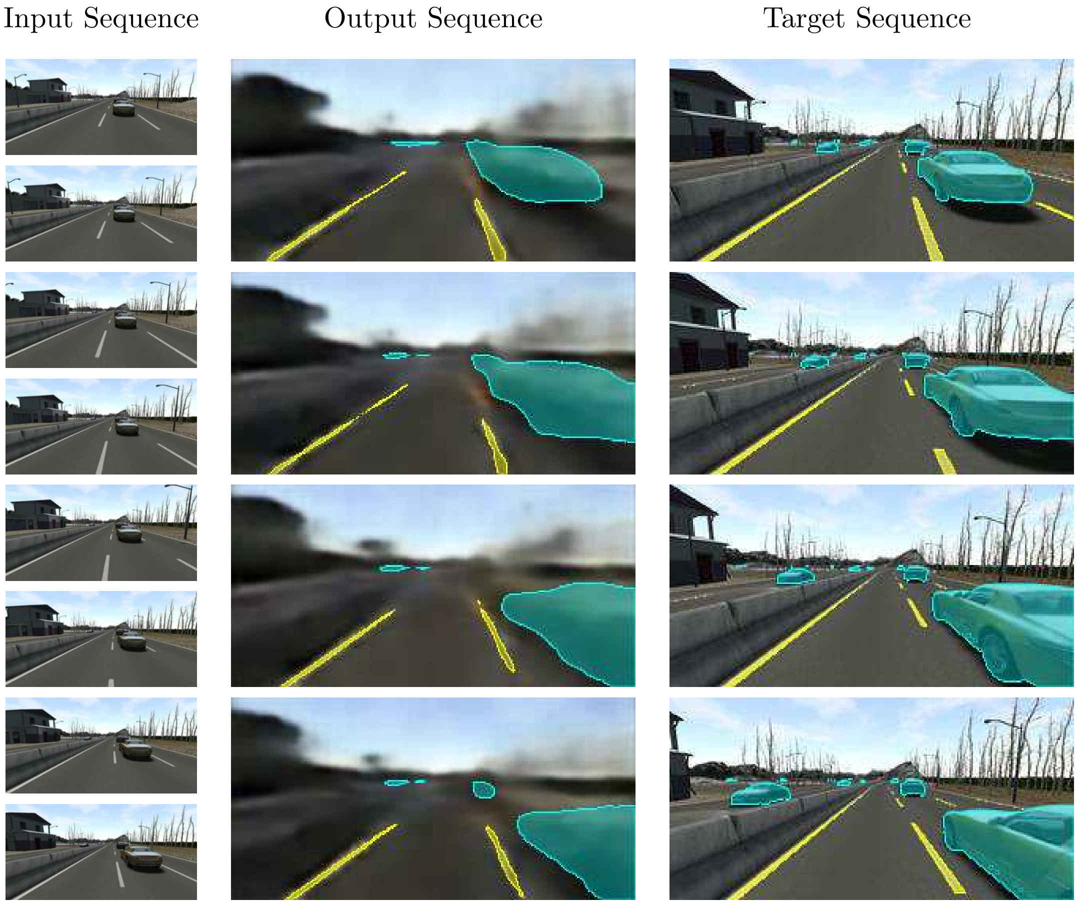


Figure 7.4: Result of *Net4* predicting 4 future frames from an input sequence of 8 frames, in a driving scenario on a freeway.

of pixels belonging to the class of `lanes` over the entire image size. Consequently, lane markings get easily occluded by other elements in the scene. However, the U-Net model outperforms all other models, although the scores are still comparable. I would like to stress again that the purpose of my networks is not mere segmentation of visual input, as discussed in Section §6.2.3. The segmentation operation must be considered a supporting task, forcing the models to learn a semantic organization of the representations. This internal organization is totally missing in the U-Net and FCN-8 models.

As regards *Net4*, Figs. 7.4 to 7.7 present four results of visual prediction for the different categories of driving scenarios. Each figure displays the 8-frames input sequence and the 4-frames predicted sequence in the future, together with the target sequence as a reference. Since the SYNTHIA sequences are acquired at 5 FPS, the network is predicting 0.8 seconds

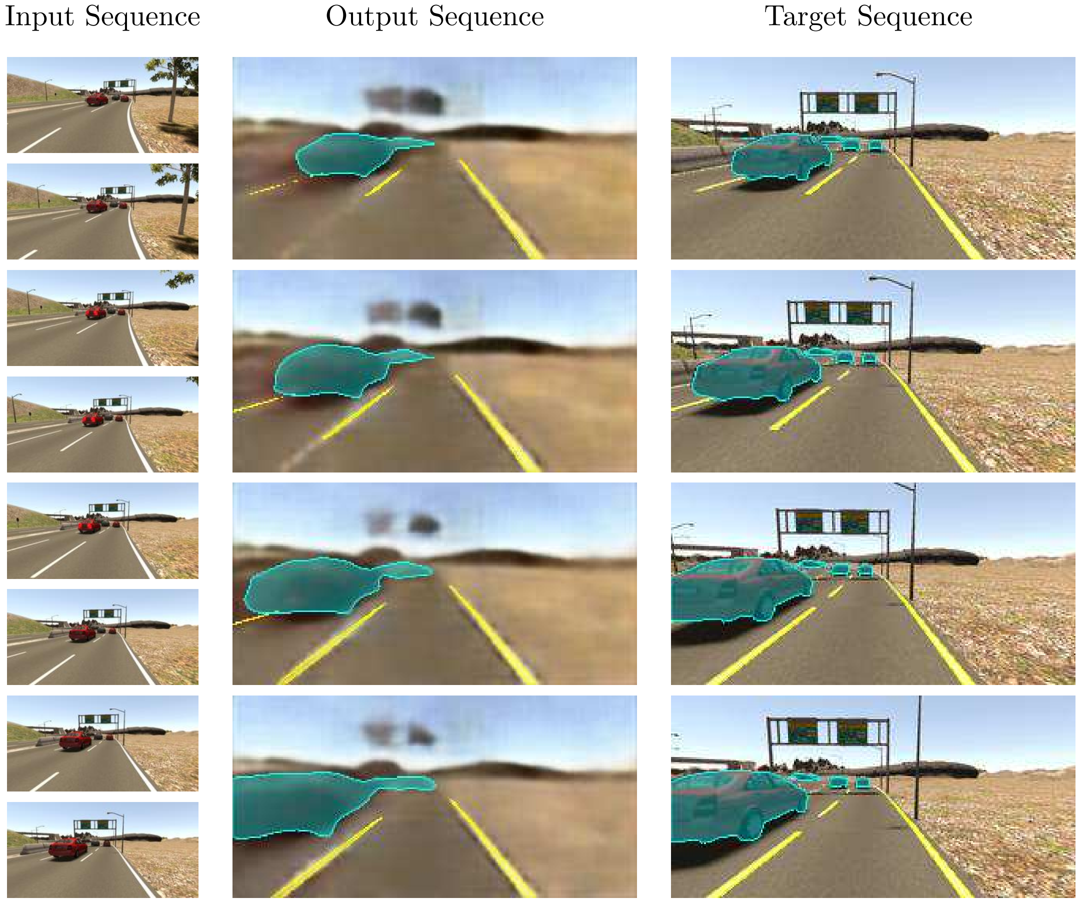


Figure 7.5: Result of *Net4* predicting 4 future frames from an input sequence of 8 frames, with sunny scenario.

in the future. The results are fairly accurate in all the scenarios, showing that the model can predict a variety of changes in the `cars` and `lanes` entities through time. In particular, the results in the "freeway" and "sunshine" scenarios (Figs. 7.4 and 7.5) demonstrate that the model can predict an overtake maneuver from the left as well as from the right. Another interesting result is the different predictions in presence of a crosswalk: in the "city" scenario (Fig. 7.6), a car is moving perpendicularly to the lane of the ego car, so the network correctly predicts to hold still at the crosswalk; in the "darkness" scenario (Fig. 7.7), cars are driving in the same direction of the ego car, so the model predicts not to stop at the crosswalk and moves forward.

Moving to quantitative results, Table 7.4 reports the performance of *Net4* for the different categories of driving sequences. It is immediate to note that the `cars` scores are

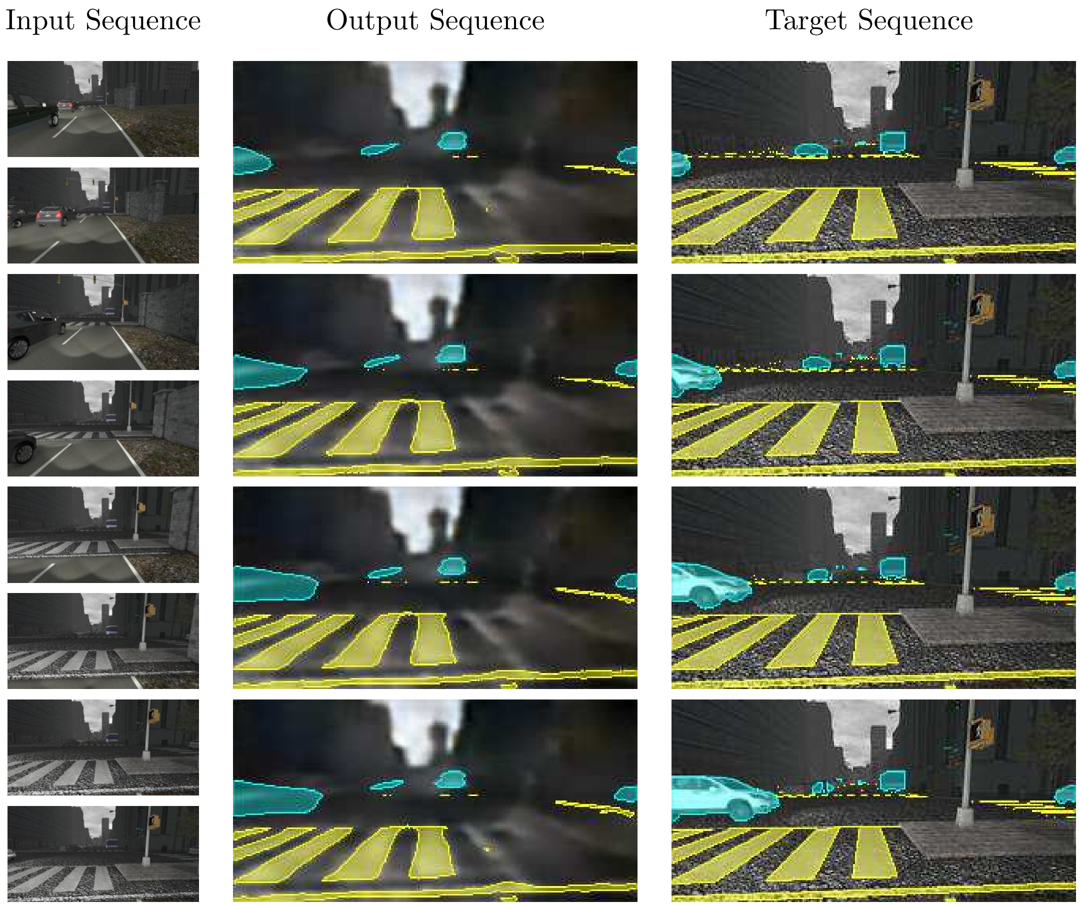


Figure 7.6: Result of *Net4* predicting 4 future frames from an input sequence of 8 frames, in a driving scenario of a city.

always higher than the `lanes` scores, just like in Table 7.3. However, the `cars` predictions worsen more significantly for the distant frames with a decay of 16%, while the `lanes` scores lose only 9%. This can be explained by the fact that, generally, in a driving sequence the lane markings change more smoothly and predictably compared to the cars, which can modify their trajectory all of a sudden.

Table 7.5 presents another quantitative comparison of different implementations of *Net4* based on the type of internal recursive node: basic RNNs [59], GRUs [34] and LSTMs [96]. The results indicate the GRUs are the best choice in my case. While it is not surprising that the basic RNNs obtain the lowest score, the fact that GRUs outperform LSTMs might seem unexpected. The reason could be twofold: first, the number of parameters in the model increases by more than 30% when switching from GRUs to LSTMs; second, in the context

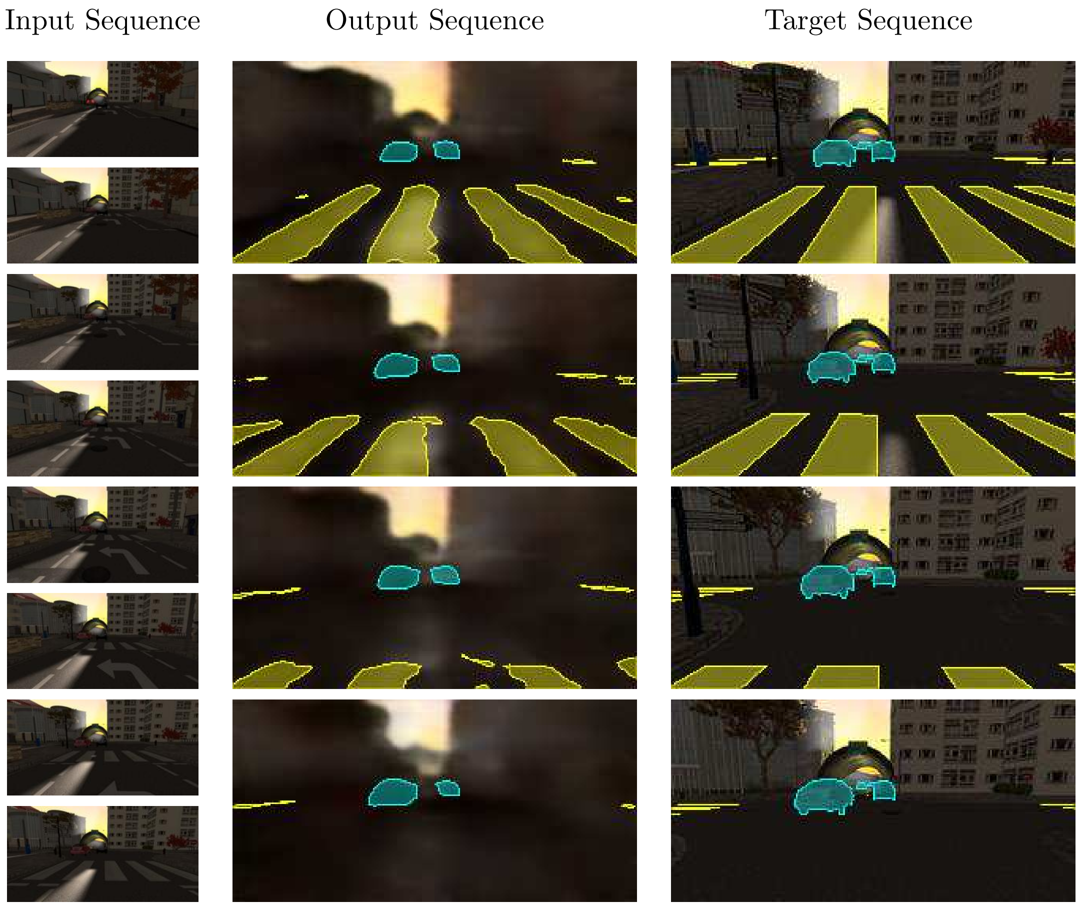


Figure 7.7: Result of *Net4* predicting 4 future frames from an input sequence of 8 frames, in a scenario with adverse illumination.

of driving, it is not so crucial to memorize scenarios occurred several seconds before. It is well known that LSTMs are the most powerful recursive node for long-term prediction, because of their ability to keep track of events in the remote past. However, while driving, the environment and the surrounding vehicles change so rapidly that it is often useless to try to draw a connection between the current scenario and, for example, the scenario seen 10 seconds before—note that the typical timescale of vehicle dynamics is less than one second. This situation is clearly far from Natural Language Processing, where LSTMs give their best.

	Frame 9		**Frame 10**		**Frame 11**		**Frame 12**	
	IoU car	IoU lane	IoU car	IoU lane	IoU car	IoU lane	IoU car	IoU lane
City	0.7543	0.5692	0.7173	0.5472	0.6799	0.5421	0.6381	0.5220
Freeway	0.6928	0.5197	0.6336	0.4698	0.5967	0.4487	0.5589	0.4296
Sunshine	0.7223	0.5338	0.6768	0.5001	0.6661	0.4831	0.6106	0.4693
Darkness	0.7000	0.5226	0.6570	0.5120	0.6130	0.5014	0.5834	0.4832
All	0.7078	0.5268	0.6639	0.5075	0.6315	0.4946	0.5931	0.4782

Table 7.4: Performance of *Net4* predicting a 4-frames sequence from a 8-frames input sequence. The scores are divided into `cars` and `lanes` classes, and they are organized into the four categories of driving conditions.

	Frame 9		**Frame 10**		**Frame 11**		**Frame 12**	
	IoU car	IoU lane	IoU car	IoU lane	IoU car	IoU lane	IoU car	IoU lane
RNN	0.6836	0.4884	0.5963	0.4231	0.5100	0.3957	0.4598	0.3668
GRU	0.7078	0.5268	0.6639	0.5075	0.6315	0.4946	0.5931	0.4782
LSTM	0.6810	0.5196	0.6604	0.4911	0.6426	0.4696	0.6119	0.4623

Table 7.5: Comparison of different recursive nodes in the implementation of *Net4*.

7.2.1 Analysis of the Latent Representations

In the following sections, I discuss in more detail the properties of the latent representations and show some interesting manipulations on the vectors. Let me start with a statistical evaluation of the latent representations learned by the three presented models of autoencoder. Table 7.6 reports the scores for two indicators measuring the temporal discrepancy and the predictivity error. The first indicator ξ measures the ratio between the difference of two latent vectors that are contiguous in time and the variance over the entire dataset $\mathcal{Z}$ of latent vectors. Therefore, this indicator evaluates how much subsequent vectors have consistent neural values—the lower the score, the better. The evaluation is performed independently for each neural unit of the latent vector and then averaged:

$$\xi_{\mathcal{Z}} = \frac{1}{N_{\mathrm{V}} M} \sum_{i}^{N_{\mathrm{V}}} \frac{\sum_{\mathbf{z}\in\mathcal{Z}} \left(z_i - z_i^{(1)}\right)^2}{\upsilon_i}, \tag{7.13}$$

where z_i is the i-th element of $\mathbf{z}$, $z_i^{(1)}$ is the i-th element of the successor of $\mathbf{z}$, υ_i is the i-th element of the variance vector of $\mathbf{z}$ over $\mathcal{Z}$, and M is the cardinality of $\mathcal{Z}$.

The second indicator ρ measures the "predictability" of the representations, i.e., how well one can predict from two consecutive vectors a third vector by linear regression. The

	Temporal discrepancy $\xi_{\mathcal{Z}}$	Predictivity error $\rho_{\mathcal{Z}'}$
Net1	0.299	0.186
Net2	0.297	0.189
Net3	0.180	0.077

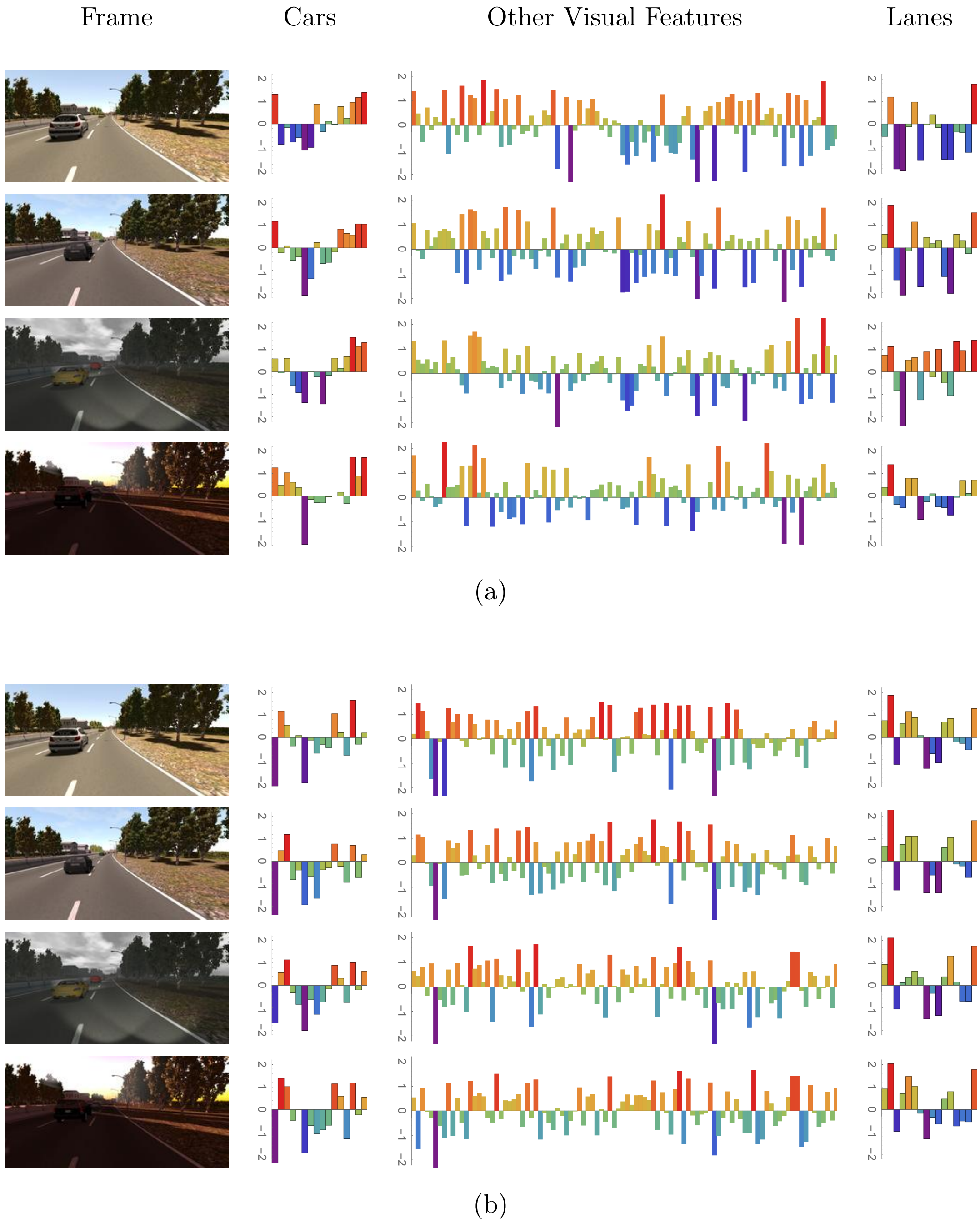
Table 7.6: Statistics on the latent representations learned by the presented perceptual models. For both indicators, the lower the better.

metric is the mean square of the residual obtained when using two consecutive latent vectors to predict one neuron of a third vector by linear regression. To have an acceptable computation time, this index is computed on a subspace $\mathcal{Z}'$ ten times smaller than $\mathcal{Z}$. By calling $\varepsilon(\mathbf{A}, \mathbf{b})$ the residual of the least squares approximation of the normal equation $\mathbf{Ax} = \mathbf{b}$, ρ can be written as follows:

$$\rho_{\mathcal{Z}'} = \frac{1}{N_\mathrm{V}} \sum_{i}^{N_\mathrm{V}} \varepsilon \left(\begin{bmatrix} \cdots & \cdots \\ \mathbf{z} & \mathbf{z}^{(1)} \\ \cdots & \cdots \end{bmatrix}_{\mathbf{z}\in\mathcal{Z}'} , \begin{bmatrix} \cdots \\ z_i^{(2)} \\ \cdots \end{bmatrix}_{\mathbf{z}\in\mathcal{Z}'} \right) . \tag{7.14}$$

Table 7.6 clearly shows how *Net1* and *Net2* have comparable scores, while *Net3* performs significantly better. In fact, only *Net3* introduces the temporal consistency inside the latent representations, and this is well reflected in the results.

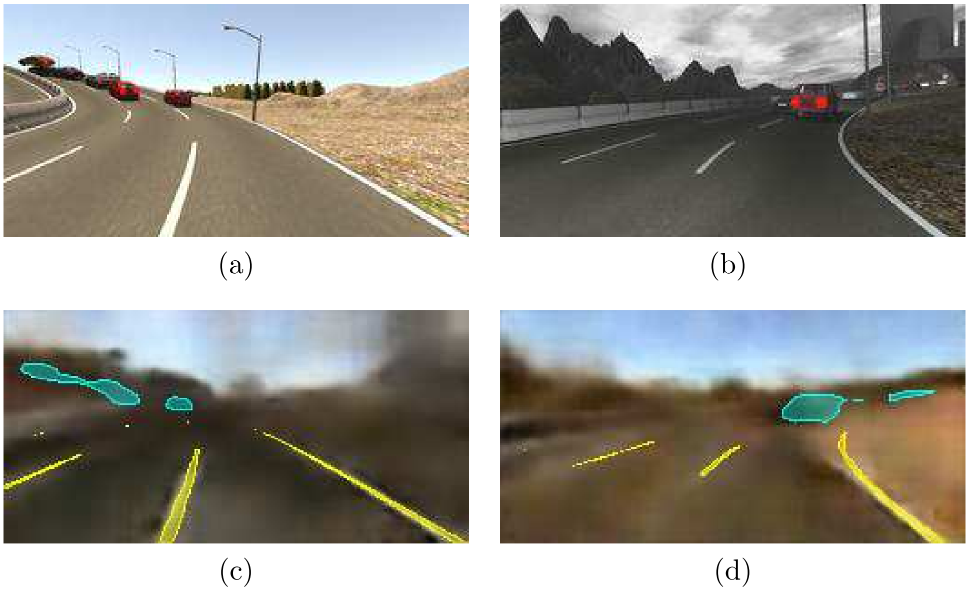
Fig. 7.8 presents a visual inspection of the latent representations learned by *Net2* and *Net3*. For each model, the left column of the figure shows four images of the same driving scenario under different lighting conditions. For each input image, I plot the values of the 128 neurons composing the latent representation computed by the model, separating the 16 neurons representing the `cars` entities (second column from the left), the 16 neurons representing the `lanes` entities (right column), and the remaining 96 neurons representing generic visual features (third column from the left). Ideally, only the generic 96 neurons should change among the four cases, because the input images differ only in the lighting conditions and have almost identical `cars` and `lanes` entities. Comparing the performance of *Net2* (a) and *Net3* (b), it is clear how the latter learns more consistent representations. In the case of (b), the variation in the neurons representing the `cars` and `lanes` concepts is minimal. The variation in the general 96 neurons is also very localized, and the neurons exhibit a similar overall distribution. This fits with the fact that the four images have the same surrounding (the trees, the soil on the right). Conversely, the representations learned by *Net2* do not appear as consistent; the `cars` and `lanes` neurons change visibly for each input, and even the other 96 visual features do not share any particular pattern in the four cases. Therefore, it is safe to conclude that forcing at once semantic organization and temporal coherence leads to more robust and disentangled representations.



(a)

(b)

Figure 7.8: Visualization of the latent representations learned by *Net2* (a) and *Net3* (b), for four images depicting the same scenario under different lighting conditions. Each row shows the values of the 128 neurons of the latent representation of the image on the left. The neurons corresponding to the `cars` and `lanes` concepts are plotted separately.



(a)

(b)

(c)

(d)

Figure 7.9: Result of swapping the conceptual segments between two latent spaces learned by *Net3*. Image (c) is the result of combining the `cars` and `lanes` neurons of (a) with the rest of the vector of (b). Image (d) is the opposite, combining the `cars` and `lanes` neurons of (b) with the rest of the vector of (a).

7.2.2 Manipulation of the Latent Representations

Here I present two interesting ways of manipulating the latent vectors to generate representations of novel scenarios. The first manipulation comes from exchanging segments between latent representations of different images. Fig. 7.9 shows the imaginary scenarios created by swapping the neurons of the `cars` and `lanes` concepts between two input images. Fig. 7.9(c) is decoded from a latent vector composed of $\mathbf{z}_{\mathrm{C}}$ and $\mathbf{z}_{\mathrm{L}}$ taken from the representation of (a), and $\widetilde{\mathbf{z}}$ coming from the representation of (b). Similarly, Fig. 7.9(d) is the result of combining $\mathbf{z}_{\mathrm{C}}$ and $\mathbf{z}_{\mathrm{L}}$ from the representation of (b) together with $\widetilde{\mathbf{z}}$ from the vector representing (a). This is a nice example of how the model can create artificial—yet plausible—scenarios.

The second type of manipulation is the interpolation between latent vectors. Fig. 7.10 shows an example of interpolation using the latent representations learned by *Net3*. Each column is the result of taking the latent vector of a first frame (first row in the figure) and linearly interpolate it with the latent vector of a second frame (last row). I generate 5 intermediate latent vectors, which are passed to the decoders of *Net3* to produce novel images. The images prove to be a smooth and gradual shift from the first input to the second, and they successfully provide new plausible driving scenarios not seen before by the network.

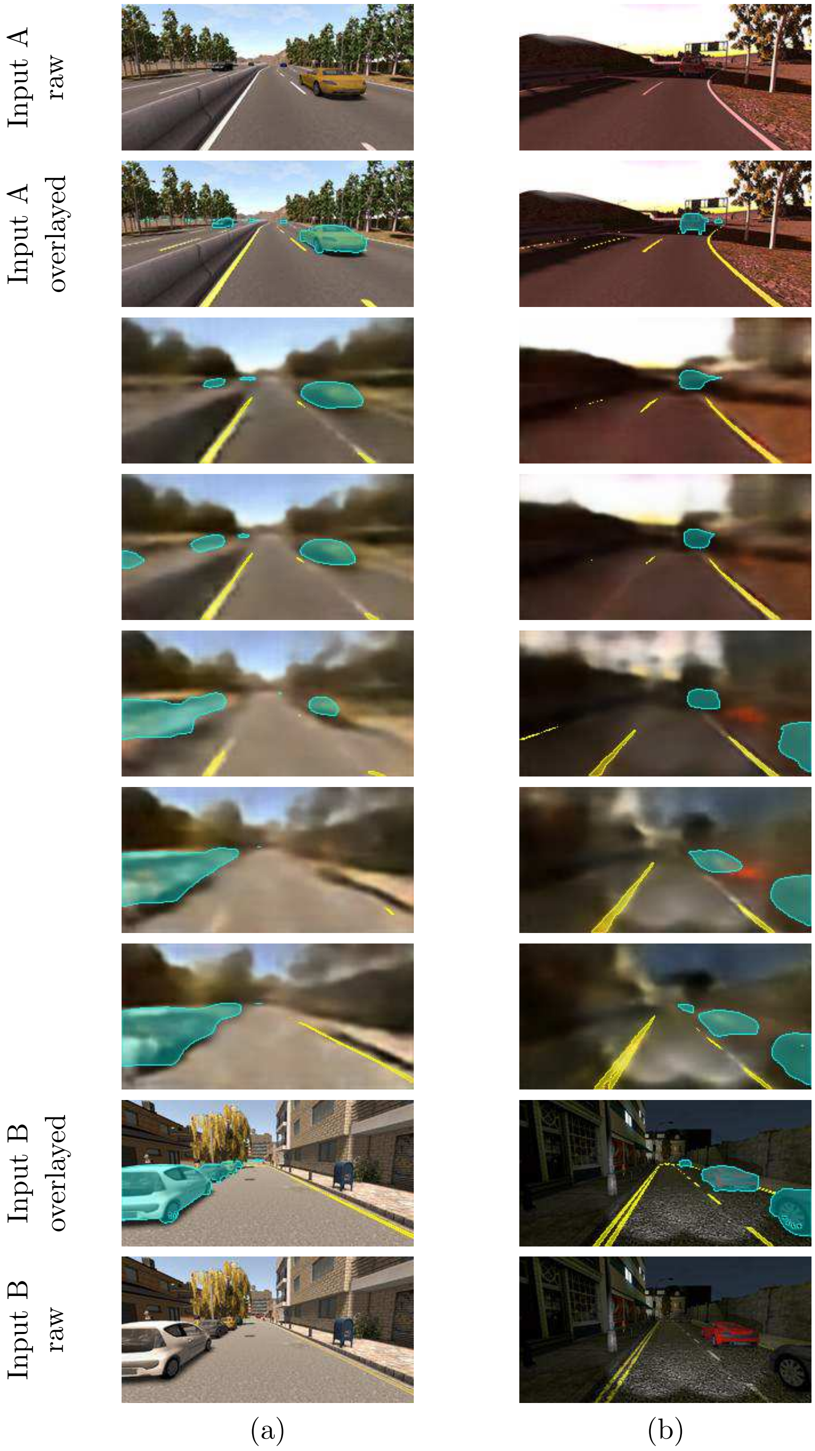
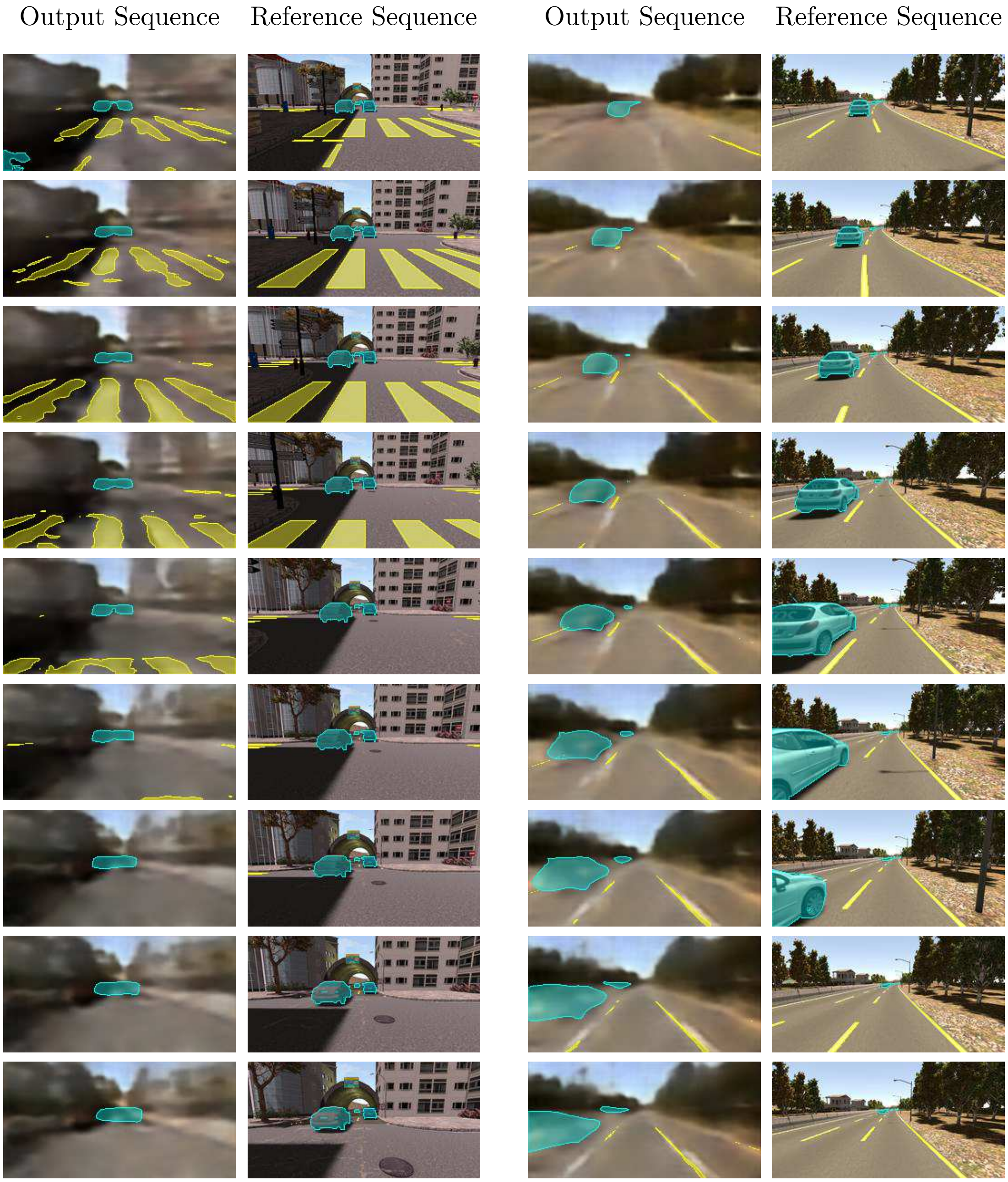


Figure 7.10: Two examples of interpolation between latent representations learned by *Net3*, for sunny scenarios (a) and scenarios with adverse illumination (b). The first two rows display the first input frame, with and without the colored overlay showing the `cars` and `lanes` entities. Similarly, the last 2 rows show the second input frame. The 5 central rows are the result of the linear interpolation between the latent representations of the two inputs.



(a) (b)

Figure 7.11: Two examples of *Net4* emulating mental imagery, in a urban driving scenario (a) and a freeway (b). Odd columns show the output of the model, while even columns are a reference of the corresponding frames in the temporal sequence.

7.2.3 Emulating Imagery

As final result of the dynamic models, I present the outcome of emulating the phenomenon of mental imagery using *Net4*—I have introduced the concept of mental imagery in Section §2.1.2. To mimic this process, the network is called iteratively and each iteration receives as input the output of the previous iteration. In the context of recurrent neural networks, this technique is sometimes called *hallucination*. In the specific case, at each iteration I take the first of the 4 output vectors and use it as the eighth input vector of the next iteration.

Fig. 7.11 illustrates the results of 9 iterations of imagery for two different scenarios, along with the corresponding reference frames (the input images are omitted for practical reasons). Note that, although the imagery process must inevitably start with all input frames taken from the original dataset, the results provided in the figure comes from forward iterations when all input vectors are result of previous iterations. In both driving scenarios, it is possible to appreciate how the model can predict a quite plausible future from just its own representations of the world.

Chapter 8

High-level Representation Spaces

All the models presented in the previous chapters share the idea of a solution close to the way the human brain solves the problem of visual perception. Along these lines, the preferred reference system for representations is the visual space, organized along two dimensions just like most cortical maps in the visual system. This is the space where the fundamental simulative process of imagination takes place.

Most of the existing systems for automated driving do not use a reference system resembling the visual space or any mental space of representation. Usually, they have a modular structure, as described in Section §4.1, and each module uses its own specific representation space. The modules at higher levels generally adopt representations focused on mathematical simplicity and practical implementation, rather than biological plausibility.

Since some of these representation spaces are now well established, it is important to verify the possibility of extending my research towards higher-level spaces of representation. This chapter describes my early explorations in this direction. This work is the result of a collaboration with the Intelligent Vehicles group from TU Delft, where I had the pleasure of being a visiting researcher under the supervision of Dr. Julian F.P. Kooij.

The first section summarizes the existing high-level representation spaces relevant in the context of autonomous driving. Moreover, it presents my attempt to formulate a variant of representation space that could conciliate between mathematical efficiency and biological plausibility. The second section describes the architecture of the model and the different choices of input and output formats I have experimented with. The third section presents the preliminary results obtained so far, with a description of the custom metrics adopted to evaluate the model performance. The last section illustrates a further possible direction of this line of research.

8.1 Abstract Representations of Driving Scenarios

A straightforward approach to represent a driving scenario is to use a vector of physical quantities for each relevant object in the scene. For example, the vector $\mathbf{o}_k$ of an object k can be composed of the spatial coordinates of the object center, the heading ψ, and the speed v:

$$\mathbf{o}_k = \left[X^{(k)}, Y^{(k)}, Z^{(k)}, \psi_k, v_k\right]^T. \tag{8.1}$$

If the object is a vehicle, additional components can be the length and width of the vehicle. Then, the overall scenario is characterized by the following set:

$$O = \{\mathbf{o}_k\}_{k \in [0 \dots N]}, \tag{8.2}$$

where the ego car is usually indicated with $k = 0$. This example of abstract representation relies on basic mathematical entities, like vectors of continuous real values and sets. Therefore, it is easy to employ this representation in several conventional algorithms for vehicle control. In fact, this approach has been widely adopted in the field of autonomous vehicles [7, 234, 171], although sometimes the values in equation (8.1) are discretized to save representation space [166, 188].

This kind of representation finds no correspondence with any brain mechanism. First of all, the variable O in equation (8.2) is identical for all the possible permutations of its elements, while in a brain process every composite representation is linked to a specific physical layout of components. More importantly, it is particularly difficult to implement the case where O contains an arbitrary number of components: this issue implies a severe incompatibility with cognitive processes. The problem derives from an algorithmic drawback of artificial neural networks, as that managing representations with a variable number of dimensions is almost intractable.

8.1.1 Occupancy Grids

The need for a representation dealing with a variable number of objects has lead to the research for an alternative feasible strategy. A valid solution has come from the field of robotic perception and navigation, under the name of *occupancy grids* [58]. An occupancy grid G is a 2D lattice of binary elements $g_{i,j}$, where the element with index i, j is related to a patch $A_{i,j}$ of the continuous world space defined as follows. Let the Z-axis be parallel to the heading of the ego car pointing in the travel direction, the X-axis be on the ground plane perpendicular to Z pointing towards the right of the ego vehicle, and the Y-axis be perpendicular to the ground plane pointing upwards. The patch $A_{i,j}$ is the area on the ground plane defined by a neighborhood of the point $\langle X_i, Z_j \rangle$ in the world space:

$$A_{i,j} = \left\{ [X, Z]^T \,\middle|\, X \in \left[X_i - \frac{\Delta X}{2}, X_i + \frac{\Delta X}{2}\right], Z \in \left[Z_j - \frac{\Delta Z}{2}, Z_j + \frac{\Delta Z}{2}\right] \right\}, \tag{8.3}$$

where the space tessellation is uniform:

$$\Delta X = \quad X_{i+1} - X_i \qquad \forall i, \tag{8.4}$$

$$\Delta Z = \quad Z_{j+1} - Z_j \qquad \forall j. \tag{8.5}$$

The occupancy grid can be also represented as a binary image of width W and height H. In this case, the origin $\langle 0, 0\rangle$ of the system of coordinates in the image space be in the top-left corner, while the bottom-right corner of the image has coordinates $\langle W-1, H-1\rangle$. Assuming that the occupancy grid G is representing a space occupied by a set O of objects, the pixels of the binary image are defined as follows:

$$g_{i,j} = \begin{cases} 1 & \text{if } \exists\, \mathbf{o}_k \in O \mid [X^{(k)}, Z^{(k)}] \in A_{i,j} \\ 0 & \text{otherwise} \end{cases}, \tag{8.6}$$

where the pixel coordinates are computed as follows:

$$i \;=\; \frac{W}{2} + \frac{X_i}{\Delta X}, \tag{8.7}$$

$$j \;=\; H - \frac{Z_j - \widetilde{Z}}{\Delta Z}. \tag{8.8}$$

Therefore, the pixel $g_{i,j}$ indicates if the corresponding patch $A_{i,j}$ of world space contains an object. The parameter $\widetilde{Z}$ represents the length of the “blind zone” immediately in front of the ego vehicle which is not visible from the point of view of the camera. Note that, since the set O of vectors identifying the objects is the result of measurement affected by uncertainty, the values of $g_{i,j}$ are often probabilities in $[0 \dots 1]$ (graylevels) rather than simple binary values.

Probabilistic occupancy grids have become the standard representation in robotics. There are now many established algorithms for *occupancy grid mapping*, i.e., the generation of consistent maps from noisy and uncertain measurement data. Grid mapping is achieved by sequencing the robot into many possible poses, acquiring measures, and incrementally adapting the probability values $g_{i,j}$ [225]. Occupancy grids have become a popular format of representation also for high-level modules of autonomous vehicle systems, even before the appearance of deep learning [232, 47]. They are well suited for artificial neural networks as well, because the format of a 2D matrix combines effectively with convolutional neural networks [109, 139, 142]. Moreover, values in a grid cell can span multiple channels, including additional information such as the semantic of the object or its velocity [97, 109, 61]. Currently, occupancy grids are frequently adopted in deep learning solutions for motor control [210, 104], and a recent overview of their applications can be found in [153].

8.1.2 Biologically Plausible Occupancy Grids

The principle of imitating human cognition and brain organization needs always to be balanced with the advantages of structures and processes suitable for computers. In the case of

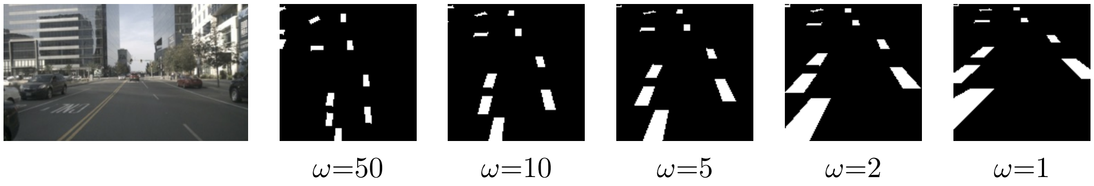


Figure 8.1: Same traffic scenario (leftmost image) represented by warped occupancy grids rendered with different values of ω.

high-level representations of driving scenarios, achieving this balance turns out to be more tricky, mainly for two reasons. First, there are fewer neuroscientific explanations in higher-level visual representations than in early visual processing stages. Second, occupancy grids offer undeniable computational advantages because of their matricial format.

In this section, I investigate how to improve occupancy grids towards a biological plausibility and, at the same time, how to preserve their computational advantages. I focus on modifying two key aspects that go against biological plausibility: the point of view and the uniform tessellation of the world space. Firstly, an occupancy grid—as conventionally used in the context of driving—corresponds to a top-down view of the world space, the so-called *bird's-eye view* (BEV). This orthographic view is not only impossible for a human driver but also challenging to obtain with on-board sensors. Without the use of a LIDAR, obtaining a BEV requires an *inverse perspective mapping* [140] of the front camera, a transformation that is still prone to errors and distortions, even though it has been greatly improved with the use of deep learning [24]. Secondly, at every level of the biological visual system, the retinal space is never represented uniformly in the cortical space. In the primary cortical visual area, the space warping with respect to the eye view is known as *cortical magnification* [101, 57, 19]. This warping has the effect of enlarging, in the visual cortex, the central space of the scene with respect to the peripheral areas, by a factor up to 20.

I propose to apply a similar warping to the occupancy grid in order to magnify close objects and to reduce the size of distant objects. Just like with cortical magnification, in the "warped occupancy grid" the more an object is relevant, the more it is represented in detail. In the case of driving, the relevance of an object depends on the time required by the ego car to reach it: the closer objects are the most crucial and require more precision. The cortical magnification is typically formulated with the polar-log transformation. In my approach, I define the warping transformation as a logarithmic transformation in the longitudinal dimension (Z-axis), and as a linear transformation in the lateral dimension (X-axis). In this way, every element of the warped occupancy grid corresponds to a square patch of the world space. The warping transformation is the following:

$$w(Z) = \log(Z + \omega) - \log(\omega), \tag{8.9}$$

where ω is the constant defining the amount of warping, with the maximum at $\omega = 1.0$

and no effect at all for $\omega \to \infty$. Fig. 8.1 gives an example of how ω influences the appearance of a warped occupancy grid. While the equations (8.7) and (8.8) show the forward transformation of a world point into image coordinates for the case of the linear occupancy grid, the forward transformation for the warped occupancy grid is the following:

$$
\begin{aligned}
i &= \frac{W}{2} + \frac{H\Delta Z}{w(H\Delta Z)} \frac{w(Z_j)}{Z_j} \frac{X_i}{\Delta X}, && (8.10)\\
j &= H - H\frac{w(Z_j) - w(\widetilde{Z})}{w(H\Delta Z)}. && (8.11)
\end{aligned}
$$

In the linear occupancy grid, the size of a vehicle is constant in every point of the image space, whereas in the warped occupancy grid closer vehicles have a larger size. Hence, errors in the model prediction have less impact for the vehicles in the proximity of the ego car.

Weighted Occupancy Grids

Besides the linear and warped occupancy grids, I introduce a third variant of occupancy grid I have called "weighted occupancy grid". I exploit this type of occupancy grid to better evaluate the error distribution in relation to the distance of the objects from the ego point of view, and to perform a comparison with the other formulations of occupancy grid, as I will show in the results of Section §8.3.2.

The weighted occupancy grid is equal to the linear occupancy grid, except that during the computation of the loss I apply a matrix of weights favoring the area closer to the ego car. The matrix ensures that pixels corresponding to the same real-world distance from the origin have the same weights. An element $m(i, j)$ of the matrix is computed as follows:

$$
\begin{aligned}
d(i,j) &= \left\| \left[\Delta X i - \frac{W}{2}, \widetilde{Z} + \Delta Z(H - j) \right] \right\|, && (8.12)\\
m(i,j) &= e^{-\dfrac{d(i,j) - d\left(\frac{W}{2}, H\right)}{d(0,0)}}. && (8.13)
\end{aligned}
$$

In addition, the matrix is normalized by the mean of the weights, so that the sum of all the weights is equal to 1: in this way, the order of magnitude of the loss is not affected by the application of this matrix.

8.2 Models

In this section, I present the early development towards a model that uses a higher level representation space in output. More specifically, I compare some variations of the model using three different output spaces: the standard form of occupancy grid described in §8.1.1, the warped variant I have defined in §8.1.2, and the weighted occupancy grid useful

Encoder	convolution	$5 \times 5 \times 64$
	convolution	$5 \times 5 \times 64$
	convolution	$3 \times 3 \times 128$
	convolution	$3 \times 3 \times 128$
	dense	512
Latent space		256
Decoder	dense	2048
	deconvolution	$3 \times 3 \times 64$
	deconvolution	$3 \times 3 \times 32$
	deconvolution	$5 \times 5 \times 1$
Total parameters		3 million

Table 8.1: Parameters describing the architecture of the preliminary implementation of the model.

for a more complete comparison. In addition, I consider two different possibilities for the input format. The most straightforward choice of input is a frame from the ego camera stream. The second type of input I consider is a preliminary form of visual attention applied to the image frame, which I will explain in detail in Section §8.2.2.

The general model is based on an standard convolutional encoder-decoder scheme where the input is an RGB image of 800×450 pixels representing either the original frame or an "attention map" emphasizing the vehicles; the output is a graylevel image of 128×128 pixels corresponding to one of the three formats of occupancy grid. The provisional architecture of this model is summarized in Table 8.1. The loss function used to train the model is the binary cross-entropy function. Before describing the input format of "attention maps", I illustrate in the next section the dataset adopted in the development of this model.

8.2.1 NuScenes Dataset

The model presented in this chapter defines the output in a format that can be directly exploited in real driving contexts. Hence, the model requires a dataset of real-world recordings of driving scenarios. This is not the case in the models illustrated in Chapters 6 and 7, which both predict in the camera space. I have mentioned in Section §6.1.2 that SYNTHIA is an artificially generated dataset, and it has annotation only in the image space. The current model, therefore, is incompatible with this dataset. The dataset in this case has to be composed of recordings of real scenarios, and it must provide annotations of 3D bounding boxes of the surrounding vehicles. After reviewing the main options available, as in Section §4.2, I have opted for the nuScenes dataset [26].

The nuScene dataset is organized into 850 annotated video sequences, for a total of more than 100,000 frames. Fig. 8.2 depicts some examples of frames taken from the dataset,

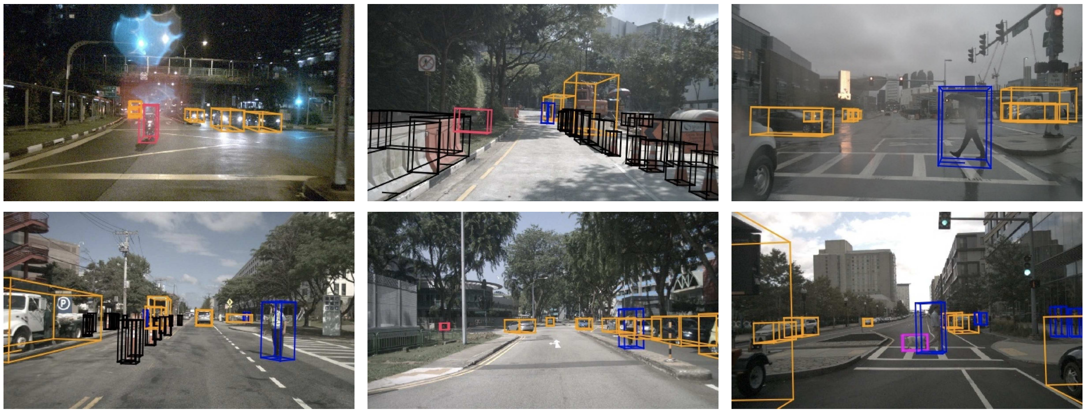


Figure 8.2: Samples from the nuScenes dataset, showing the variety of environmental conditions and annotations of 3D bounding boxes.

showing the significant variety of environments, illumination, and weather conditions. Still, at this early stage of development, the wide variety of scenarios might be a hindrance to the learning of the model. In fact, there are numerous atypical video sequences where, for example, the ego car is completely stationary or parked on the side of the road. In other video sequences, the camera is so covered by raindrops that the scene is barely visible. The disparate conditions provided by the many video sequences are undoubtedly a precious asset useful for the evaluation of a mature model. However, considering that the model is still in a prototyping phase, at this time I have chosen to remove the more disorienting video sequences from the dataset, namely the videos recorded at nighttime, in the rain, and where the ego car is stationary. Moreover, a smaller dataset allows the training and testing process to be swifter, and this is ideal at this early stage of development.

Hence, I have reduced the dataset to 100 video sequences. I have randomly allocated 70% of these sequences to the training set, 25% to the validation, and 5% to the test set, ensuring no overlap among the three sets. In addition, to generate the ground truth data, I have mapped the 3D bounding box annotations into binary images of occupancy grids with size 128×128 pixels, using the forward transformations defined in Sections §8.1.1 and §8.1.2. I have set $\omega = 2$, $\widetilde{Z} = 3.5$ m, and $\Delta X = \Delta Z = 0.5$ m so that each pixel of the occupancy grid corresponds to a square of 0.25 m^2 in the world space.

8.2.2 Attention Mechanism

Here I present the second format of input I have experimented with in the development of the current model. There are unavoidable discrepancies between human perception and artificial perception, especially in the context of driving. Normally in artificial perception, the camera of a car acquires images covering the whole driving scene populated by numerous

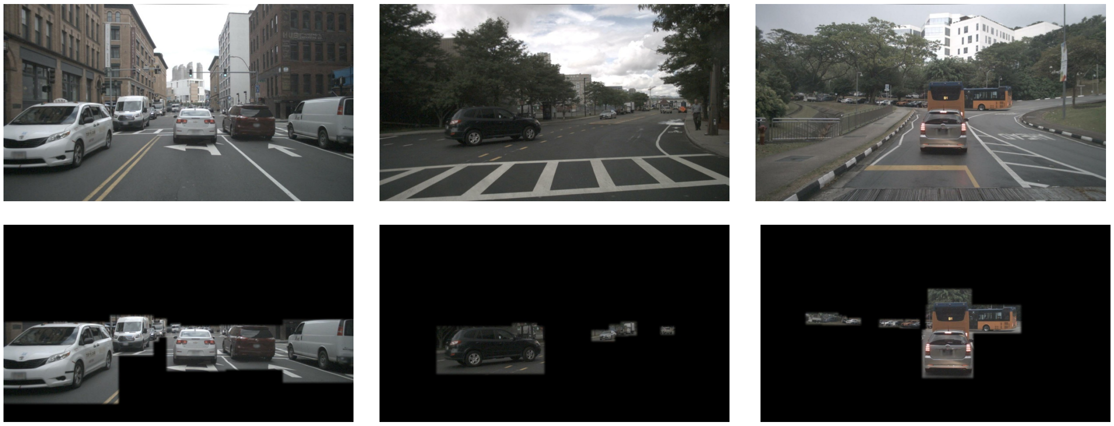


Figure 8.3: Examples of the attention mechanism applied in three different scenarios: the first row shows the original images, and the second row displays the results of "paying attention" to the vehicles.

objects, which may be relevant or negligible. Then, complex processes analyze the entire images to attempt to recognize the relevant objects and locate them in the world geometry.

In human perception, there exists a set of cognitive operations that deals with cluttered visual scenes by selecting important information and by filtering out irrelevant information. For example, when driving through the countryside, I direct my attention towards the road and the cyclist in front of me, and not towards the colorful fruits on the trees by the roadside. These mechanisms take the name of *visual attention*, one of the most studied topics in visual science [51, 189, 115, 156]. Visual attention also interacts with the process of perceptual learning described in Section §2.3; the attention mechanism reinforces the learning rule related with the objects that are most often salient in the scene [192]. During driving, the combination of attention and perceptual learning enhances the accurate and fast perception of the surrounding vehicles.

I propose a strategy that attempts to imitate the role of visual attention to reduce the computational complexity of my artificial neural model. When dealing with real traffic scenes like those of the nuScenes dataset, it is evident how the salient parts of the scenes are mainly the vehicles—see the examples in the top row of Fig. 8.3. The attention should be directed mostly to the vehicles, rather than to the surrounding buildings or foliage. The task of salience detection can be easily solved with current deep learning models; there is no need to invent anything new for this purpose, as many effective and simple neural networks are available to perform accurate prediction of salient locations [126, 52, 25, 110]. In my case, I simulate the effect of attention simply by detecting the 2D bounding boxes of surrounding vehicles with the well-established YOLO-v3 model [187], and by masking all the remaining non-significant areas. The bottom row of Fig. 8.3 shows the result of this process. Note that there are many computational models [49, 106, 50, 123] that implement aspects of natural visual attention more accurately. However, the purpose of these models is to help investigate the neurocomputational basis of visual attention, and they are not aimed at engineering applications requiring efficiency and high performance.

8.3 Results

This section presents the results gathered from the preliminary development of the presented model. To evaluate how the choice of format in the input and output representations affects the performance of the model, I have adopted a set of specific metrics, illustrated in the following section.

8.3.1 Evaluation Metrics

The three evaluation metrics I have considered are intersection over union (IoU), average precision (AP), and distance between centroids. While the IoU is computed between the overall images of target and predicted occupancy grid, the AP and the centroids are computed on the matches between connected regions extracted from the occupancy grid images. In addition, to better assess how the warped occupancy grid format improves the prediction of closer vehicles, I also apply the three metrics separately in three different classes of depth: “close”, “middle”, and “far” ranges.

Connected Regions and Classes of Depth

Each sample $x \in \mathcal{X}$ in the dataset $\mathcal{X}$ is processed in terms of connected regions, defined as follows. Let $R(x) = \{r_1, r_2, \cdots, r_{N_R}\}$ be the set of N_R connected regions of the target occupancy grid image associated with x. Similarly, let $S(x, \theta) = \{s_1, s_2, \cdots, s_{N_S}\}$ be the set of N_S connected regions in the predicted occupancy grid image computed by the model on sample x, after applying a binarization threshold θ. For sake of simplicity, from now on I will imply the dependency on x and write simply $S(\theta)$ and R.

The matches between target and predicted connected regions are computed using a greedy algorithm evaluating the IoU score, as in [78]—a possible future alternative is to use the Hungarian algorithm [127]. With $S^*(\theta)$ representing the set of predicted connected regions that successfully match with some target connected regions, the set of successful matches is $M(\theta) = \{\langle i, j \rangle : r_i \in R,\ s_j \in S^*(\theta)\}$.

I partition the sets of connected regions into three ranges of depth—“close”, “middle”, and “far”—defined by the limits Δ_1 and Δ_2 in meters. With $h(\cdot)$ representing the depth (in real-world coordinates) of the centroid of a connected region, the set of matched regions

$S^*(\theta)$ can be divided as follows:

$$\begin{aligned} S^*_C(\theta) &= \left\{s_j \in S^*(\theta) : \langle i,j \rangle \in M(\theta) \wedge h(r_i) < \Delta_1 \right\}, \\ S^*_M(\theta) &= \left\{s_j \in S^*(\theta) : \langle i,j \rangle \in M(\theta) \wedge \Delta_1 \leq h(r_i) \leq \Delta_2 \right\}, \\ S^*_F(\theta) &= \left\{s_j \in S^*(\theta) : \langle i,j \rangle \in M(\theta) \wedge h(r_i) > \Delta_2 \right\}. \end{aligned}$$

In addition, I partition all regions of $S(\theta)$—matched and non-matched—into the three depth classes:

$$\begin{aligned} S_C(\theta) &= S^*_C(\theta) \cup \left\{s_i \in S(\theta) : s_i \notin S^*(\theta) \wedge h(s_i) < \Delta_1 \right\}, \\ S_M(\theta) &= S^*_M(\theta) \cup \left\{s_i \in S(\theta) : s_i \notin S^*(\theta) \wedge \Delta_1 \leq h(s_i) \leq \Delta_2 \right\}, \\ S_F(\theta) &= S^*_M(\theta) \cup \left\{s_i \in S(\theta) : s_i \notin S^*(\theta) \wedge h(s_i) > \Delta_2 \right\}. \end{aligned}$$

It holds that $S^*_C(\theta) \cup S^*_M(\theta) \cup S^*_F(\theta) = S^*(\theta)$ and $S^*_C(\theta) \cap S^*_M(\theta) \cap S^*_F(\theta) = \emptyset$.

Average Precision

The precision score $p(\theta)$ of a sample is defined as follows:

$$p(\theta) = \begin{cases} 0 & \text{if } |S(\theta)| = 0, \\ \dfrac{|S^*(\theta)|}{|S(\theta)|} & \text{otherwise.} \end{cases} \tag{8.14}$$

The average precision $\bar{p}$ summarizes the shape of the precision/recall curve, and it is defined as the mean precision at a set of equally-spaced recall levels $\Theta = [0, \frac{1}{N_\Theta}, \ldots, 1 - \frac{1}{N_\Theta}, 1]$:

$$\bar{p} = \frac{1}{N_\Theta} \sum_{\theta \in \Theta} \max_{\tilde{\theta} \in \Theta \,:\, \tilde{\theta} \geq \theta} p(\tilde{\theta}). \tag{8.15}$$

It is straightforward to specialize the equations (8.14) and (8.15) for each of the three classes of depth, obtaining $\bar{p}_C$, $\bar{p}_M$, and $\bar{p}_F$.

Distance between Centroids

This metric evaluates the distances between the centroids of predicted and target connected regions. For two regions r_i and s_j, the centroid distance is the following:

$$q\left(r_i, s_j\right) = \left\| c(r_i) - c(s_j) \right\|. \tag{8.16}$$

A new set of matching $D(\theta)$ is computed similarly to $M(\theta)$, but using the centroid distance instead of the IoU score to evaluate the goodness of a match. Then, the final score $\bar{q}$ is the mean distance between the centroids of matching regions. In addition, as in the case of the average precision, it is possible to compute the score for each class of depth, obtaining $\bar{q}_C$, $\bar{q}_M$, and $\bar{q}_F$.

Intersection over Union

The third evaluation metric is simply the IoU between the predicted and the target occupancy grid images, using a fixed binarization threshold θ_I. As for the previous metrics, I consider three additional IoU scores computed on different portions of the image according to the three depth ranges.

8.3.2 Preliminary Results

This section illustrates the results obtained by the presented model in the current, yet initial, phase of development. The model has been trained for 200 epochs on the reduced version of the nuScenes dataset described in §8.2.1. I have evaluated the model on all the combinations of input and output formats presented in this chapter. The visual input types are the original frames and the attention maps. The high-level output representations are the linear occupancy grids, the weighted occupancy grids, and the warped occupancy grids. The results are evaluated with the metrics IoU, average precision, and distances between centroids; each metric is further applied separately to the three ranges of depth defined in §8.3.1. In the final evaluation of the model, I have set the binarization threshold $\theta_I = 0.5$ and the number of recall levels $N_\Theta = 40$.

Table 8.2 shows the results grouped by evaluation metric, and each group includes the six combinations of input/output and the four classes of depth range. The numbers marked in bold represent the best scores achieved for that metric in one of the classes of depth. An overall view of the results seems to suggest that all the model variations perform very similarly. There is no combination that obtains significantly better scores in any of the metrics considered. However, it is possible to identify some patterns that seem to occur consistently. Firstly, the model variations using the original frames as input (`FRM`) never achieve the best score in any of the metrics. It can be deduced that the input format of attention maps (`ATT`) offers a significant advantage in the training of the neural network. Therefore, the simple mechanism proposed here to imitate human visual attention proves to be another effective application of a neurocognitive principle into a computational algorithm. Secondly, the model variations that perform better in the close range (`CLO`) are the models having as output format the warped occupancy grids (`WRP`). This result is consistent with the idea that the grid warping should mimic the effect of cortical magnification, as described in Section §8.1.2. Hence, the model predicts with more accuracy the vehicles in the proximity of the ego car. Conversely, in the far range of depth (`FAR`), the warped occupancy grids obtain the lowest scores, whereas the format of linear occupancy grids (`OCC`) has the best performance in all the metrics. This is, again, coherent with the fact that a standard occupancy grid represents all vehicles in the scene with the same size in the grid space: even a car close to the camera (which occupies a large portion of the image frame) or a very distant car (displayed in few pixels of the frame) will occupy a similar number of grid elements.

IoU ↑

	ALL	CLO	MID	FAR
ATT → OCC	0.619	0.734	0.588	**0.535**
ATT → WOC	**0.623**	0.749	**0.601**	0.518
ATT → WRP	0.559	**0.774**	0.552	0.351
FRM → OCC	0.602	0.708	0.573	0.526
FRM → WOC	0.616	0.731	0.595	0.522
FRM → WRP	0.561	0.756	0.550	0.378

(a)

Average Precision ↑

	ALL	CLO	MID	FAR
ATT → OCC	0.472	0.557	0.396	**0.462**
ATT → WOC	**0.482**	0.593	**0.411**	0.442
ATT → WRP	0.467	**0.714**	0.365	0.323
FRM → OCC	0.448	0.517	0.369	0.458
FRM → WOC	0.459	0.531	0.393	0.453
FRM → WRP	0.449	0.638	0.355	0.354

(b)

Centroids ↓

	ALL	CLO	MID	FAR
ATT → OCC	0.940	0.823	1.109	**0.863**
ATT → WOC	0.944	0.772	1.186	0.873
ATT → WRP	**0.931**	**0.540**	**1.067**	1.187
FRM → OCC	0.933	0.752	1.184	0.888
FRM → WOC	0.936	0.745	1.167	0.895
FRM → WRP	0.983	0.611	1.095	1.244

(c)

Table 8.2: Performance of the model evaluated with three metrics: intersection over union (a), average precision (b), and distance between centroids (c). The model is tested on different types of input and output: original frames (FRM), attention maps (ATT), linear occupancy grids (OCC), weighted occupancy grids (WOC), and warped occupancy grids (WRP). The results are organized in classes of depth: close range (CLO), middle range (MID), far range (FAR), and entire range (ALL).

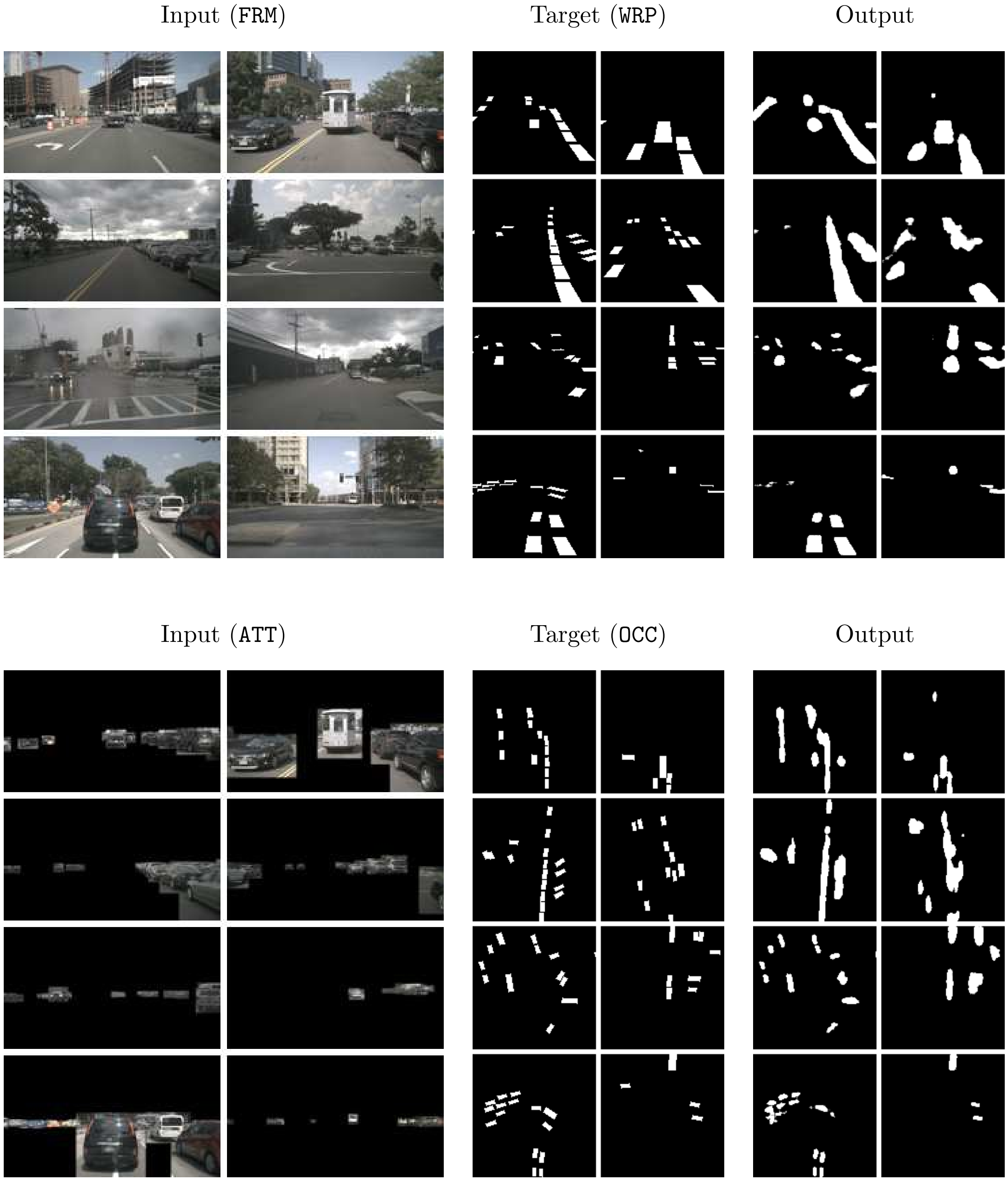


Figure 8.4: Visual results on eight test samples. The model is tested with two combinations of input and output: FRM → WRP (top) and ATT → OCC (bottom). The predicted output is displayed with a binarization threshold of 0.5.

Fig. 8.4 shows the visual results obtained on a selection of eight test samples by two of the model variations evaluated here: `FRM` → `WRP` and `ATT` → `OCC`. For clarity, the predicted outputs are rendered with a binarization threshold of 0.5. In both cases, it is possible to observe a general tendency to merge several vehicles together into a single "blob"; this could be a consequence of the current formulation of the loss function. Nonetheless, these preliminary results—both numeric and visual—appear certainly promising and reinforce the validity of this new direction of research.

8.4 Spaces of Motor Affordances

I conclude this chapter with a few words about a possible research direction towards a more sophisticated space of representation. Investigating a higher-level form of representation might clash with the overall spirit of my project—benefiting from the imitation of brain mechanisms. First, it is easier to investigate lower-level representation spaces because they are directly correlated with the input stimuli; it is difficult to trace back to the stimuli from motor-oriented representations. Secondly, the space of motor commands in the brain concerns only the activation of muscle fibers, and not the vehicle commands; there is a complex and indirect relationship between muscle activations and the control of the vehicle.

A potential direction is to design a representation space inspired by the concept of *affordances*. This term was introduced by James Gibson [79] to describe the value of a perceived object and what it provides to the observer, either in a beneficial or harmful way. The concept of affordance is still popular in modern cognitive science, although it manifests a notable drawback [88]: it postulates a direct mapping from optical flow to affordances, which is most likely implausible without several intermediate processing steps.

Another research line in neuroscience proposes that the brain motor representations space works with quantities similar, if not identical, to the quantities physically involved in the dynamic optimization of motor coordination problems. It is possible to find a correspondence between the typical quantities adopted in optimal control theory and the empirical data of coordination of limb actions [237, 226, 214]. In fact, arm movements tend to minimize the integral of the squared jerk (the third time derivative of position) over the time of a trajectory [130, 137].

The research group to which I belong has been extensively investigating the adoption of jerk in motor representation spaces. A prominent example is the *artificial motor cortex* [42], a representation space defined by two dimensions: the steering rate $\left(\frac{s}{m}\right)$ and the longitudinal jerk $\left(\frac{m}{s^3}\right)$. The magnitude of a point in the space represents the optimality for the corresponding motor control in the current trajectory—negative values are equal to inhibition. Another valid representation space takes into account also the time dependency of the trajectory [53]. The space is defined by the lateral jerk $\left(\frac{m}{s^2}\right)$ and the control horizon, i.e. the distance ahead of the car. The points of the space represents, again, the level of optimality. At the moment, the output is computed analytically; in my future work, I will explore the possibility of generating with artificial neural networks this form of high-level representation spaces, using as input perceptual and odometry data.

Chapter 9

Conclusions

In this dissertation, I have presented my research on visual perception for autonomous driving. Motivated by the superior human capability of driving, and following the key tenet of artificial intelligence, I have drawn inspiration from human cognition to try to design similar intelligent behaviors in an autonomous driving agent. In my research, I have identified four main neurocognitive principles relevant for the task of driving: visual mental imagery, perceptual learning, convergence-divergence zones, and predictive brain. To implement these principles, I have selected the most appropriate computational tools within the consolidated and successful field of deep learning. The solutions I have implemented make use of convolutional networks, autoencoders, variational Bayesian inference, and gated recurrent units.

The main contribution of my work is a method to learn to represent visual scenarios into compact vectors that are at once semantically organized and temporally coherent. My approach differs from other related works precisely in the learning of the representations: first, there is a semantic organization, in the sense that distinct parts of the representation are explicitly associated with specific concepts useful in the context of driving; second, the temporal coherence that is achieved through self-supervision allows the representation to be exploited for mental imagery and prediction of plausible future scenarios. I conclude this dissertation summarizing the main achievements obtained by the presented models, as well as their limitations, and illustrating future directions of my research.

9.1 Findings and Limitations

The most valuable findings of my research is probably the following: to have demonstrated that neurocognitive principles can be an effective "blueprint" to design mechanisms of perception for driving automation. It remains necessary, however, to seek a compromise between the cognitive inspiration and the computational advantages of deep learning methods. I have found the best trade-off to be the adoption of convolutional variational autoencoders to emulate the convergence-divergence zones in the brain. The result is a model able to code perceptual concepts using low-dimension representations. These representations are inspectable and interpretable in that they have a semantic organization, separating explicitly the concepts of `cars` and `lanes` entities found in the scene.

I have demonstrated the effectiveness of the conceptual representations by using them to predict the dynamics of future driving scenarios. The prediction is made possible by the temporal coherence of the latent space. This feature results from a refined learning procedure adopting self-supervision, which is a valid computational interpretation of the neurocognitive principle of predictive brains. The representations obtained by my model can be potentially applied to various downstream driving tasks; in this dissertation, I have presented the examples of predicting long-term future frames in a video sequence, emulating the phenomenon of mental imagery, and generating novel plausible scenarios using just linear interpolation. However, once learned, the representations can be deployed in many possible contexts. For example, I am currently working on using the representations to predict future occupancy grids.

It is important to point out the limitations I have found in my models so far. A significant limitation is that the models predict and perform imagery limitedly in the visual perceptual space. This space is inherently egocentric and two dimensional, and as such it cannot be directly exploited for vehicle control. A more methodological limitation is that the aim of imitating the human cognitive capabilities is partially hampered by the use of supervision—the model inevitably requires a supervised training to learn the conceptual organization. Alas, supervision cannot exist in the brain. Firstly, there is no such thing as a "ground truth" for biological neurons: there is no information that can be compared with the actual neural activation to compute a loss and modify the synaptic connections. Secondly, human learning comprises forms of associative rules, which share some similarity with artificial neural learning, but it also includes other forms like learning from linguistic descriptions. For example, a person can learn about cars and traffic rules just by reading about them in a manual for driving school. Moreover, supervision entails another drawback: learning requires large annotated datasets, which are not always available or of sufficient quality.

A further limitation of my models concerns the number of object categories taken into account in the space of conceptual representations. Humans have no limit in the number of different types of entities they can categorize in a scenario. However, the set of concepts actually necessary for the task of driving is more limited. For ease of being inspectable, I have chosen to considered only the two most crucial concepts, namely `cars` and `lanes`, although they are not sufficient when moving to more complex urban settings involving vulnerable road users.

9.2 Future Work

I conclude this dissertation with a list of developments I plan for the continuation of my research. The future work mainly addresses the current limitations I have illustrated in the previous section. There are essentially three main directions for future efforts that can improve my current models:

1. extending the range of data on which the models can be applied;
2. improving the biological plausibility;
3. adopting a higher-level space of representations.

Concerning the data, two approaches can be pursued: experimenting with additional datasets, and including more driving concepts into the learning process. The main models here presented are trained on the SYNTHIA dataset, which is one of the few large-scale datasets providing lane marking annotations. The recent release of the Berkeley DeepDrive dataset offers now an extended set of annotations of lane markings and drivable areas for 100,000 video frames. In addition, while SYNTHIA is a synthetic dataset, Berkeley DeepDrive has the advantage of being composed of real-world recordings. Hence, this dataset can be the perfect candidate for the first approach to extend the range of data. As far as implementing the second approach, the learning strategy applied for the two concepts `cars` and `lanes` can be reused as it is to learn additional concepts. The models can be expanded with new decoders dedicated to relevant conceptual entities not considered so far, e.g. vulnerable road users such as pedestrians and cyclists, or further road elements like traffic lights.

The biological plausibility of my current models is limited by the use of supervision to learn the separate conceptual representations. The research on how to avoid supervised learning has a long history and does not seem to have reached a conclusion. Even Hinton—one of the inventors of backpropagation—expressed his discontent with supervision and investigated several unsupervised alternatives [93, 94, 2, 48]. In more recent years, he experimented with some variations of standard supervision that are more close to the brain mechanisms of learning; however, the learning variations achieved much lower performance than the supervised counterpart [9]. The current research direction focuses on formulating a learning rule with properties halfway between standard supervision and synaptic plasticity [13], and it attempts to find learning strategies that could resemble features of human cognition like consciousness [10], incremental learning [168], and exemplar learning [39]. It is clear how the research on alternatives to supervision is still an open and rapidly evolving field, which needs to be regularly monitored.

Lastly, the future direction towards a more sophisticated space of representations is already ongoing. In addition to the main perceptual models, I have presented a work in progress exploiting a different representation space, namely the warped occupancy grid. This space has the interesting feature of being, at the same time, predisposed to be used for vehicle control and consistent with the human neurocognition. An additional future objective could be to obtain a full perceptual pipeline that from camera images projects representations in the motor representational space of the kind used in Dreams4Cars [53, 42].

Bibliography

[1] Anna Abraham, editor. *The Cambridge Handbook of the Imagination.* Cambridge University Press, Cambridge (UK), 2020.

[2] David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169, 1985.

[3] Alexander Amini, Wilko Schwarting, Guy Rosman, Brandon Araki, Sertac Karaman, and Daniela Rus. Variational autoencoder for end-to-end control of autonomous driving with novelty detection and training de-biasing. In *IEEE International Conference on Intelligent Robots and Systems*, pages 568–575, 2019.

[4] Alexander Andreopoulos and John K. Tsotsos. 50 years of object recognition: Directions forward. *Computer Vision and Image Understanding*, 117:827–891, 2013.

[5] Kartik Audhkhasi, Andrew Rosenberg, George Saon, Abhinav Sethy, and Bhuvana Ramabhadran. Recent progress in deep end-to-end models for spoken language processing. *IBM Journal of Research and Development*, 61:2:1–2:10, 2017.

[6] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 39:2481–2495, 2017.

[7] Haoyu Bai, Shaojun Cai, Nan Ye, David Hsu, and Wee Sun Lee. Intention-aware online POMDP planning for autonomous driving in a crowd. In *International Conference on Robotics and Automation*, pages 454–460, 2015.

[8] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst. *arXiv*, abs/1812.03079, 2018.

[9] Sergey Bartunov, Adam Santoro, Blake A. Richards, Luke Marris, Geoffrey E. Hinton, and Timothy Lillicrap. Assessing the scalability of biologically-motivated deep learning algorithms and architectures. In *Advances in Neural Information Processing Systems*, 2018.

[10] Yoshua Bengio. The consciousness prior. *arXiv*, abs/1709.08568, 2017.

[11] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:1798–1828, 2013.

[12] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems*, pages 153–160, 2007.

[13] Yoshua Bengio, Thomas Mesnard, Asja Fischer, Saizheng Zhang, and Yuhuai Wu. STDP-compatible approximation of backpropagation in an energy-based model. *Neural Computation*, 29:555–577, 2017.

[14] Pietro Berkes and Laurenz Wiskott. Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, 5:579–602, 2005.

[15] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *Proceedings of IEEE International Conference on Image Processing*, pages 3464–3468, 2016.

[16] Tim Bliss and Graham Collingridge. A synaptic model of memory: long-term potentiation in the hippocampus. *Nature*, 361:31–39, 1993.

[17] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, and Jake Zhao. End-to-end learning for self-driving cars. *arXiv*, abs/1604.07316, 2016.

[18] Mariusz Bojarski, Philip Yeres, Anna Choromanaska, Krzysztof Choromanski, Bernhard Firner, Lawrence Jackel, and Urs Muller. Explaining how a deep neural network trained with end-to-end learning steers a car. *arXiv*, abs/1704.07911, 2017.

[19] Richard Born, Alexander R. Trott, and Till S. Hartmann. Cortical magnification plus cortical plasticity equals vision? *Vision Research*, 111:161–169, 2015.

[20] L'eon Bottou and Yann LeCun. Large scale online learning. In *Advances in Neural Information Processing Systems*, pages 217–224, 2004.

[21] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv*, abs/1511.06349, 2015.

[22] Markus Braun, Sebastian Krebs, Fabian Flohr, and Dariu M Gavrila. Eurocity persons: A novel benchmark for person detection in traffic scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1844–1861, 2019.

[23] Rodney A Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–159, 1991.

[24] Tom Bruls, Horia Porav, Lars Kunze, and Paul Newman. The right (angled) perspective: Improving the understanding of road scenes using boosted inverse perspective mapping. In *IEEE Intelligent Vehicles Symposium*, pages 302–309, 2019.

[25] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Fredo Durand. What do different evaluation metrics tell us about saliency models? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:740–757, 2019.

[26] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 11621–11631, 2020.

[27] Pico Caroni, Flavio Donato, and Dominique Muller. Structural plasticity upon learning: regulation and functions. *Nature Reviews Neuroscience*, 13:478–490, 2012.

[28] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. DeepDriving: Learning affordance for direct perception in autonomous driving. In *Proc. of IEEE International Conference on Computer Vision*, 2015.

[29] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 40:834–848, 2017.

[30] Long Chen, Wen Tang, and Nigel John. Self-supervised monocular image depth learning and confidence estimation. *arXiv*, abs/1803.05530, 2018.

[31] Shitao Chen, Songyi Zhang, Jinghao Shang, Badong Chen, and Nanning Zheng. Brain-inspired cognitive model with attention for self-driving cars. *IEEE Transactions on Cognitive and Developmental Systems*, DOI 10.1109/TCDS.2017.2717451, 2017.

[32] Hong Cheng. *Autonomous Intelligent Vehicles – Theory, Algorithms, and Implementation.* Springer-Verlag, Berlin, 2011.

[33] Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frederic Dufaux. Self-supervised learning for autonomous vehicles perception: A conciliation between analytical and learning methods. *IEEE Transactions on Intelligent Vehicles*, abs/1910.01636, 2020.

[34] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734. Association for Computational Linguistics, 2014.

[35] Patricia Smith Churchland and Terrence J. Sejnowski. Neural representation and neural computation. *Philosophical Perspectives*, 4:343–382, 1990.

[36] Andy Clark. *Surfing Uncertainty: Prediction, Action and the Embodied Mind.* Oxford University Press, Oxford (UK), 2016.

[37] Brian Colder. Emulation as an integrating principle for cognition. *Frontiers in Human Neuroscience*, 5:Article 54, 2011.

[38] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, 2016.

[39] Aurelio Cortese, Benedetto De Martino, and Mitsuo Kawato. The neural and cognitive architecture for learning from a small sample. *Current Opinion in Neurobiology*, 55:133–141, 2019.

[40] Marc N. Coutanche and Sharon L. Thompson-Schill. Creating concepts from converging features in human cortex. *Cerebral Cortex*, 25:2584–2593, 2015.

[41] Mauro Da Lio, Francesco Biral, Enrico Bertolazzi, Marco Galvani, Paolo Bosetti, David Windridge, Andrea Saroldi, and Fabio Tango. Artificial co-drivers as a universal enabling technology for future intelligent vehicles and transportation systems. *IEEE Transactions on intelligent transportation systems*, 16(1):244–263, 2014.

[42] Mauro Da Lio, Riccardo Donà, Gastone Pietro Rosati Papini, and Kevin Gurney. Agent architecture for adaptive behaviors in autonomous driving. *IEEE Access*, 8:154906–154923, 2020.

[43] Mauro Da Lio, Alessandro Mazzalai, David Windridge, Serge Thill, Henrik Svensson, Mehmed Yüksel, Kevin Gurney, Andrea Saroldi, Luisa Andreone, Sean R Anderson, and Hermann-Josef Heich. Exploiting dream-like simulation mechanisms to develop safer agents for automated driving: The "dreams4cars" eu research and innovation action. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2017.

[44] Mauro Da Lio, Alice Plebe, Daniele Bortoluzzi, Gastone Pietro Rosati Papini, and Riccardo Donà. Autonomous vehicle architecture inspired by the neurocognition of human driving. In *Proceedings of the 4th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS)*, pages 507–513. Science and Technology Publications, 2018.

[45] Antonio Damasio. The brain binds entities and events by multiregional activation from convergence zones. *Neural Computation*, 1:123–132, 1989.

[46] Antonio Damasio. Time-locked multiregional retroactivation: A systems-level proposal for the neural substrates of recall and recognition. *Cognition*, 33:25–62, 1989.

[47] Radu Danescu, Florin Oniga, and Sergiu Nedevschi. Modeling and tracking the driving environment with a particle-based occupancy grid. *IEEE Transactions on Intelligent Transportation Systems*, 12:1331–1342, 2011.

[48] Peter Dayan, Geoffrey E. Hinton, Radford M.Neal, and Richard S. Zemel. The Helmholtz machine. *Neural Computation*, 7:889–904, 1995.

[49] Gustavo Deco. Biased competition mechanisms for visual attention in a multimodular neurodynamical system. In Stefan Wermter, Jim Austin, and David Willshaw, editors, *Emergent neural computational architectures based on neuroscience: towards neuroscience-inspired computing*, pages 114–126. Springer-Verlag, Berlin, 2001.

[50] Gustavo Deco and Edmund Rolls. A neurodynamical cortical model of visual attention and invariant object recognition. *Vision Research*, 44:621–642, 2004.

[51] R. Desimone. Neural circuits for visual attention in the primate brain. In G. A. Carpenter and S. Grossberg, editors, *Neural Networks for Vision and Image Processing*. MIT Press, Cambridge (MA), 1992.

[52] Samuel F. Dodge and Lina J. Karam. Visual saliency prediction using a mixture of deep neural networks. *IEEE Transactions on Image Processing*, 27:4080–4090, 2018.

[53] Riccardo Donà, Gastone Pietro Rosati Papini, Mauro Da Lio, and Luca Zaccarian. On the stability and robustness of hierarchical vehicle lateral control with inverse/forward dynamics quasi-cancellation. *IEEE Transactions on Vehicular Technology*, 68:10559–10570, 2019.

[54] Barbara Dosher and Zhong-Lin Lu. *Perceptual Learning: How Experience Shapes Visual Perception*. MIT Press, Cambridge (MA), 2020.

[55] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[56] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[57] Robert O. Duncan and Geoffrey M. Boynton. Cortical magnification within human primary visual cortex correlates with acuity thresholds. *Neuron*, 38:659–671, 2003.

[58] Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22:46–57, 1989.

[59] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14:179–221, 1990.

[60] Hesham M. Eraqi, Mohamed N. Moustafa, and Jens Honer. End-to-end deep learning for steering autonomous vehicles considering temporal dependencies. *arXiv*, abs/1710.03804, 2017.

[61] Özgür Erkent, Christian Wolf, Christian Laugier, David Sierra Gonzalez, and Victor Romero Cano. Semantic grid estimation with a hybrid Bayesian and deep neural network approach. In *International Conference on Intelligent Robots and Systems*, pages 1–8, 2018.

[62] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[63] Udo Di Fabio, Manfred Broy, Renata Jungo Brungger, Ulrich Eichhorn, Armin Grunwald, and Dirk Heckmann. Ethics commission automated and connected driving. Technical report, Federal Minister of Transport and Digital Infrastructure, Germany, 2017.

[64] Manfred Fahle and Tomaso Poggio, editors. *Perceptual Learning*. MIT Press, Cambridge (MA), 2002.

[65] Martha J. Farah. Psychophysical evidence for a shared representational medium for mental images and percepts. *Journal of Experimental Psychology: General*, 114:91–103, 1985.

[66] Daniel E. Feldman. Synaptic mechanisms for plasticity in neocortex. *Annual Review of Neuroscience*, 32:33–55, 2009.

[67] Michael Felsberg, Per-Erik Forssén, and Hanno Scharr. Channel smoothing: Efficient robust smoothing of low-level signal features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:209–222, 2006.

[68] Michael Felsberg, Hanno Scharr, and Per-Erik Forssén. The B-spline channel representation: Channel algebra and channel based diffusion filtering. Technical Report LiTH-ISY-R-2461, Linköping University, Linköping (SW), 2002.

[69] Justin C. Fisher. Does simulation theory really involve simulation? *Plenum Press*, 19:417–432, 2006.

[70] Per-Erik Forssén, Gösta Granlund, and Johan Wiklund. Channel representation of colour images. Technical Report LiTH-ISY-R-2418, Linköping University, Linköping (SW), 2002.

[71] W. J. Freeman. *Neurodynamics: an exploration of the Mesoscopic Brain Dynamics*. Springer-Verlag, Berlin, 2000.

[72] Karl Friston. Learning and inference in the brain. *Neural Networks*, 16:1325–1352, 2003.

[73] Karl Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11:127–138, 2010.

[74] Karl Friston. A free energy principle for biological systems. *Entropy*, 14:2100–2121, 2012.

[75] Karl Friston and Stefan Kiebel. Predictive coding under the free–energy principle. *Philosophical transactions of the Royal Society B*, 364:1211–1221, 2009.

[76] Karl Friston and Klaas E. Stephan. Free–energy and the brain. *Synthese*, 159:417–458, 2007.

[77] Kunihiko Fukushima. Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[78] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.

[79] James J Gibson. *The Ecological Approach to Perception.* Houghton Miflin, Boston (MA), 1979.

[80] Alvin Goldman. *Simulating Minds: The Philosophy, Psychology, and Neuroscience of Mindreading.* Oxford University Press, Oxford (UK), 2006.

[81] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[82] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, doi.org/10.1002/rob.21918:1–25, 2019.

[83] Rick Grush. The emulation theory of representation: Motor control, imagery, and perception. *Behavioral and Brain Science*, 27:377–442, 2004.

[84] Umut Güçlü and Marcel A J van Gerven. Deep neural networks reveal a gradient in the complexity of neural representations across the ventral stream. *Journal of Neuroscience*, 35:10005–10014, 2015.

[85] David Ha and Jürgen Schmidhuber. World models. *arXiv*, abs/1803.10122, 2018.

[86] Danijar Hafner, Pedro A. Ortega, Jimmy Ba, Thomas Parr, Karl Friston, and Nicolas Heess. Action and perception as divergence minimization. *arXiv*, abs/2009.01791, 2020.

[87] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[88] Manuel Heras-Escribano. *The Philosophy of Affordances*. Palgrave Macmillan, London, 2019.

[89] Germund Hesslow. Conscious thought as simulation of behaviour and perception. *Trends in Cognitive Sciences*, 6:242–247, 2002.

[90] Germund Hesslow. The current status of the simulation theory of cognition. *Brain*, 1428:71–79, 2012.

[91] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[92] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 28:504–507, 2006.

[93] Geoffrey E. Hinton and Terrence J. Sejnowski. Optimal perceptual inference. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 448–453, New York, 1983.

[94] Geoffrey E. Hinton, Terrence J. Sejnowski, and David H. Ackley. Boltzmann machines: Constraint networks that learn. Technical Report 84-119, Carnegie-Mellon University, Computer Science Department, 1984.

[95] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*, abs/1207.0580, 2012.

[96] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[97] Stefan Hoermann, Martin Bach, and Klaus Dietmayer. Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling. In *International Conference on Robotics and Automation*, pages 2056–2063, 2018.

[98] Jakob Hohwy. *The Predictive Mind*. Oxford University Press, Oxford (UK), 2013.

[99] Yu Huang and Yue Chen. Autonomous driving with deep learning: A survey of state-of-art technologies. *arXiv*, abs/2006.06091, 2020.

[100] David Hubel and Torsten Wiesel. Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962.

[101] David Hubel and Torsten Wiesel. Uniformity of monkey striate cortex: a parallel relationship between field size, scatter, and magnification factor. *Journal of Comparative Neurology*, 158:295–305, 1974.

[102] Daniel D. Hutto and Erik Myin. *Radicalizing enactivism: basic minds without content*. MIT Press, Cambridge (MA), 2013.

[103] Shantanu Ingle and Madhuri Phute. Tesla autopilot: Semi autonomous driving, an uptick for future autonomy. *International Research Journal of Engineering and Technology*, 3:369–372, 2016.

[104] David Isele, Reza Rahimi, Akansel Cosgun, Kaushik Subramanian, and Kikuo Fujimura. Navigating occluded intersections with autonomous vehicles using deep reinforcement learning. In *International Conference on Robotics and Automation*, pages 2034–2039, 2018.

[105] Alumit Ishai and Dov Sagi. Common mechanisms of visual imagery and perception. *Science*, 268:1772–1774, 1995.

[106] Laurent Itti and Christof Koch. Computational modelling of visual attention. *Nature Reviews Neuroscience*, 2:194–203, 2001.

[107] Pierre Jacob and Marc Jeannerod. *Ways of Seeing – The Scope and Limits of Visual Cognition*. Oxford University Press, Oxford (UK), 2003.

[108] Joel Janai, Fatma Güney, Aseem Behl, and Andreas Geiger. Computer vision for autonomous vehicles, problems, datasets and state of the art. *Foundations and Trends in Computer Graphics and Vision*, 12:1–308, 2020.

[109] Hyeong-Seok Jeon, Dong-Suk Kum, and Woo-Yeol Jeong. Traffic scene prediction via deep learning: Introduction of multi-channel occupancy grid map as a scene representation. In *IEEE Intelligent Vehicles Symposium*, pages 2104–2110, 2018.

[110] Lai Jiang, Mai Xu, Zulin Wang, and Leonid Sigal. DeepVS2.0: A saliency-structured deep learning method for predicting dynamic visual attention. *International Journal of Computer Vision*, 10.1007/s11263-020-01371-6, 2020.

[111] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, early access, 2020.

[112] Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viegas andMartin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017.

[113] Rudolf E. Kálmán. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960.

[114] Nidhi Kalra and Susan 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.

[115] Nancy Kanwisher and Ewa Wojciulik. Visual attention: insights from brain imaging. *Nature Reviews Neuroscience*, 1:3310–3318, 2000.

[116] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet. Lyft level 5 perception dataset 2020. `https://level5.lyft.com/dataset`, 2019.

[117] Seyed-Mahdi Khaligh-Razavi and Nikolaus Kriegeskorte. Deep supervised, but not unsupervised, models may explain it cortical representation. *PLoS Computational Biology*, 10:e1003915, 2014.

[118] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations*, 2014.

[119] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of International Conference on Learning Representations*, 2014.

[120] Michael D. Kirchhoff. Predictive processing, perceiving and imagining: Is to perceive to imagine, or something close to it? *Philosophical Studies*, 175:751–767, 2018.

[121] Stephen M Kosslyn. *Image and Mind*. Harvard University Press, Cambridge (MA), 1980.

[122] Stephen M Kosslyn. *Image and Brain: the Resolution of the Imagery Debate*. MIT Press, Cambridge (MA), 1994.

[123] Sofia Krasovskaya and W. Joseph MacInnes. Salience models: A computational cognitive neuroscience review. *Vision*, 3:vision3040056, 2019.

[124] Nikolaus Kriegeskorte and Jörg Diedrichsen. Peeling the onion of brain representations. *Annual Review of Neuroscience*, 42:407–432, 2019.

[125] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1090–1098, 2012.

[126] Srinivas S. S. Kruthiventi, Kumar Ayush, and R. Venkatesh Babu. DeepFix: A fully convolutional neural network for predicting human eye fixations. *IEEE Transactions on Image Processing*, 26:4446–4456, 2017.

[127] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[128] Tejas D. Kulkarni, William F. Whitney, Pushmeet Kohli, and Joshua B. Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547, 2015.

[129] Sampo Kuutti, Saber Fallah, Richard Bowden, and Phil Barber. Deep learning for autonomous vehicle control – algorithms, state-of-the-art, and future prospects. *Synthesis Lectures on Advances in Automotive Technology*, 3:1–80, 2019.

[130] Jozsef Laczko, Slobodan Jaric, Jozsef Tihanyi, Vladimir M. Zatsiorsky, and Mark L. Latash. Components of the end-effector jerk during voluntary arm movements. *Journal of Applied Biomechanics*, 16:14–25, 2000.

[131] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 6874–6883, 2017.

[132] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[133] Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551, 1989.

[134] Carissa M Lemon, Denton DeLoss, and George J Andersen. Training to improve collision detection in older adults. In *Ninth International Driving Symposium on Human Factors in Driver Assessment*, pages 305–311, 2017.

[135] Xiaofei Li, Fabian Flohr, Yue Yang, Hui Xiong, , Markus Braun, Shuyue Pan, Keqiang Li, and Dariu M. Gavrila. A new benchmark for vison-based cyclist detection. In *IEEE Intelligent Vehicles Symposium*, pages 1028–1033, 2016.

[136] Zachary C. Lipton. The mythos of model interpretability. In *ICML Workshop on Human Interpretability in Machine Learning*, pages 96–100, 2016.

[137] Dan Liu and Emanuel Todorov. Evidence for the flexible sensorimotor strategies predicted by optimal feedback control. *Journal of Neuroscience*, 27:9354–9368, 2007.

[138] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[139] Chenyang Lu, Marinus Jacobus Gerardus van de Molengraft, and Gijs Dubbelman. Monocular semantic occupancy grid mapping with convolutional variational encoder–decoder networks. *IEEE Robotics and Automation Letters*, 4:445–452, 2019.

[140] Hanspeter A Mallot, Heinrich H. Bülthoff, J. J. Little, and Stefan Bohrer. Inverse perspective mapping simplifies optical flow computation and obstacle detection. *Biological Cybernetics*, 3:177–185, 1991.

[141] Kingson Man, Jonas Kaplan, Hanna Damasio, and Antonio Damasio. Neural convergence and divergence in the mammalian cerebral cortex: from experimental neuroanatomy to functional neuroimaging. *The Journal of Comparative Neurology*, 521:4097–4111, 2013.

[142] Liviu A. Marina, Bogdan Trasnea, Cocias Tiberiu, Andrei Vasilcoi, Florin Moldoveanu, and Sorin M. Grigorescu. Deep grid net (DGN) a deep learning system for real-time driving context understanding. In *International Conference on Robotic Computing*, pages 399–402, 2019.

[143] Henry Markram, Eilif Muller, Srikanth Ramaswamy, and Michael W. Reimann et al. Reconstruction and simulation of neocortical microcircuitry. *Cell*, 163:456–492, 2015.

[144] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* W. H. Freeman, San Francisco (CA), 1982.

[145] Marsel Mesulam. From sensation to cognition. *Trends in Cognitive Sciences*, 2:455–462, 1998.

[146] Cade Metz. The sadness and beauty of watching Google's AI play Go. *Wired*, March 11, 2016.

[147] Kaspar Meyer and Antonio Damasio. Convergence and divergence in a neural architecture for recognition and memory. *Trends in Neuroscience*, 32:376–382, 2009.

[148] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *arXiv*, abs/2001.05566, 2020.

[149] Marvin Minsky and Seymour Papert. *Perceptrons: An introduction to computational geometry.* MIT Press, Cambridge (MA), 1969.

[150] Amar Mitiche and Ismail Ben Ayed. *Variational and Level Set Methods in Image Segmentation.* Springer-Verlag, Berlin, 2010.

[151] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[152] Samuel T. Moulton and Stephen M Kosslyn. Imagining predictions: mental imagery as mental emulation. *Philosophical transactions of the Royal Society B*, 364:1273–1280, 2009.

[153] Sajjad Mozaffari, Omar Y. Al-Jarrah, Mehrdad Dianati, Paul Jennings, and Alexandros Mouzakitis. Deep learning-based vehicle behavior prediction for autonomous driving applications: A review. *IEEE Transactions on Intelligent Transportation Systems*, early access:1–15, 2020.

[154] Urs Muller, Jan Ben, Eric Cosatto, Beat Flepp, and Yann LeCun. Off-road obstacle avoidance through end-to-end learning. In *Advances in Neural Information Processing Systems*, pages 739–746, 2006.

[155] David Mumford and Jayant Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42:577–685, 1989.

[156] Scott O. Murray and Sheng He. Contrast invariance in the human lateral occipital complex depends on attention. *Cerebral Cortex*, 16:606–611, 2001.

[157] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proc. of IEEE International Conference on Computer Vision*, pages 4990–4999, 2017.

[158] Alejandro Newell and Jia Deng. How useful is self-supervised pretraining for visual tasks? In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 7345–7355, 2020.

[159] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 427–436, 2015.

[160] Alva Noë. *Action in Perception*. MIT Press, Cambridge (MA), 2004.

[161] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Proceedings of European Conference on Computer Vision*, pages 868–884, 2016. Part IV.

[162] André Ofner and Sebastian Stober. Towards bridging human and artificial cognition: Hybrid variational predictive coding of the physical world, the body and the brain. In *Advances in Neural Information Processing Systems*, 2018.

[163] Juan Sebastian Olier, Emilia Barakova, Carlo Regazzoni, and Matthias Rauterberg. Reframing the characteristics of concepts and their relation to learning and cognition in artificial agents. *Cognitive Systems Research*, 44:50–68, 2017.

[164] J Kevin O'Regan and Alva Noë. A sensorimotor account of vision and visual consciousness. *Behavioral and Brain Science*, 24:939–1031, 2001.

[165] World Health Organization. Global status report on road safety: summary, 2018.

[166] Denis Osipychev, Duy Tran, Weihua Sheng, Girish Chowdhary, and Ruili Zeng. Proactive MDP-based collision avoidance algorithm for autonomous cars. In *International Conference on Cyber Technology in Automation, Control, and Intelligent Systems*, pages 983–988, 2015.

[167] Ümit Özgüner, Tankut Acarman, and Keith Redmill. *Autonomous Ground Vehicles*. Artech House, London, 2011.

[168] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[169] Michel Pasquier and Richard J. Oentaryo. Learning to drive the human way: a step towards intelligent vehicles. *International Journal of Vehicle Autonomous Systems*, 6:24–47, 2008.

[170] Gaurav H. Patel, David M. Kaplan, and Lawrence H. Snyder. Topographic organization in the brain: searching for general principles. *Trends in Cognitive Sciences*, 18:351–363, 2014.

[171] Chris Paxton, Vasumathi Raman, Gregory D. Hager, and Marin Kobilarov. Combining neural networks and tree search for task and motion planning in challenging environments. In *International Conference on Intelligent Robots and Systems*, pages 749–755, 2017.

[172] Joel Pearson, Colin W.G. Clifford, and Frank Tong. The functional impact of mental imagery on conscious perception. *Current Biology*, 18:982–986, 2008.

[173] Joel Pearson and Stephen M Kosslyn. The heterogeneity of mental representation: Ending the imagery debate. *Proceedings of the Natural Academy of Science USA*, 112:10089–10092, 2015.

[174] Alice Plebe, Vincenzo Cutello, and Mario Pavone. Optimizing costs and quality of interior lighting by genetic algorithm. In *Studies in Computational Intelligence*, volume 829, pages 19–39. Springer, Cham, 2019.

[175] Alice Plebe and Mauro Da Lio. Variational autoencoder inspired by brain's convergence-divergence zones for autonomous driving application. In *Proceedings of the 20th International Conference on Image Analysis and Processing (ICIAP)*, volume 11751 of *Lecture Notes in Computer Science*, pages 367–377. Springer, Cham, 2019.

[176] Alice Plebe and Mauro Da Lio. Visual perception for autonomous driving inspired by convergence–divergence zones. In *Proceedings of the 11th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 204–208. IEEE, 2019.

[177] Alice Plebe and Mauro Da Lio. On the road with 16 neurons: Towards interpretable and manipulable latent representations for visual predictions in driving scenarios. *IEEE Access*, 8:179716–179734, 2020.

[178] Alice Plebe, Mauro Da Lio, and Daniele Bortoluzzi. On reliable neural network sensorimotor control in autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 21:711–722, 2020.

[179] Alice Plebe, Riccardo Donà, Gastone Pietro Rosati Papini, and Mauro Da Lio. Mental imagery for intelligent vehicles. In *Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS)*, pages 43–51. Science and Technology Publications, 2019.

[180] Alice Plebe and Giorgio Grasso. Particle physics and polyedra proximity calculation for hazard simulations in large-scale industrial plants. In *Proceedings of the 12th International Conference of Computational Methods in Sciences and Engineering (ICCMSE)*, pages 090003–1–090003–4. American Institute of Physics Publishing, 2016.

[181] Alice Plebe and Giorgio Grasso. Conceptual integrity without concepts. *International Journal of Software Engineering and Knowledge Engineering*, 28(7):955–981, 2018.

[182] Alice Plebe and Mario Pavone. Multi-objective genetic algorithm for interior lighting design. In *Proceedings of the 3rd International Workshop on Machine learning, Optimization, and Big Data (MOD)*, volume 10710 of *Lecture Notes in Computer Science*, pages 222–233. Springer, Cham, 2017.

[183] Alice Plebe, Gastone Pietro Rosati Papini, Riccardo Donà, and Mauro Da Lio. Dreaming mechanism for training bio-inspired driving agents. In *Proceedings of the 2nd International Conference on Intelligent Human Systems Integration (IHSI)*, pages 429–434. Springer, Cham, 2019.

[184] Alexander D. Protopapas, Michael Vanier, and James M. Bower. Simulating large networks of neurons. In Christof Koch and Idan Segev, editors, *Methods in Neuronal Modeling from Ions to Networks*. MIT Press, Cambridge (MA), 1998. second edition.

[185] Zenon Pylyshyn. What the mind's eye tells the mind's brain: A critique of mental imagery. *Psychological Bulletin*, 80:1, 1973.

[186] Gabriëlle Ras, Marcel van Gerven, and Pim Haselager. Explanation methods in deep learning: Users, values, concerns and challenges. In Hugo Jair Escalante, Sergio Escalera, Isabelle Guyon, Xavier Baró, Yagmur Güçlütürk, Umut Güçlü, and Marcel van Gerven, editors, *Explainable and Interpretable Models in Computer Vision and Machine Learning*, pages 19–36. Springer-Verlag, Berlin, 2018.

[187] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. *arXiv*, abs/1804.02767, 2018.

[188] Eike Rehder, Maximilian Naumann, Niels Ole Salscheider, and Christoph Stiller. Cooperative motion planning for non-holonomic agents with value iteration networks. *arXiv*, abs/1709.05273, 2017.

[189] John H. Reynolds, Leonardo Chelazzi, and Robert Desimone. Competitive mechanisms subserve attention in macaque areas V2 and V4. *Nature Neuroscience*, 2:1019–1025, 1999.

[190] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of Machine Learning Research*, pages 1278–1286, 2014.

[191] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *Proc. of European Conference on Computer Vision*, pages 102–118, 2016.

[192] Pieter R. Roelfsema, Arjen van Ooyen, and Takeo Watanabe. Perceptual learning rules based on reinforcers and attention. *Trends in Cognitive Sciences*, 14:64–71, 2009.

[193] Edmund Rolls. *Cerebral Cortex: Principles of Operation.* Oxford University Press, Oxford (UK), 2016.

[194] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[195] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3234–3243, 2016.

[196] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organisation in the brain. *Psychological Review*, 65:386–408, 1958.

[197] Frank Rosenblatt. *Principles of Neurodynamics: Perceptron and the Theory of Brain Mechanisms.* Spartan, Washington (DC), 1962.

[198] Azriel Rosenfeld. *Picture Processing by Computer.* Academic Press, New York, 1969.

[199] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[200] David E. Rumelhart and James L. McClelland, editors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* MIT Press, Cambridge (MA), 1986.

[201] Dan Ryder. Problems of representation II: naturalizing content. In John Symons and Paco Calvo, editors, *The Routledge Companion to Philosophy of Psychology*, pages 251–279. Routledge, London, 2009.

[202] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive GAN for predicting paths compliant to social and physical constraints. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1349–1358, 2018.

[203] Ruslan R. Salakhutdinov and Geoffrey E. Hinton. Deep Boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, pages 448–455, 2009.

[204] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv*, abs/1708.08296, 2017.

[205] Eder Santana and George Hotz. Learning a driving simulator. *arXiv*, abs/1608.01230, 2016.

[206] Yuka Sasaki, Jose E. Nanez, and Takeo Watanabe. Advances in visual perceptual learning and plasticity. *Nature Reviews Neuroscience*, 11:53–60, 2010.

[207] Jürger Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[208] Yannick Schulz, Avinash Kini Mattar, Thomas Hehn, and Julian Kooij. Hearing what you cannot see: Acoustic vehicle detection around corners. *IEEE Robotics and Automation Letters*, 2021.

[209] Wilko Schwarting, Alyssa Pierson, Javier Alonso-Mora, Sertac Karaman, and Daniela Rus. Social behavior for autonomous vehicles. *Proceedings of the Natural Academy of Science USA*, 116:24972–24978, 2019.

[210] Christoph Seeger, Michael Manz, Patrick Matters, and Joachim Hornegger. Locally adaptive discounting in multi sensor occupancy grid fusion. In *IEEE Intelligent Vehicles Symposium*, pages 266–271, 2016.

[211] Aaron R. Seitz. Perceptual learning. *Current Biology*, 27:R631–R636, 2017.

[212] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

[213] Weijing Shi, Mohamed Baker Alawieh, Xin Li, and Huafeng Yu. Algorithm and hardware implementation for visual perception system in autonomous vehicle: A survey. *Integration*, 59:148–156, 2017.

[214] Lior Shmuelof, John W. Krakauer, and Pietro Mazzoni. How is a motor skill learned? change and invariance at the levels of task success and trajectory control. *Journal of Neurophysiology*, 108:578–594, 2012.

[215] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[216] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, abs/1409.1556, 2015.

[217] Santokh Singh. Critical reasons for crashes investigated in the National Motor Vehicle Crash Causation Survey. Technical Report DOT HS 812 115, National Highway Traffic Safety Administration, Washington (DC), 2015.

[218] Ganesh Sistu, Isabelle Leang, Sumanth Chennupati, Stefan Milz, Senthil Yogamani, and Samir Rawashdeh. NeurAll: Towards a unified model for visual perception in automated driving. *arXiv*, abs/1902.03589, 2019.

[219] Marc A. Sommer and Robert H. Wurtz. Composition and topographic organization of signals sent from the frontal eye field to the superior colliculus. *Journal of Neurophysiology*, 83:1979–2001, 2000.

[220] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In Jorge Cardoso, Tal Arbel, Gustavo Carneiro, Tanveer Syeda-Mahmood, J Manuel R.S. Tavares, Mehdi Moradi, Andrew Bradley, Hayit Greenspan, J Paulo Papa, Anant Madabhushi, Jacinto C. Nascimento, Jaime S. Cardoso, Vasileios Belagiannis, and Zhi Lu, editors, *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 240–248, 2017.

[221] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.

[222] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv*, abs/1312.6199, 2013.

[223] Bertrand Thirion, Edouard Duchesnay, Edward Hubbard, Jessica Dubois, Jean-Baptiste Poline, Denis Lebihan, and Stanislas Dehaene. Inverse retinotopy: Inferring the visual content of images from brain activation patterns. *NeuroImage*, 33:1104–1116, 2006.

[224] Jean-Philippe Thivierge and Gary F. Marcus. The topographic brain: from neural connectivity to cognition. *Trends in Neuroscience*, 30:251–259, 2007.

[225] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics.* MIT Press, Cambridge (MA), 2006.

[226] Emanuel Todorov. Optimality principles in sensorimotor control. *Nature Neuroscience*, 7:907–915, 2004.

[227] Kay Ueltzhöffer. Deep active inference. *Biological Cybernetics*, 112:547–573, 2018.

[228] Jessica Van Brummelen, Marie O'Brien, Dominique Gruyer, and Homayoun Najjaran. Autonomous vehicle perception: The technology of today and tomorrow. *Transportation Research Part C: Emerging Technologies*, 89:384–406, 2018.

[229] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv*, abs/1807.03748, 2019.

[230] Václav Šmídl and Anthony Quinn. *The Variational Bayes Method in Signal Processing.* Springer-Verlag, Berlin, 2005.

[231] Dequan Wang, Coline Devin, Qi-Zhi Cai, Fisher Yu, and Trevor Darrell. Deep object-centric policies for autonomous driving. In *International Conference on Robotics and Automation*, pages 8853–8859, 2019.

[232] Thorsten Weiss, Bruno Schiele, and Klaus Dietmayer. Robust driving path detection in urban and highway scenarios using a laser scanner and online occupancy grids. In *IEEE Intelligent Vehicles Symposium*, pages 184–189, 2007.

[233] Adrian Weller. Challenges for transparency. In *ICML Workshop on Human Interpretability in Machine Learning*, pages 55–62, 2017.

[234] Tim A. Wheeler, Philipp Robbel, and Mykel J. Kochenderfer. A probabilistic framework for microscopic traffic propagation. In *International Conference on Intelligent Transportation Systems*, pages 262–267, 2015.

[235] Nick Wiltsher. Imagination: A lens, not a mirror. *Philosophers' Imprint*, 19:no. 30, 2019.

[236] David Windridge, Henrik Svensson, and Serge Thill. On the utility of dreaming: A general model for how learning in artificial agents can benefit from data hallucination. *Adaptive Behavior*, https://doi.org/10.1177/1059712319896489, 2020.

[237] Daniel M Wolpert. Computational approaches to motor control. *Trends in Cognitive Sciences*, 1:209–216, 1997.

[238] Daniel M Wolpert, Jörg Diedrichsen, and Randall Flanagan. Principles of sensorimotor learning. *Nature Reviews Neuroscience*, 12:739–751, 2011.

[239] Daniel M Wolpert, Zoubin Ghahramani, and Michael I Jordan. An internal model for sensorimotor integration. *Science*, 269:1880–1882, 1995.

[240] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3530–3538, 2017.

[241] Ying Yang, Michael J Tarr, and Robert E Kass amd Elissa M Aminoff. Exploring spatio–temporal neural dynamics of the human visual cortex. *Human Brain Mapping*, 40:4213–4238, 2019.

[242] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2636–2645, 2020.

[243] Carlos Zednik. Solving the black box problem: A normative framework for explainable artificial intelligence. *Philosophy & Technology*, pages 1–24, 2019.

[244] Xinyu Zhang, Mo Zhou, Huaping Liu, and Amir Hussain. A cognitively inspired system architecture for the *Mengshi* cognitive vehicle. *Cerebral Cortex*, doi.org/10.1007/s12559-019-09692-6:1–10, 2019.

[245] Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann LeCun. Stacked what-where autoencoders. In *International Conference on Learning Representations*, pages 1–12, 2016.