UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Maksim Ploter

Eyes Wide Shut: Analyzing Object Detection Performance Under Degraded Sensor Input Scenarios

Master's Thesis (30 ECTS)

Supervisor:
Tambet Matiisen, MSc

Tartu 2025

Eyes Wide Shut: Analyzing Object Detection Performance Under Degraded Sensor Input Scenarios

Abstract:

Autonomous Driving Systems (ADS) promise safer roads, better traffic flow, and reduced environmental impact. The Society of Automotive Engineers (SAE) International's J3016 standard [1] stipulates stringent safety requirements for ADS, particularly concerning their operational behavior during dynamic driving task performance-relevant system failures. The perception task, which includes the fundamental computer vision task of object detection, is a key capability that distinguishes ADS from a "regular" vehicle. In the last decade, there has been remarkable progress in various computer vision tasks, and the object detection task in particular. However, many contemporary state-of-the-art models are specialists, with strong inductive biases for specific data types, making them difficult, if not impossible, to use for ADS. To address this limitation, the thesis introduces two novel recurrent architectures: the Recurrent Perceiver (RPerceiver) and its multi-modal variant, the Recurrent Perceiver Multi-Modal (RPerceiverMM). The efficacy of these architectures was evaluated on a novel benchmark dataset, "detection-moving-mnist-easy", proposed in this thesis. The experimental results suggest the proposed models' effectiveness in leveraging temporal information, particularly in challenging cases such as objects that are partially visible while leaving the video frame. Furthermore, this research investigated specific training procedures designed to simulate complete sensor failures and non-deterministic data availability. The findings indicate that these proposed training strategies significantly improve model robustness, demonstrating enhanced performance when faced with conditions analogous to real-world ADS sensor system failures. This work contributes to the development of more resilient perception systems crucial for the safe deployment of ADS. The code was open-sourced at GitHub [1].

Keywords: Autonomous Driving Systems, Deep Learning, Transformer, Computer Vision, Video Object Detection

CERCS: P170, P175, P176, T111

[1] https://github.com/maxploter/MultiSensorDropout/

Silmad pärani kinni: objektituvastuse jõudluse analüüs halvenenud sensori sisendiga olukordades

Lühikokkuvõte:

Autonoomsed juhtimissüsteemid (ADS) lubavad ohutumaid teid, paremat liiklusvoogu ja väiksemat keskkonnamõju. Autotööstuse Inseneride Ühingu (SAE) standard J3016 [1] sätestab ranged ohutusnõuded ADS-dele, eriti mis puudutab nende talitlust dünaamilise sõiduülesande täitmisega seotud süsteemirikete korral. Tajufunktsioon, mis hõlmab fundamentaalset arvutinägemise ülesannet – objektituvastust –, on võtmevõimekus, mis eristab ADS-e tavalistest sõidukitest. Viimasel kümnendil on toimunud märkimisväärne edasiminek erinevates arvutinägemise ülesannetes, eriti objektituvastuses. Paljud kaasaegsed tipptasemel mudelid on aga spetsialiseerunud, omades tugevaid induktiivseid eeldusi konkreetsete andmetüüpide suhtes, mis muudab nende kasutamise ADS-de jaoks keeruliseks, kui mitte võimatuks. Selle piirangu lahendamiseks tutvustab käesolev magistritöö kahte uudset rekurrentset arhitektuuri: Rekurrentne Perceiver (RPerceiver) ja selle mitmemodaalne variant Rekurrentne Perceiver MitmeModaalne (RPerceiverMM). Nende arhitektuuride tõhusust hinnati käesolevas töös välja pakutud uudsel võrdlusandmestikul ”detection-moving-mnist-easy”. Eksperimentaalsed tulemused viitavad väljapakutud mudelite efektiivsusele ajalise teabe ärakasutamisel, eriti keerulistes olukordades, näiteks objektide puhul, mis on kaadrist väljudes osaliselt nähtavad. Lisaks uuriti selles töös spetsiifilisi treeningprotseduure, mis on loodud simuleerima täielikke anduririkkeid ja andmete mittedeterministlikku kättesaadavust. Tulemused näitavad, et need väljapakutud treeningstrateegiad parandavad oluliselt mudeli robustsust, demonstreerides paremat jõudlust tingimustes, mis sarnanevad reaalsete ADS-i andurisüsteemide riketele. See töö aitab kaasa vastupidavamate tajusüsteemide arendamisele, mis on ADS-ide ohutu kasutuselevõtu jaoks üliolulised.

Võtmesõnad: Autonoomsed Juhtimissüsteemid, Süvaõpe, Transformer, Masinnägemine, Video Objektituvastus

CERCS: P170, P175, P176, T111

Contents

1. Introduction 5
2. Background 7
    2.1 Autonomous Driving Systems 7
    2.2 Video Object Detection 12
    2.3 General Purpose Perceiver Model 17
3. Methods 20
    3.1 Model 20
    3.2 Dataset 23
    3.3 Dropout and Shuffle 26
    3.4 Loss Function 26
    3.5 Metrics 30
4. Experiments 32
    4.1 Training for Bounding Boxes Prediction Task 32
    4.2 Training for Center Points Prediction Task 32
    4.3 Comparison Analysis 32
    4.4 Ablation Study 36
5. Discussion 40
6. Conclusion 42
References 43
7. Acknowledgements 49
Appendices 50
License 55

1. Introduction

Autonomous Driving Systems (ADS) promise safer roads, better traffic flow, and reduced environmental impact [2]. However, achieving this requires ADS to operate reliably under diverse and challenging conditions. Their perception stacks, reliant on multi-sensor data, are vulnerable to hardware and software failures like complete sensor failure or non-deterministic input availability, the two failure modes specifically investigated in this thesis. Such failures can lead to catastrophic accidents, undermining public trust [3]. Therefore, ADS need perception models that support multi-modal hardware and are robust to the failures.

Video object detection is a fundamental perception task that has applications in ADS. While video data affords rich spatio-temporal information compared to still images, it also introduces complexities. Initial approaches to the task frequently concentrated on postprocessing the outputs of still-image detectors. These methods employed techniques like non-maximum suppression Seq-NMS [4] to link detections or utilized tracking and optical flow [5, 6] to propagate information for temporal consistency. However, the core still-image detectors of these techniques does not learn from temporal data, and the methods are not end-to-end. This limitation restricts recovery from occlusions or substantial appearance shifts.

To mitigate these shortcomings, subsequent research integrated temporal modeling using Recurrent Neural Networks (RNN) and variants. Examples include Association LSTM [7] associates object features between frames, and STMN [8] with its Convolutional Gated Recurrent Unit (ConvGRU) for processing spatio-temporal information from feature maps. Alternative strategies focused on feature filtering, including techniques like attention mechanisms [9]. PSLA [10] weighs feature importance across frames and establishes correspondence without computationally expensive optical flow, by using a special attention approach called progressive sparse stride.

More recently, Transformer-based architectures [11] have emerged as leading models on the ImageNet VOD benchmark [12]. Models such as DETR [13] and Deformable DETR [14] reformulated the object detection task as a direct set prediction problem. These concepts were extended to the video domain through architectures like TransVOD [15] and PTSEFormer [16], which interpret complex spatio-temporal relationships, capture multi-scale features, and learn long-range dependencies.

Despite these advances, a gap remains in developing generalist perception architectures for ADS. Many state-of-the-art models are specialists, with strong inductive biases for specific data types. This limits their flexibility for the diverse sensor modalities (LiDAR, radar, thermal, video) in ADS. They often lack native mechanisms for fusing heterogeneous, high-dimensional inputs without complex, multi-component systems. This highlights the need for general-purpose architectures capable of processing diverse data types without domain-specific assumptions and scaling efficiently, enabling more versatile and robust ADS perception.

To address these limitations, this thesis introduces novel recurrent architectures, the Recurrent Perceiver (RPerceiver) and its multi-modal variant, Recurrent Perceiver Multi-Modal (RPerceiverMM). These models are designed as highly generalist recurrent modules, inspired by the Perceiver architecture [17], capable of processing diverse and high-dimensional sequential inputs, including multi-sensor data streams. We present a comprehensive training framework and experimental setup specifically designed to assess not only the object detection performance but also the robustness of these architectures against simulated sensor failure scenarios, demonstrating their potential for safety-critical applications like ADS. Main contributions are as follows:

- A Recurrent Perceiver architecture is proposed that, when unrolled in time, can be interpreted as RNN. Furthermore, the model supports multiple sensory inputs.
- A novel benchmark dataset, ”detection-moving-mnist-easy”, is introduced, designed to evaluate performance on two distinct tasks: bounding box detection and object center point (keypoint) prediction.
- Specific training procedures (e.g. input dropout, shuffling) and evaluation protocols designed to simulate and assess model robustness against potential hardware and software failures, such as complete sensor outages or non-deterministic input availability in single-sensor and multi-sensor setups, are proposed.

This thesis is organized as follows: section 2 provides background information on autonomous driving system architecture, overviews state of the art video object detection approaches, and introduces the original Perceiver model architecture [17]. Section 3 presents the novel Recurrent Perceiver architecture, introduces the benchmark used for training and testing, and explains the training procedure. Section 4 explains the experiments and presents the results.

2. Background

This section establishes the thesis's foundation by reviewing essential concepts, beginning with Autonomous Driving Systems (ADS). It utilizes the SAE J3016 standard [1] to define ADS, its automation levels, and critical safety requirements such as Dynamic Driving Task (DDT) performance during failures and DDT Fallback to a minimal risk condition, while also covering challenges like sensor failures and non-deterministic software, and their mitigation strategies. The section then examines Video Object Detection (VOD), a crucial perception task for ADS, outlining its evolution from basic frame-by-frame detection to sophisticated techniques leveraging temporal data, including post-processing, motion-information-based approaches (e.g. RNNs, LSTMs), feature filtering (e.g. attention mechanisms), and top-performing Transformer-based models. Finally, the General Purpose Perceiver model is introduced, detailing its architecture and novel cross-attention mechanism that allows it to process diverse, high-dimensional inputs by first projecting them into a smaller latent space, thereby addressing traditional Transformer scaling issues, paving the way for the thesis's proposed architectural modifications for enhanced sequential data processing relevant to these domains.

2.1 Autonomous Driving Systems

The Society of Automotive Engineers (SAE) International's J3016 standard [1] establishes a comprehensive framework for Autonomous Driving Systems. The standard defines an Autonomous Driving System (ADS) as the hardware and software collectively capable of performing the entire Dynamic Driving Task (DDT) on a sustained basis. The DDT encompasses all real-time operational and tactical functions requisite for on-road vehicle operation. It is important to note that SAE specifically employs the term 'ADS' to describe automation systems at Level 3, Level 4, or Level 5 of driving automation 1.

SAE J3016 [1] stipulates stringent safety requirements for ADS, particularly concerning their operational behavior during DDT performance-relevant system failures and the subsequent execution of the DDT Fallback. DDT Fallback refers to the response by the user or the ADS to either perform the DDT or achieve a minimal risk condition following such a system failure. A minimal risk condition, as defined by the standard, is a stable, stopped condition to which a user or an ADS may bring a vehicle after performing the DDT fallback in order to reduce the risk of a crash when a given trip cannot or should not be continued. The standard defines these safety expectations, including the management of DDT Fallback and the achievement of a minimal risk condition, differently for each automation level.

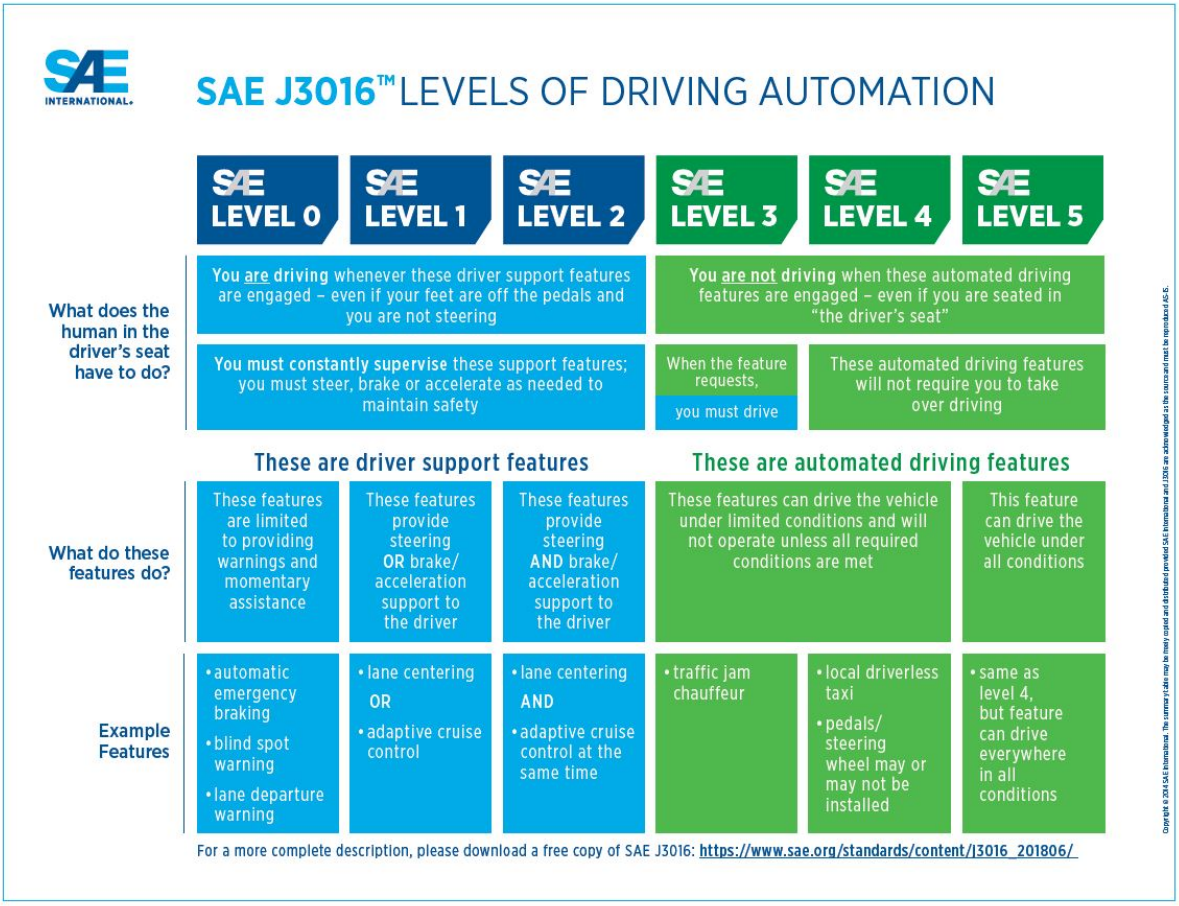

Figure 1. Levels of Driving Automation. This figure illustrates the six levels of driving automation as defined by the SAE J3016 standard, ranging from no automation (Level 0) to full vehicle autonomy (Level 5). The graphic clarifies the role of the human driver at each level and provides examples of corresponding driver support and automated driving features [1].

For Level 3 Conditional Driving Automation, a key mandate of SAE J3016 [1] is that in the event of a system failure, the ADS, after issuing an intervention request to the designated fallback-ready user, is obligated to continue performing the DDT for a duration of several seconds. This capability to maintain DDT performance for a critical period after the request underscores the system's operational capacity under failure and provides the necessary transition time. This temporal window is intended to afford the fallback-ready user, who is expected to be vigilant and prepared, sufficient time to assess the situation and either resume vehicle control or transition the vehicle to a minimal risk condition, should they ascertain its necessity. A presumption within the standard is that the fallback-ready user will be receptive to such intervention requests and to conspicuous system failures.

In contrast, Level 4 High and Level 5 Full Driving Automation are subject to more stringent safety requirements, a core aspect of which is their mandatory capability to perform the complete DDT Fallback under a performance-related system failure. These advanced systems must autonomously execute this fallback procedure and attain a minimal risk condition independently of human intervention. Consequently, Level 4 and 5 systems are designed to autonomously address failures and ensure vehicle safety, for example, by maneuvering to a roadside stop.

A foundational principle articulated by SAE for ADS pertains to their inherent operational resiliency—specifically, their capacity to maintain safe operations amidst system failures. At Level 3, this resiliency manifests through an orchestrated human-machine interaction, which ensures the user is afforded adequate time to resume control. Conversely, for Levels 4 and 5, resiliency implies that the ADS autonomously executes the fallback procedure and transitions the vehicle to a minimal risk condition. These safety-critical functionalities are deemed indispensable for fostering public trust and facilitating the extensive and safe deployment of autonomous driving technologies.

The operational scope of an Autonomous Driving System (ADS) is defined by its Operational Design Domain (ODD). The ODD specifies the precise conditions under which a given ADS is designed to function reliably. These conditions encompass a wide range of factors, including, but not limited to, roadway types, geographical constraints, speed limitations, and environmental conditions [1].

Broadly, ADS architectures are categorized into two principal approaches: modular and end-to-end [18], both architectures are shown in the Figure 2. The modular approach, also termed the mediated perception approach and considered conventional, organizes the driving task into

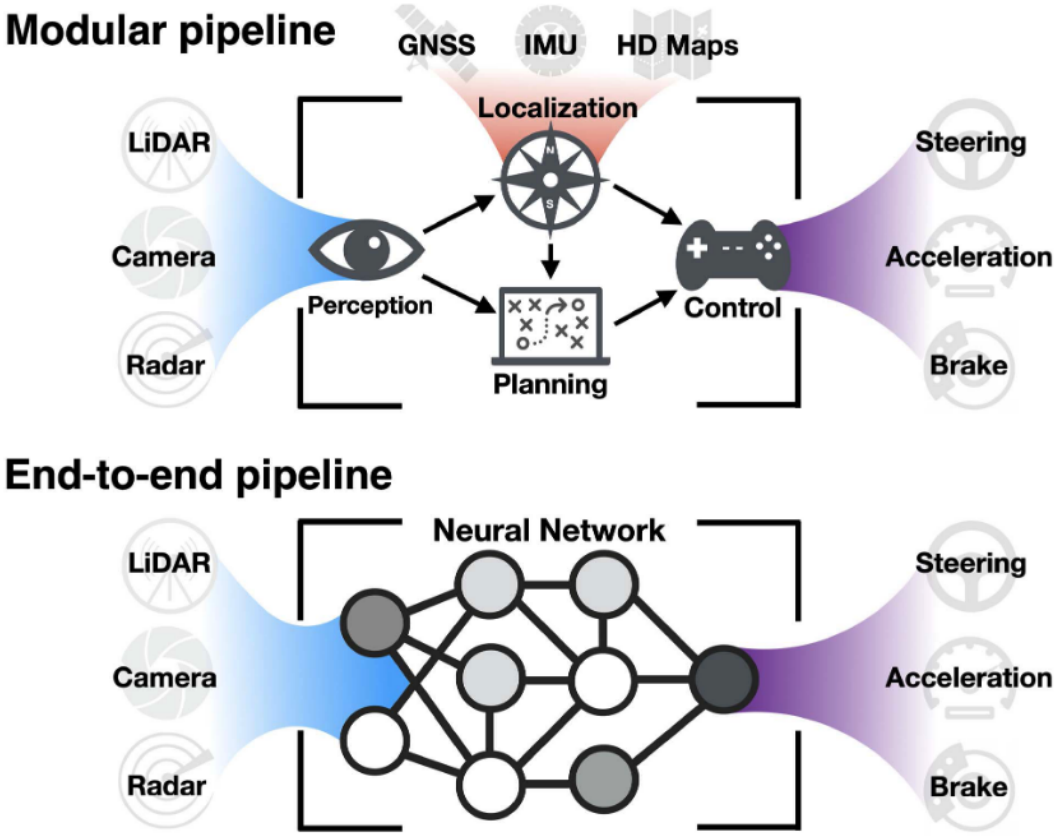

Figure 2. Modular and end-to-end pipelines. The modular pipeline for autonomous driving consists of many interconnected modules, whereas the end-to-end approach treats the entire pipeline as one learnable machine learning task [18].

a series of interconnected, self-contained modules such as perception, localization, planning, and control [18]. Conversely, end-to-end driving, sometimes referred to as behavior reflex, conceptualizes the entire pipeline—from processing sensory inputs to generating vehicle control commands (e.g., steering, acceleration)—as a singular, learnable machine learning task, often actualized through a unified neural network [18]. Irrespective of the architectural approach, ADSs are equipped with a comprehensive suite of sensors to construct a detailed model of their environment. This sensor suite commonly includes cameras, LiDAR (light detection and ranging), radar, and ultrasonic sensors [19]. Recognizing that sensors vary in technology and purpose, each with inherent weaknesses, necessitates sensor fusion—the combination of inputs from multiple sensors—to overcome these individual limitations, reduce errors, and thereby achieve a consistent, accurate, and robust perception of the environment [19].

Given the reliance on a complex sensor suite and operation in diverse environments, the potential for sensor failures presents a significant challenge in ADS design and validation. Sensor failures can manifest in various forms, from partial degradation of performance under adverse environmental conditions—such as cameras being affected by glare or low light, or LiDAR and RADAR performance being impacted by heavy precipitation—to the complete malfunction of a sensor unit [19]. While a core design objective is to prevent catastrophic system failure, the occurrence of individual sensor faults or the degradation of a specific sensing modality is a credible event that the ADS must be engineered to handle gracefully. Key mitigation strategies include advanced sensor fusion algorithms, which intelligently combine data from disparate sensor sources to enhance accuracy and compensate for the limitations or failure of any single sensor. In instances of complete failure of a critical sensor or an entire sensor modality, redundancy is the primary mitigation strategy.

The ADS's computational hardware and software stack processes data streams generated by its diverse sensor array. Integrating and fusing this data presents a fundamental challenge due to different sensors inherently operating at varying sampling rates, possessing different internal processing latencies, and being subject to communication delays. Temporal calibration, which involves accurately timestamping and synchronizing these heterogeneous data streams, is a mitigation strategy that can address aspects of this non-deterministic behavior to some extent [19]. The software systems integral to ADSs, often employing middleware like the Robot Operating System (ROS) or its real-time focused iteration ROS 2.0 for inter-process communication and data synchronization. Traditional middleware such as ROS 1.0 was not originally architected for stringent real-time performance, and its underlying mechanisms can introduce variability in message latencies and task execution times [20]. While newer frameworks like ROS 2.0 aim to enhance real-time performance and determinism using technologies such as the Data Distribution Service (DDS) [20], achieving complete elimination of non-deterministic behavior in these large-scale distributed software systems remains exceptionally challenging. Therefore, despite mitigation strategies like temporal calibration and advancements in middleware, non-deterministic software behavior remains an issue.

The Dynamic Driving Task (DDT), as defined by SAE J3016, encompasses all real-time operational and tactical functions required for on-road vehicle operation [1]. These functions include lateral and longitudinal vehicle motion control, as well as monitoring the driving environment. A critical set of subtasks of the DDT is Object and Event Detection and Response (OEDR). OEDR involves the subtasks of monitoring the driving environment (detecting,

recognizing, and classifying objects and events, and preparing to respond as needed) and executing an appropriate response to complete the DDT [1]. It is the system's capability to perform the entire DDT, including OEDR, that distinguishes an ADS (Levels 3-5) from lower-level driving automation system. The perception aspect of OEDR is also known as perception and a variety of computer vision tasks fall under this category, such as object detection, semantic segmentation, 3D object detection, and others [3].

To fulfill the stringent safety requirements articulated for Autonomous Driving Systems, notably by standards such as SAE J3016 [1], it is imperative that ADSs are designed and validated to be resilient against the aforementioned failures, including complete sensor failure and the impacts of non-deterministic software behavior. This thesis contributes to addressing these challenges by proposing specific training procedures and evaluation protocols. These are designed to simulate and rigorously assess perception model robustness and its potential for ADS.

2.2 Video Object Detection

Object detection is a foundational challenge in computer vision and has been a subject of research for several decades [21]. The goal of the object detection task is to find objects of a given description in images and videos. Advancements in deep learning techniques for feature representation learning [22, 23], in conjunction with the significant development and application of deep Convolutional Neural Networks (CNN), have driven remarkable progress in various computer vision tasks, such as image classification [24], object detection [25]. Extending detection capabilities from static images to video sequences introduces the task of video object detection, which involves not only localizing objects within each frame but also leveraging temporal information across frames for improved accuracy and consistency. The introduction of specific challenges, such as the video object detection track in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [12], provided benchmark datasets and standardized evaluation protocols, significantly accelerating research and development in the video object detection domain.

Due to the inherent similarity between detecting objects in single images and in video frames, the most straightforward approach to video object detection task is to apply a single-image object detector independently to each frame. This method, often referred to as frame-by-frame detection, treats each frame as independent image. However, such an approach ignores the rich temporal and contextual information available across consecutive video frames. Neglecting this temporal dimension often leads to suboptimal performance, characterized by issues like inconsistent

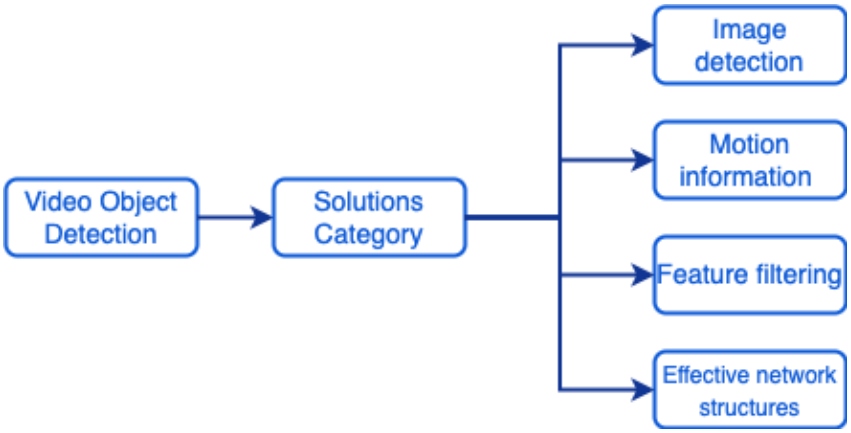

Figure 3. Classification of video object detection solutions.

bounding box predictions across frames, flickering detections, and reduced robustness to challenges specific to video, such as motion blur, occlusion, morphological diversity, and illumination variations within the video [26]. Consequently, while applying static detectors frame-by-frame serves as a simple baseline, it is generally not considered an effective or optimal solution for the complexities of video object detection.

Video object detection algorithms can be broadly classified based on their architectural solution to the video object detection task. For this thesis, the classification proposed by the survey [26] is adopted. This survey categorizes these algorithms into four main types: those based on image detection (postprocessing methods), those utilizing motion information (introducing additional models), those employing feature filtering, and other effective network structures.

One strategy involves applying a postprocessing step to the outputs of a still-image detector to enhance temporal consistency [4–6]. This often utilizes detections from adjacent frames to refine the results for the current frame. Common postprocessing techniques include Sequence Non-Maximum Suppression (Seq-NMS), which links high-scoring detection boxes across frames into sequences [4], or leveraging tracking and optical flow to propagate detection scores [5, 6]. While these methods can improve upon static image detectors, exploiting temporal information solely during postprocessing is considered suboptimal, as crucial temporal and motion cues are disregarded during the primary detector training phase. Therefore, such algorithms have difficulty overcoming consecutive failures of the static image detector when the object of interest experiences long-term occlusions or significant appearance changes.

To address the limitation of postprocessing, another category of methods utilizing motion information by introducing additional models to integrate motion and temporal information

directly into the model training, often creating end-to-end solutions. This category is subdivided into models based on optical flow, contextual information, and trajectory information. For the purpose of this thesis, a subcategory of contextual information is reviewed, which utilizes Recurrent Neural Networks (RNN) and their variants. The power of RNNs in long-range temporal representation has become a valuable tool for the video object detection task [7, 8, 27], enabling the design of end-to-end networks. One of the challenges posed by such methods is how to associate objects within the RNN structure across multiple frames.

One notable early attempt to tackle this association problem is the Association LSTM framework [7]. The framework consists of an SSD image detector [28] and a variant of the RNN model, the Long Short-Term Memory (LSTM) network [29]. SSD detects frame-wise, image-based object detection results (bounding box, score, and object feature) which are stacked and fed into the LSTM. The association LSTM not only regresses and classifies directly on object locations and categories but also associates features which represent each output object. By minimizing the matching error between these features, the network learns how to associate objects in two consecutive frames. Additionally, the method works in an online manner, which is important for real-world applications. The authors acknowledge that a weakness of these frameworks is that the LSTM module is a post-hoc addition, since performance is limited by the quality of the initial SSD detections. Missed or poorly localized detections by the SSD are difficult for the LSTM to recover. Furthermore, the SSD parameters are not updated during training.

Building upon the idea of modeling temporal dependencies but aiming for deeper integration and leveraging spatial information more effectively, the Spatial-Temporal Memory Network (STMN) was proposed [8]. Unlike the Association LSTM which operates on vector-form features within a standard LSTM, STMN introduces a Spatial-Temporal Memory Module (STMM). This module utilizes a modified bidirectional convolutional Gated Recurrent Unit (ConvGRU) [30], allowing it to process and retain information in a spatially structured manner directly from convolutional feature maps generated by the detector backbone. By preserving spatial locality within the recurrent computation, STMM can better capture appearance changes and motion patterns. Experimental results indicated that the effectiveness of such convolutional memory modules increases with the length of the input sequence, as more relevant long-range context can be accumulated, leading to improved detection accuracy, particularly for challenging scenarios involving occlusion or significant appearance variations [8].

The principle of using convolutional recurrent units to efficiently propagate spatio-temporal information proved beneficial not only for accuracy but also for enabling real-time video object detection on resource-constrained platforms. In this work [27], an architecture is introduced that augments a single-image object detector (SSD with a pruned MobileNet [31] base) by interweaving efficient convolutional LSTM (ConvLSTM) [32] layers, specifically ”Bottleneck-LSTM” layers, to refine and propagate feature maps across frames. Despite its more complex architecture, an array of modifications is proposed that allow this model to be faster and more lightweight than mobile-focused single-frame models.

Another category of video object detection methods employs feature filtering techniques to enhance performance by selectively focusing on relevant spatiotemporal information while suppressing redundant or irrelevant data. This approach draws inspiration from the human visual system, which can rapidly identify salient regions within a scene, thereby optimizing cognitive resources for efficient analysis [33]. Similarly, feature filtering mechanisms in neural networks aim to prioritize critical features and reduce unnecessary computations, leading to improvements in both accuracy and efficiency [26]. These methods can be broadly subdivided based on the specific filtering mechanism employed, notably attention mechanisms [9, 11] and deformable convolutions [34].

Attention mechanisms allow networks to dynamically weigh the importance of different features or spatial locations across video frames. This selective focus enables the propagation of crucial information, particularly for objects undergoing significant appearance changes or movements, potentially offering advantages over methods relying solely on adjacent frame correlations or optical flow, which can struggle with large displacements and add significant model complexity [10, 26].

One prominent example is the Progressive Sparse Local Attention (PSLA) framework [10]. Addressing limitations associated with optical flow, such as increased model size and difficulty handling large displacements or high-level features, PSLA provides an alternative mechanism for establishing feature correspondence and propagating information between frames [10]. Instead of calculating pixel-level flow, PSLA operates directly on feature maps. While similar to STMN [8] which also uses local correlation for alignment, PSLA differs by utilizing a sparse neighborhood and softmax normalization for better spatial correspondence, aiming to improve both speed and accuracy [10]. It uses a special attention approach called progressive sparse stride, which pays more attention to nearby features (for small movements) and less attention to features

farther away (for larger movements). By calculating weighted correspondences based on feature similarity within this sparse local neighborhood, PSLA aligns and aggregates features across time, enabling temporal feature updating and enhancement without relying on an explicit optical flow model. This approach aims to achieve a better balance between accuracy, speed, and model size compared to traditional flow-based methods [10]. Nevertheless, managing the complexity of attention calculations remains a factor.

The final category outlined by [26], termed ”Other Effective Network Structures”, incorporates a range of exploratory and innovative methodologies for video object detection. While Transformer-based approaches were not explicitly itemized under this classification in the survey [26], the author of this thesis has taken the considered step of including them within this category. This decision is predicated on an analysis of their architectural principles and their significant subsequent impact on the field, which aligns with the innovative spirit of the ”Other Effective Network Structures” designation. Prominent among these emergent architectures are those predicated on the Transformer model [11]. Initially transformative within the domain of natural language processing, the Transformer architecture has subsequently exhibited considerable promise for applications in computer vision, most notably in the task of object detection [13].

Pioneering works like Detection Transformer (DETR) [13] reformulated object detection as a direct set prediction problem. To address some of the initial limitations of DETR, such as slow convergence and high computational cost, Deformable DETR [14] was introduced. It incorporates a deformable attention module that only attends to a small set of key sampling points around a reference, significantly improving efficiency and performance. However, the encoder in early DETR variants was identified as a computational bottleneck, a point highlighted in subsequent research [35].

Building on these foundational image-based object detection Transformers, researchers began adapting them for the complexities of video. Subsequent adaptations for the video domain have demonstrated considerable success, with DETR-based methodologies achieving high performance. A notable example specifically designed for video is TransVOD [15]. TransVOD constructs an end-to-end Video Object Detection framework utilizing a spatial-temporal Transformer. This architecture is designed to effectively detect objects and maintain their identities by modeling relationships across frames, thus tracking how objects move and change throughout a video sequence. Further advancements in this domain include models like

PTSEFormer [16], which introduces a novel pyramidal temporal-spatial encoder. This encoder aims to efficiently capture multi-scale spatio-temporal features by progressively integrating information from different temporal and spatial resolutions, enhancing the model's ability to handle variations in object scale and motion.

The impact of these Transformer-based models is underscored by their strong performance on established benchmarks. Indeed, Transformer-based methods are among the top-scoring approaches in challenges such as the ImageNet VID (Video Object Detection) task [12]. Their inherent ability to model long-range dependencies and learn complex spatio-temporal relationships positions them as a critical area of ongoing research and development in the field of video object detection.

2.3 General Purpose Perceiver Model

Most architectures used by AI systems today are specialized. For instance, models presented in the section 2.2 built for the video object detection task might excel at processing 2D video frames, but they are hardly ideal for other data types, such as the LiDAR point clouds or radar output used in ADS. Handling multiple data modalities, like the sounds and images that make up videos, presents even greater complexity and usually involves complex, hand-tuned systems built from many different parts, even for simple tasks. Real-world problems, such as building an ADS, possess these complexities, so there is a desperate need to build a simple yet effective, more general, and versatile architecture that can handle all types of data.

Perceiver [17] was introduced as a general-purpose architecture that is capable of processing different data types such as images, point clouds, audio, video, and, what is important, combinations of those to fuse together. The Perceiver builds upon the Transformer [11], an architecture that uses an operation called ”Attention” to map inputs into outputs [9]. While attention is simple and widely applicable, the way Transformers utilize attention can become memory expensive as the number of inputs grows. Consequently, Transformers perform well with inputs containing at most a few thousand elements, but common data forms like images and videos can easily comprise millions of elements. This fact poses a challenge to the generalist architecture: scaling the Transformer's attention operation to very large inputs without introducing domain-specific assumptions. The Perceiver addresses this by using attention to first encode the inputs into a small latent array. This latent array can then be processed further at a cost independent of the input's size, allowing the Perceiver's memory and computational needs to scale gracefully as the input size grows, even for particularly deep models. This ”graceful growth” enables the

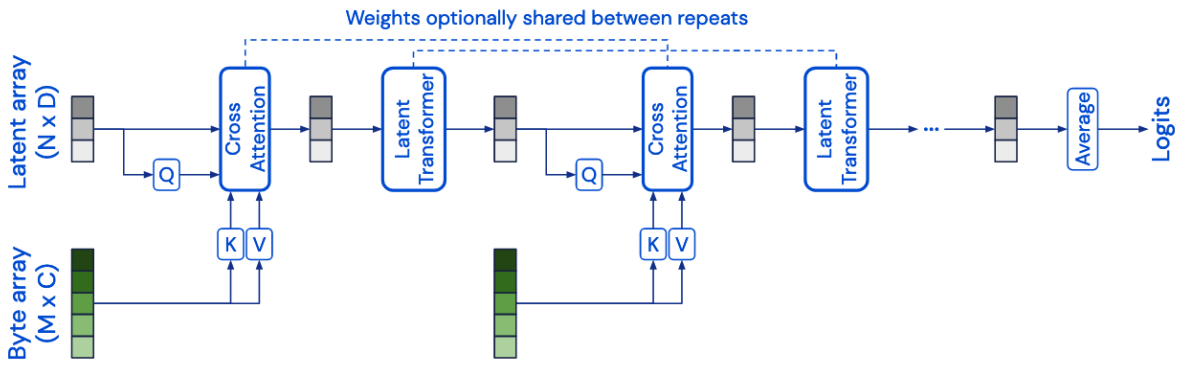

Figure 4. The Perceiver is an architecture based on attentional principles that scales to high-dimensional inputs such as images, videos, audio, point-clouds, and multimodal combinations without making domain-specific assumptions. The Perceiver uses a cross-attention module to project an high-dimensional input byte array to a fixed-dimensional latent bottleneck (the number of input indices M is much larger than the number of latent indices N) before processing it using a deep stack of Transformer-style self-attention blocks in the latent space. The Perceiver iteratively attends to the input byte array by alternating cross-attention and latent self-attention blocks [17].

Perceiver to achieve an unprecedented level of generality - it is competitive with domain-specific models on benchmarks based on images, 3D point clouds, and combined audio and images [17].

The Perceiver architecture consists of two primary components: (i) a cross-attention module that maps an input byte array (e.g. a pixel array) and a latent array to an updated latent array, and (ii) a Transformer tower that maps this latent array to another latent array [17]. The Perceiver architecture is illustrated in Figure 4. The model can be conceptualized as a recurrent neural network (RNN) unrolled in depth with the same input, rather than unrolled in time. A central challenge addressed by this architecture is the scaling of attention mechanisms to accommodate very large and diverse inputs. The Perceiver mitigates the quadratic complexity bottleneck inherent in standard Transformers by employing its cross-attention module. This module introduces an asymmetry into the attention operation when applied directly to the inputs.

Specifically, for a query $Q \in \mathbb{R}^{M \times D}$, key $K \in \mathbb{R}^{M \times C}$, and value $V \in \mathbb{R}^{M \times C}$ (where C and D represent channel dimensions), the computational complexity of the standard QKV attention operation—formulated as softmax$(QK^T)V$—is $\mathcal{O}(M^2)$. This is due to matrix multiplications involving the large input index dimension M. The authors of Perceiver introduced an asymmetry: while K and V are projections of the input byte array (with M elements), Q is a projection

of a learned latent array characterized by a much smaller index dimension $N \ll M$. The dimension N of this latent array is a hyperparameter. Consequently, the resulting cross-attention operation exhibits a complexity of $\mathcal{O}(MN)$. It is important to note that within the Perceiver's cross-attention, linear projection layers are applied to generate Q, K, and V such that they share a common channel dimension before the attention calculation.

This design enables Perceiver-based architectures to leverage significantly deeper Transformers compared to efficient Transformer variants that use linear complexity layers, without depending on domain-specific assumptions. Using L to represent the number of layers in the Transformer tower, the complexity of a standard Transformer operating directly on M input elements (bytes) can be represented as $\mathcal{O}(LM^2)$, whereas the complexity of Perceiver's latent Transformer (processing the N-dimensional latent array) is $\mathcal{O}(LN^2)$. Given that $N \ll M$, this substantially reduces the computational cost per layer. The overall architecture's complexity thus becomes $\mathcal{O}(MN + LN^2)$ for a latent Transformer with L layers. This decoupling of the input size from the network depth is crucial, as it permits the addition of Transformer layers at a cost that is independent of the input size.

However, because the original Perceiver produced only one output per input, it was not as versatile as researchers required. Our proposed architecture aims to address this limitation by modifying the Perceiver architecture to make it akin to a recurrent unit processing input over time.

3. Methods

This section details the methodologies employed in the current research. It starts with an introduction to novel recurrent perceiver architectures—specifically the Recurrent Perceiver (RPerceiver) and its multi-modal extension, the RPerceiverMM—which are engineered to process high-dimensional temporal data, such as video sequences, for object detection tasks. The subsequent section discusses the custom-generated ”detection-moving-mnist-easy” dataset, developed to facilitate the evaluation of model capabilities in video object detection and center point localization. Following this, training procedures are described, including dropout and shuffle techniques, implemented to enhance model robustness against sensor unreliability and non-deterministic input sequences. The section then elaborates on the adopted loss function for the set prediction problem. Finally, performance evaluation metrics are outlined, including Mean Average Precision (mAP) for object detection, and Average Displacement Error (ADE) and Final Displacement Error (FDE) for the center point prediction task.

3.1 Model

A new Recurrent Perceiver (RPerceiver) is introduced, a Recurrent Neural Network (RNN) capable of processing high-dimensional inputs. This architecture draws inspiration from the Perceiver [17], noted for its ability to handle high-dimensional data.

The Perceiver architecture has been re-engineered by incorporating a temporal dimension, effectively unrolling it over time. This modification addresses the original Perceiver’s limitation of producing only a single output per input, a characteristic that renders it unsuitable for sequence tasks with a temporal component, such as video object detection. Whereas previously the Perceiver was only unrolled in depth, the loop has now been closed, and it is unrolled in time by propagating the latent array between time steps. The architecture of the RPerceiver is illustrated in Figure 5.

A variation of the RPerceiver architecture capable of processing multi-modal inputs is proposed, termed the Recurrent Perceiver Multi-Modal (RPerceiverMM). The original Perceiver paper [17] addressed multi-modality by concatenating a learned, modality-specific encoding to each input element. Modern Autonomous Driving Systems (ADS) process information from multiple sensors, often incorporating multiple cameras positioned in different locations, which introduces a calibration challenge. Consequently, this work adopts a distinct approach to multi-modality. A sensor-specific cross-attention module is introduced to the RPerceiverMM. Within the scope of

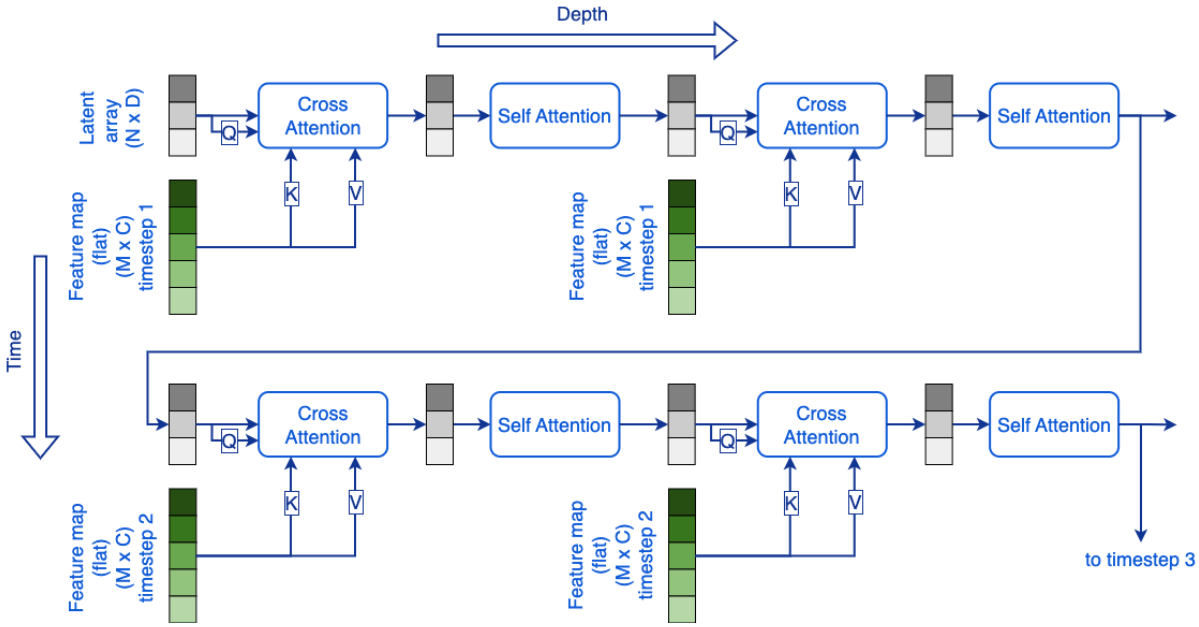

Figure 5. The Recurrent Perceiver (RPerceiver) architecture processes inputs along the time dimension by propagating the latent array between time steps. In the depth dimension (within a single time step), this example shows two RPerceiver layers.

this thesis, the focus is on a multi-view camera setup. The architecture of the RPerceiverMM is illustrated in Figure 6.

In addition to the RPerceiver module, a CNN backbone and detection heads were utilized. This backbone served to learn a $2D$ representation of the input frames. This backbone was designed with four blocks, each containing convolutional layers and ReLU activation. Each block progressively downscales spatial dimensions by a factor of 2 while concurrently increasing channel dimensionality. This process results in a final feature map possessing 128 channels and $\frac{1}{16}$ of the original spatial resolution.

The output features generated by the RPerceiver module are subsequently passed to the detection heads, which are tasked with predicting object labels and their corresponding positions. Detection heads identical to those employed in the DETR model [13] were adopted. These heads incorporate a linear layer to predict the class label via a softmax function, and a 3-layer Multi-Layer Perceptron (MLP) featuring ReLU activation functions to predict object coordinates. Two variants for object coordinates were considered: (i) bounding boxes, wherein the MLP predicts the normalized center coordinates, height, and width relative to the input image, and (ii) center points. The respective output dimensions for these variants are $N \times 4$ and $N \times 2$, where N denotes a fixed number of detection slots (queries). Given that N is typically substantially larger

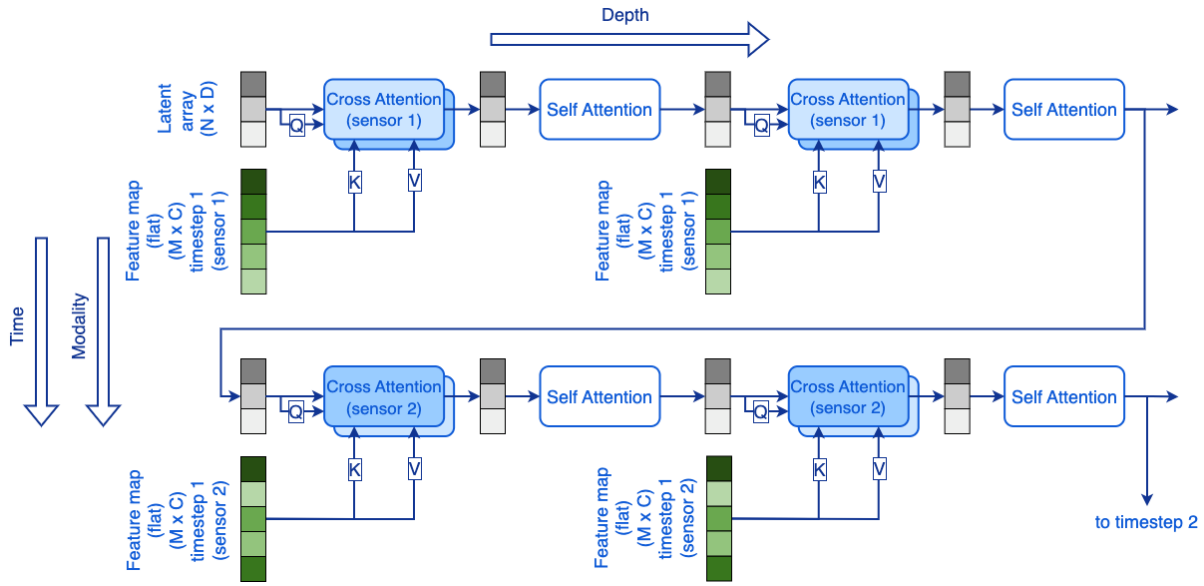

Figure 6. The Recurrent Perceiver Multi-Modal (RPerceiverMM) architecture operates along time, depth, and an additional modality dimension, designed for multi-view camera inputs. It employs camera-specific cross-attentions (shown in different colors), which have their own weights, to process information from different sensors (views). In this example, separate cross-attention blocks are used for each camera input, integrating modality-specific information.

than the actual number of objects present, a special class label, $\emptyset$, is employed to signify that no object is detected within a particular slot, thereby fulfilling a role analogous to a background class.

For bounding box predictions, a sigmoid activation function was used to predict the normalized coordinates of the bounding box center, as well as its width and height. In the case of center points, a tanh activation function was utilized. This selection was based on positioning the origin of the coordinate system at the center of the image raster, thereby emulating a bird's-eye view perspective, akin to an Autonomous Driving System (ADS) positioned centrally with comprehensive surrounding views. The complete architecture of the model is shown in Figure 7.

The intuition behind the RPerceiver architecture for object detection is that the variable N, a dimension of the latent array, represents the number of objects the RPerceiver tracks, while D represents the number of attributes for each object (e.g., position, dimension, color, speed, etc.). In order to revalidate the existence of an object across different modalities by utilizing cross-attention to match its position, dimensions, and color with sensor input. Periodically, the system must initiate tracking for new objects within its N available slots, necessitating a mechanism to monitor slot utilization.

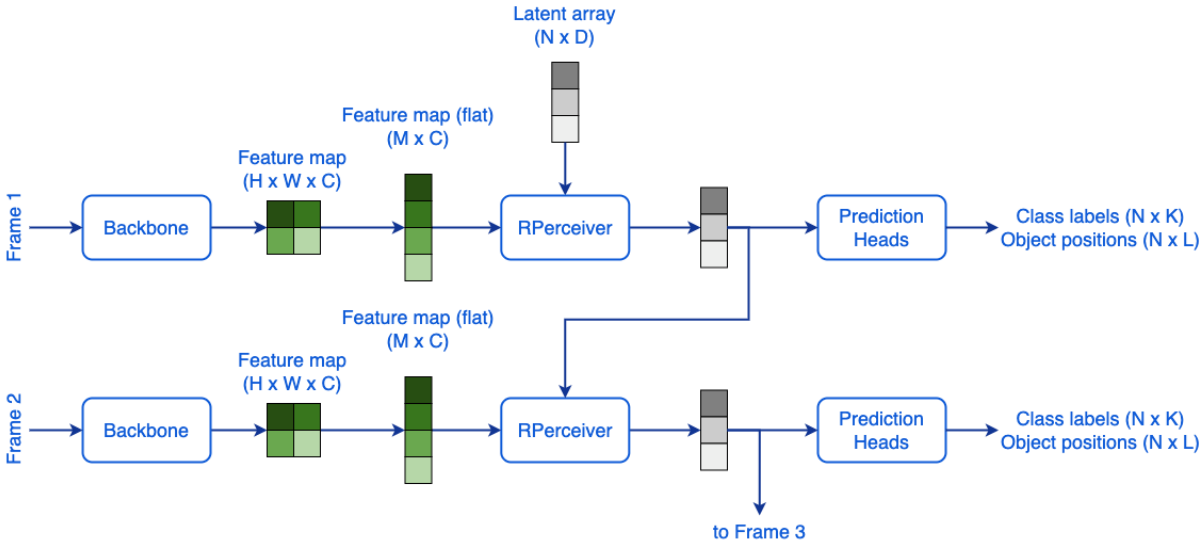

Figure 7. Full architecture of the model, including RPerceiver, Backbone, and Detection Heads. Input frames are processed by a Backbone network to extract a feature map. This flattened feature map, along with a latent array propagated from the previous time step, is fed into a Recurrent Perceiver (RPerceiver). The RPerceiver's output embedding is then used by Prediction Heads to determine class labels and object positions for the current frame.

3.2 Dataset

For this experiment, a custom dataset was generated, named "detection-moving-mnist-easy". The dataset design was inspired by the MovingMNIST dataset [36], which is commonly used for tasks like video prediction, unsupervised feature learning from video. In this case, the focus is on video object detection and a simplified variation of keypoints, where the prediction involves the center point of the object. This task is similar to keypoint detection as the object's center point, rather than the bounding box center, is predicted. The dataset is hosted on Hugging Face Hub [2].

For the first frame, a number of digits, ranging from 1 to 10, is selected with uniform probability (see Figure 8). Depending on the number of digits selected for the first frame, digits are drawn, without replacement, from the well-known MNIST dataset [37] (from the train and test splits corresponding to the dataset split). Each digit is placed on the first frame of the 128×128 canvas image. A greedy algorithm is employed to randomly place digits on the first frame while

[2] https://huggingface.co/datasets/Max-Ploter/detection-moving-mnist-easy

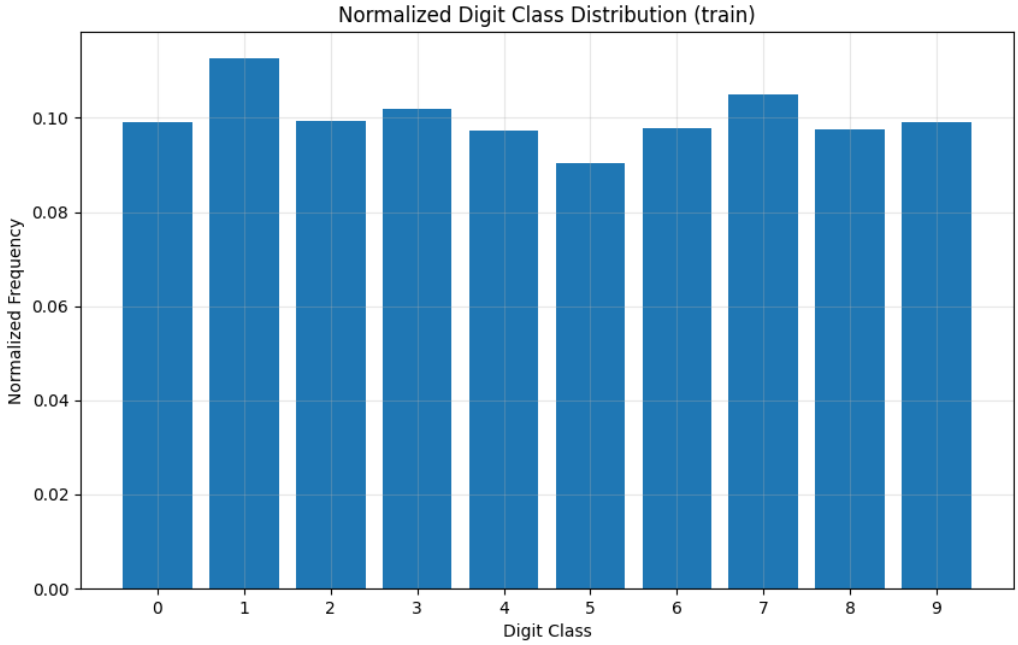

Figure 8. Distribution of classes in the ”detection-moving-mnist-easy” dataset.

attempting to avoid overlaps. This initial separation is intended to simplify object detection for the model at the start of the sequence. To each digit, an affine translation ranging from -5 to 5 pixels is randomly assigned with uniform probability. Then, corresponding affine transformations are applied to move the digits through 20 frames on the 128×128 canvas image. As a result, this process yields a tensor of size $20 \times 1 \times 128 \times 128$, representing a video sequence (see Figure 9).

To make the dataset more challenging, digit overlap is not restricted in subsequent frames. Some degree of overlap in the first frame is possible if the greedy algorithm is unable to randomly place all digits without overlap. Digits are not bounced against image boundaries, allowing each digit to potentially leave the frame. As illustrated in Figure 10, later frames may contain fewer digits.

The dataset was generated with $60,000$ training samples and $10,000$ testing samples. Annotations, automatically generated during sequence creation, include digit classes, digit center point coordinates (keypoints), bounding box coordinates, and digit identity IDs.

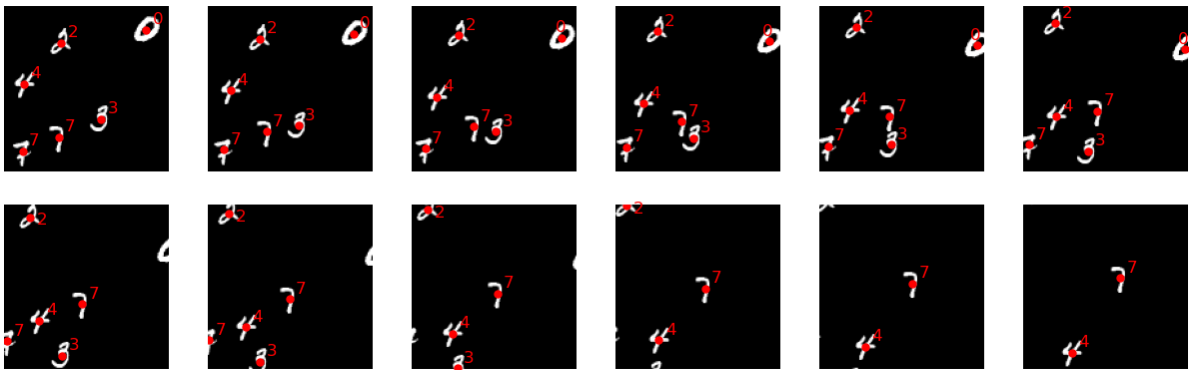

Figure 9. Example of 12 frames from the sequence. Ground truth, shown in red, indicates the ground truth digit center point and a class label.

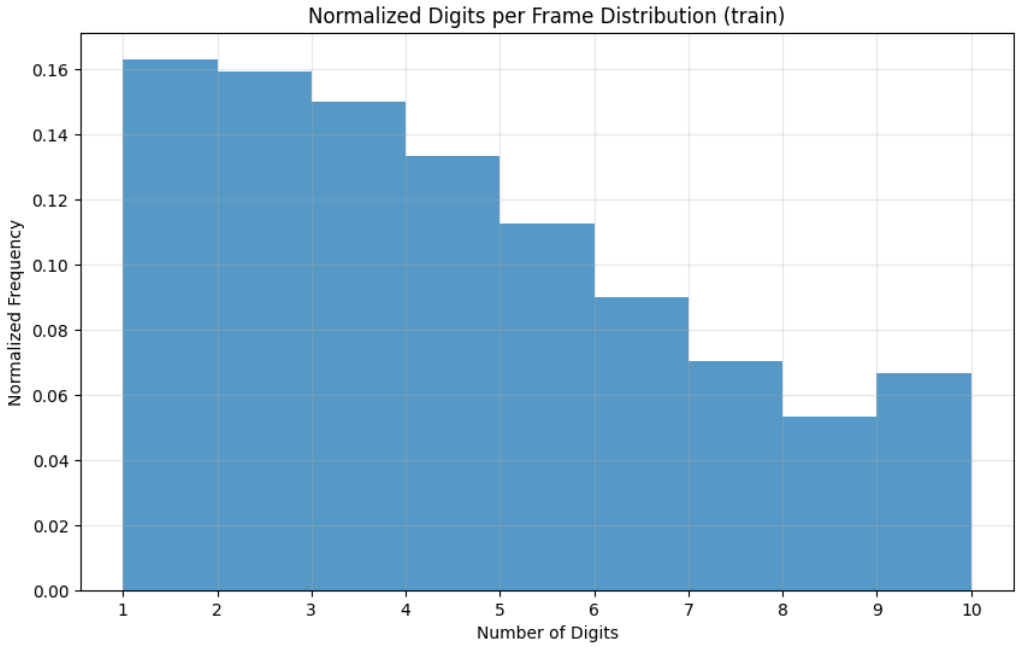

Figure 10. Normalized frequency distribution of the number of digits per frame for the **train split** of the ”detection-moving-mnist-easy” dataset. The histogram illustrates that frames containing a smaller number of digits (notably 1 to 3) are observed with the highest normalized frequency. This distribution, skewed towards fewer digits per frame, is a characteristic outcome of the dataset’s generation process where digits can move and exit the frame boundaries over time, as their movement is not confined within the frame.

3.3 Dropout and Shuffle

As previously discussed in Section 2.1, Autonomous Driving Systems (ADSs) face challenges related to sensor reliability. Two critical identified issues are the potential for complete sensor failure, where a sensor modality ceases to provide any data, and the non-deterministic availability of sensor inputs. To enhance the resilience of perception models against these real-world imperfections, this work investigates training strategies designed to improve model robustness. Specifically, the aim is to experimentally evaluate how models perform when subjected to simulated conditions mimicking both complete sensor data loss (akin to sensor failure) and non-deterministic input sequences (akin to middleware-induced timing variations). The following training procedures, `dropout` and `shuffle`, are introduced to directly address these challenges during the model training phase. A model trained without applying these training procedures is considered the baseline.

`shuffle` In this procedure, the sensor inputs are randomly permuted within each time step. Consequently, the model receives inputs from the sensors in a random order for that specific time step. This shuffling only occurs for sensor inputs within the same time step and aims to simulate the non-deterministic ordering of inputs that can occur due to middleware asynchronicity. This training procedure is therefore relevant for models that process data from multiple distinct sensors at each time step.

`dropout` This procedure simulates scenarios where sensor information is missing, directly addressing the challenge of potential complete sensor failure for one or more modalities. To achieve this, the model is trained by intermittently dropping sensor inputs (input dropout). The first half of the input sequence is kept intact (no dropout), allowing the model to accumulate features in its hidden state. The second half of the sequence may undergo frame dropout depending on the dropout probability. During training, the probability of an information dropout is gradually increased from 10% up to 86.6%.

The `dropout` procedure is shown in Figure 11. The pseudo code is shown in Appendix 7.

3.4 Loss Function

The loss calculation approach was adopted from [13, 38]. The proposed model in the Section 3.1 predicts a fixed-size set of N potential objects per timestep. The parameter N is a dimension of the latent array and is chosen to be greater than the cardinality of the largest set of ground-truth objects per frame. The model uses an attention mechanism that allows it to attend to all other

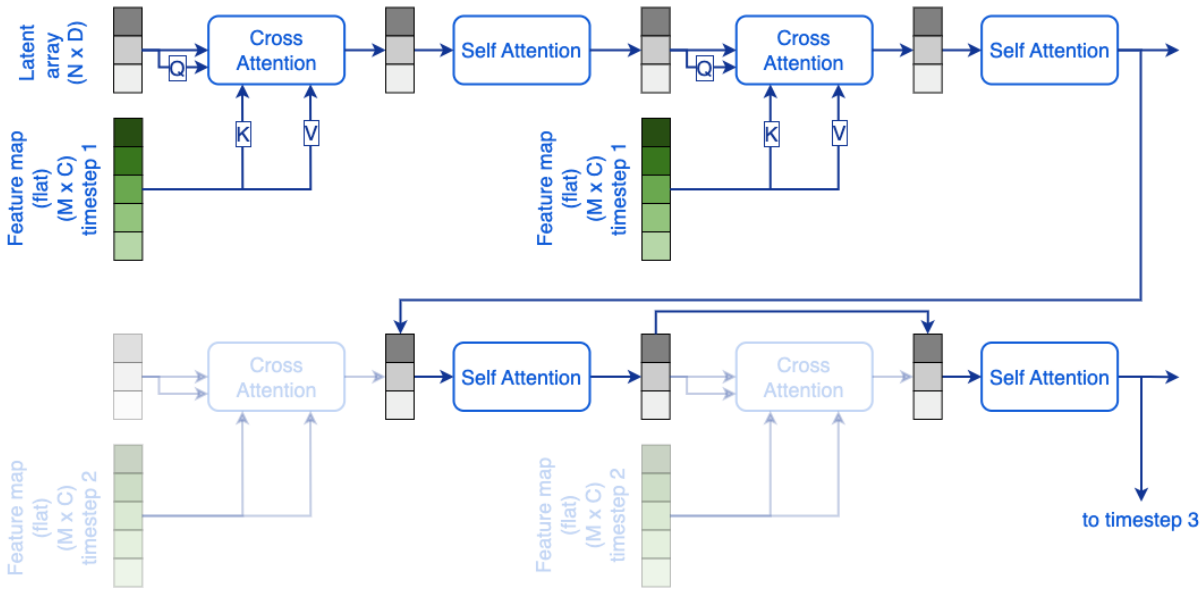

Figure 11. Diagram of the `dropout` mechanism in action. Timestep 1 shows normal operation where the input feature map is processed by the backbone (implied before cross-attention) and then fused with the latent array via cross-attention, followed by self-attention. Timestep 2 illustrates a dropout event: the input feature map and the cross-attention block are shown faded, indicating that this sensor view's data is dropped. Consequently, the cross-attention step, which incorporates new sensor information (K, V), is skipped. However, the latent array is still updated by the self-attention mechanism, demonstrating the model's reliance on its internal memory when an input is missing. Intuitively, this self-attention acts like a 'propagate forward' step in a tracker: it updates the object's presumed state based on prior motion even without current sensor input. This internal prediction maintains continuity and can be corrected by cross-attention when new sensor data becomes available and is processed.

elements in the attention map. Therefore, there is no predefined structure, like a grid in one-stage image detectors [39], associating predictions with ground-truth objects. This presents the core challenge of how to score the fixed-size prediction set against the variable-size ground-truth set. The loss function calculation consists of two steps: first, finding an optimal bipartite matching between the elements of the predicted and ground-truth sets, and second, calculating a loss.

Let us denote the set of predictions for a given timestep as $\hat{Y} = \{\hat{y}_j\}_{j=1}^{N}$ and the corresponding set of ground-truth objects for that frame as $Y = \{y_i\}_{i=1}^{M}$ where $N > M$. At each recurrence (i.e., for each frame/timestep processed), the RPerceiver outputs the entire set of N predictions $\hat{Y}$. Each prediction $\hat{y}_j \in \hat{Y}$ consists of a predicted object position $\hat{o}_j$ and a predicted class label $\hat{c}_j$ (including the possibility of the no-object class $\emptyset$). The position $\hat{o}_j$ can represent different objects depending on the specific task:

- For the **bounding box prediction task**, $\hat{o}_j$ is the predicted box $\hat{b}_j = (\hat{b}_x, \hat{b}_y, \hat{b}_w, \hat{b}_h) \in \mathbb{R}^4$, representing center coordinates, height, and width relative to the frame size.
- For the **center point prediction task**, $\hat{o}_j$ is the predicted center point coordinates $\hat{p}_j = (\hat{p}_x, \hat{p}_y) \in \mathbb{R}^2$ relative to the frame size.

Similarly, each ground-truth object $y_i \in Y$ consists of the ground-truth object position o_i (either a box b_i or a point p_i) and the ground-truth class label c_i. A matching algorithm is defined via an injective function $f : Y \to \hat{Y}$, where $f(y_i)$ is the candidate hypothesis $\hat{y}_j \in \hat{Y}$ assigned to the ground-truth object y_i. Given f, a loss function on pairs of sets Y and $\hat{Y}$ is defined as [38]:

$$\mathcal{L}(Y, \hat{Y}, f) = \sum_{i=1}^{M} \mathcal{L}_{pos}(y_i, f(y_i)) + \sum_{j=1}^{N} \mathcal{L}_c(\hat{y}_j, f^{-1}(\hat{y}_j)) \tag{1}$$

where $\mathcal{L}_{pos}$ is the object localization loss. The specific formulation of $\mathcal{L}_{pos}$ depends on the task (bounding box or center point). $\mathcal{L}_c$ is the class prediction loss, for which the Focal Loss was used [40]. $f^{-1}(\cdot)$ is an inverse of the matching function f:

$$f^{-1}(\hat{y}_j) = \begin{cases} y_i & \exists\, y_i \in Y, f(y_i) = \hat{y}_j \\ \emptyset & \text{otherwise} \end{cases}$$

Details on each loss term are provided below.

Focal Loss. The Focal Loss [40] is used to address the class imbalance between the foreground objects and the numerous potential background predictions. The class loss component $\mathcal{L}_c$ for

each prediction $\hat{y}_j$ involves the sigmoid Focal Loss. The total class loss for a prediction $\hat{y}_j$ is summed over all foreground classes $k \in \{1, ..., C\}$, where C is the number of object categories (excluding the $\emptyset$ class). The Focal Loss for a single class k and prediction j is defined using the α-balanced form:

$$\text{FL}(p_{jk}, y_{jk}) = -\alpha_t (1 - p_t)^\gamma \log(p_t) \tag{2}$$

where $p_{jk} = \sigma(x_{jk})$ is the predicted sigmoid probability for class k derived from the raw logits x_{jk}, and y_{jk} is the ground-truth label (1 if the matched ground-truth object $f^{-1}(\hat{y}_j)$ belongs to class k, and 0 otherwise). The terms p_t and α_t depend on the ground-truth label y_{jk}:

$$p_t = \begin{cases} p_{jk} & \text{if } y_{jk} = 1 \\ 1 - p_{jk} & \text{if } y_{jk} = 0 \end{cases}$$

$$\alpha_t = \begin{cases} \alpha & \text{if } y_{jk} = 1 \\ 1 - \alpha & \text{if } y_{jk} = 0 \end{cases}$$

The term $\gamma \geq 0$ is the focusing parameter, which reduces the relative loss for well-classified examples ($p_t \to 1$), thereby putting more focus on hard, misclassified examples. The term $\alpha \in [0, 1]$ is a weighting factor to address class imbalance, typically set as a hyperparameter (e.g., $\alpha = 0.25$). The total class loss contribution for prediction j in Equation 1 is calculated as:

$$\mathcal{L}_c(\hat{y}_j, y_j^{\text{match}}) = \sum_{k=1}^{C} \text{FL}(p_{jk}, y_{jk})$$

For unmatched predictions ($\hat{y}_j$ such that $y_j^{\text{match}} = 0$), the ground truth is $y_{jk} = 0$ for all foreground classes k, and the loss calculation correctly handles them as background/$\emptyset$ class predictions.

Bounding Box Loss is a linear combination of the L1 loss and the Generalized IoU (GIoU) loss [41]:

$$\mathcal{L}_{box}(b_i, \hat{b}_j) = \lambda_{L1} \mathcal{L}_{L1}(b_i, \hat{b}_j) + \lambda_{giou} \mathcal{L}_{giou}(b_i, \hat{b}_j) \tag{3}$$

where $y_i = (c_i, b_i)$ and $\hat{y}_j = (\hat{c}_j, \hat{b}_j)$. Loss terms $\mathcal{L}_{L1}, \mathcal{L}_{giou}$ are weighted by hyperparameters $\lambda_{L1}, \lambda_{giou} \in \mathbb{R}$.

Both the $\mathcal{L}_{L1}$ and $\mathcal{L}_{giou}$ components are normalized by the total number of actual ground-truth boxes (M) across the batch. Thus, for matched pairs in the bounding box task, $l_{pos}(y_i, \hat{y}_j) = \mathcal{L}_{box}(b_i, \hat{b}_j)$.

Center Point Loss. is L1 distance between the ground-truth center point p_i and the predicted center point $\hat{p}_j$:

$$\mathcal{L}_{point}(p_i, \hat{p}_j) = \mathcal{L}_{L1}(p_i, \hat{p}_j) \tag{4}$$

where $y_i = (c_i, p_i)$ and $\hat{y}_j = (\hat{c}_j, \hat{p}_j)$. The total localization loss term in Equation 1 for this task involves summing $\mathcal{L}_{point}$ over all matched pairs and normalizing by the total number of ground-truth objects (M) in the batch. Thus, for matched pairs in the center point task, $\mathcal{L}_{pos}(y_i, \hat{y}_j) = \mathcal{L}_{point}(p_i, \hat{p}_j)$.

Hungarian loss. The comparison cost function $\mathcal{L}_{match} : Yx\hat{Y} \rightarrow \mathbb{R}$ between hypotheses and ground-truth object was used. The comparison cost function $\mathcal{L}_{match}(y_i, \hat{y}_j)$ depends on the task. For the **bounding box task**, it is defined as [13]:

$$\mathcal{L}_{match}^{\text{bbox}}(y_i, \hat{y}_j) = -\mathbb{I}\{c_i \neq \emptyset\}\hat{p}_j(c_i) + \mathbb{I}\{c_i \neq \emptyset\}\mathcal{L}_{box}(b_i, \hat{b}_j) \tag{5}$$

where $\hat{p}_j(c_i)$ is the predicted probability of the ground-truth class c_i for prediction j. For the **center point task**, the cost replaces $\mathcal{L}_{box}$ with the appropriate point localization cost $\mathcal{L}_{point}$:

$$\mathcal{L}_{match}^{\text{point}}(y_i, \hat{y}_j) = -\mathbb{I}\{c_i \neq \emptyset\}\hat{p}_j(c_i) + \mathbb{I}\{c_i \neq \emptyset\}\mathcal{L}_{point}(p_i, \hat{p}_j) \tag{6}$$

Given the definition of comparison cost function $\mathcal{L}_{match}$, the optimal cost bipartite matching between Y and $\hat{Y}$ was found efficiently using the Hungarian algorithm [42]. The function f used in Equation 1 is then defined by this optimal assignment. The overall loss computed using this optimal matching is referred to as the Hungarian loss, $\mathcal{L}_{Hungarian}(Y, \hat{Y}) = \mathcal{L}(Y, \hat{Y}, f_{Hungarian})$.

3.5 Metrics

Mean Average Precision (mAP) was employed as the primary metric to evaluate the model's object detection performance. A widely used metric in object detection, mAP measures the average precision across various Intersection over Union (IoU) thresholds. It is adapted from information retrieval evaluation methods and was popularized in challenges such as PASCAL VOC [43]. Specifically, the mAP at an IoU threshold of 0.5 ($mAP@0.5$) is reported. This is a common choice established in early object detection benchmarks like PASCAL VOC [43]. The mAP at an IoU threshold of 0.75 ($mAP@0.75$), which provides a stricter evaluation criterion, is also presented. Additionally, the mAP averaged over IoU thresholds ranging from 0.5 to 0.95 with a step of 0.05 ($mAP@0.5:0.95$), as introduced by the COCO challenge [44], is included.

For evaluating the model's center point prediction performance, the Average Displacement Error (ADE) and Final Displacement Error (FDE) metrics were utilized. The ADE measures the average Euclidean distance between the predicted and ground truth center points over the sequence of frames, as shown in Equation 7.

$$\text{ADE} = \frac{\sum_{t=1}^{T} \sum_{i=1}^{M_t} ||\pi_{\text{pos}}(f_t(y_{i,t})) - \pi_{\text{pos}}(y_{i,t})||_2}{\sum_{t=1}^{T} M_t} \tag{7}$$

where:

- T is the total number of time steps (frames) in the sequence.
- M_t is the number of ground truth objects $y_{i,t}$ present at time step t.
- $y_{i,t}$ is the ground truth object (containing class and position $p_{i,t}$) at time t.
- f_t is the matching function at time t mapping ground truth objects Y_t to predictions $\hat{Y}_t$ and introduced in 3.4.
- $\pi_{\text{pos}}(\cdot)$ is a function extracting the center point position coordinates from its argument (e.g., $\pi_{\text{pos}}(y_{i,t}) = p_{i,t}$ and $\pi_{\text{pos}}(f_t(y_{i,t}))$ extracts the coordinates $\hat{p}$ from the matched prediction $\hat{y} = f_t(y_{i,t})$).

The FDE measures the Euclidean distance between the predicted and ground truth center points only at the final time step (T) of the sequence, normalized by the number of ground truth center points, as defined in Equation 8.

$$\text{FDE} = \frac{1}{M_T} \sum_{i=1}^{M_T} ||\pi_{\text{pos}}(f_T(y_{i,T})) - \pi_{\text{pos}}(y_{i,T})||_2 \tag{8}$$

where:

- T is the final time step of the sequence being evaluated.
- M_T is the number of ground truth objects $y_{i,T}$ present at the final time step T. (Note: This calculation assumes $M_T > 0$).
- $y_{i,T}$ is the i-th ground truth object at the final time step T.
- f_T is the matching function specific to the final time step T, mapping ground truths Y_T to predictions $\hat{Y}_T$ and introduced in 3.4.
- $\pi_{\text{pos}}(\cdot)$ is the function extracting the center point position coordinates from its argument.

4. Experiments

This section details the experimental setup and results. It begins by outlining the training procedures for the RPerceiver model in the bounding box prediction task and the RPerceiverMM model for the center point prediction task. Subsequently, a comparative analysis is presented, evaluating the RPerceiver architecture against the YOLOv8n baseline for object detection on the detection-moving-mnist-easy benchmark, with a focus on how spatio-temporal information is utilized and performance in challenging scenarios like overlapping objects and objects near frame borders. Finally, an extensive ablation study investigates the robustness of the RPerceiver and RPerceiverMM models under various conditions, including complete sensor failure and non-deterministic sensor input, across single-view and multi-view configurations using default, shuffle, and blind evaluation procedures.

4.1 Training for Bounding Boxes Prediction Task

The RPerceiver model was trained for the bounding box prediction task using the AdamW optimizer [45] with an initial learning rate of 10^{-4}. Training was conducted for 31 epochs, with the learning rate reduced by a factor of 10 at epochs 18 and 28. The training pipeline included data augmentation, normalization based on pre-calculated dataset statistics, and resizing frames to a fixed dimension of 320×320 pixels [46]. For a complete list of training hyperparameters, refer to Appendix 7.

4.2 Training for Center Points Prediction Task

The RPerceiverMM model was trained for the center point prediction task using the AdamW optimizer [45] with an initial learning rate of 10^{-4}. Training was conducted for 21 epochs, with the learning rate reduced by a factor of 10 at epoch 18. The training pipeline included data augmentation and normalization, ensuring robust performance. For a complete list of training hyperparameters, refer to Appendix 7.

4.3 Comparison Analysis

First, the task of object detection using the detection-moving-mnist-easy benchmark from Section 3.2 is considered. A comparative analysis against a still-image object detector of comparable size, YOLOv8n [47], is performed. The primary goal is to evaluate how the proposed RPerceiver architecture utilizes spatio-temporal information from the video input. The results are shown in Table 1.

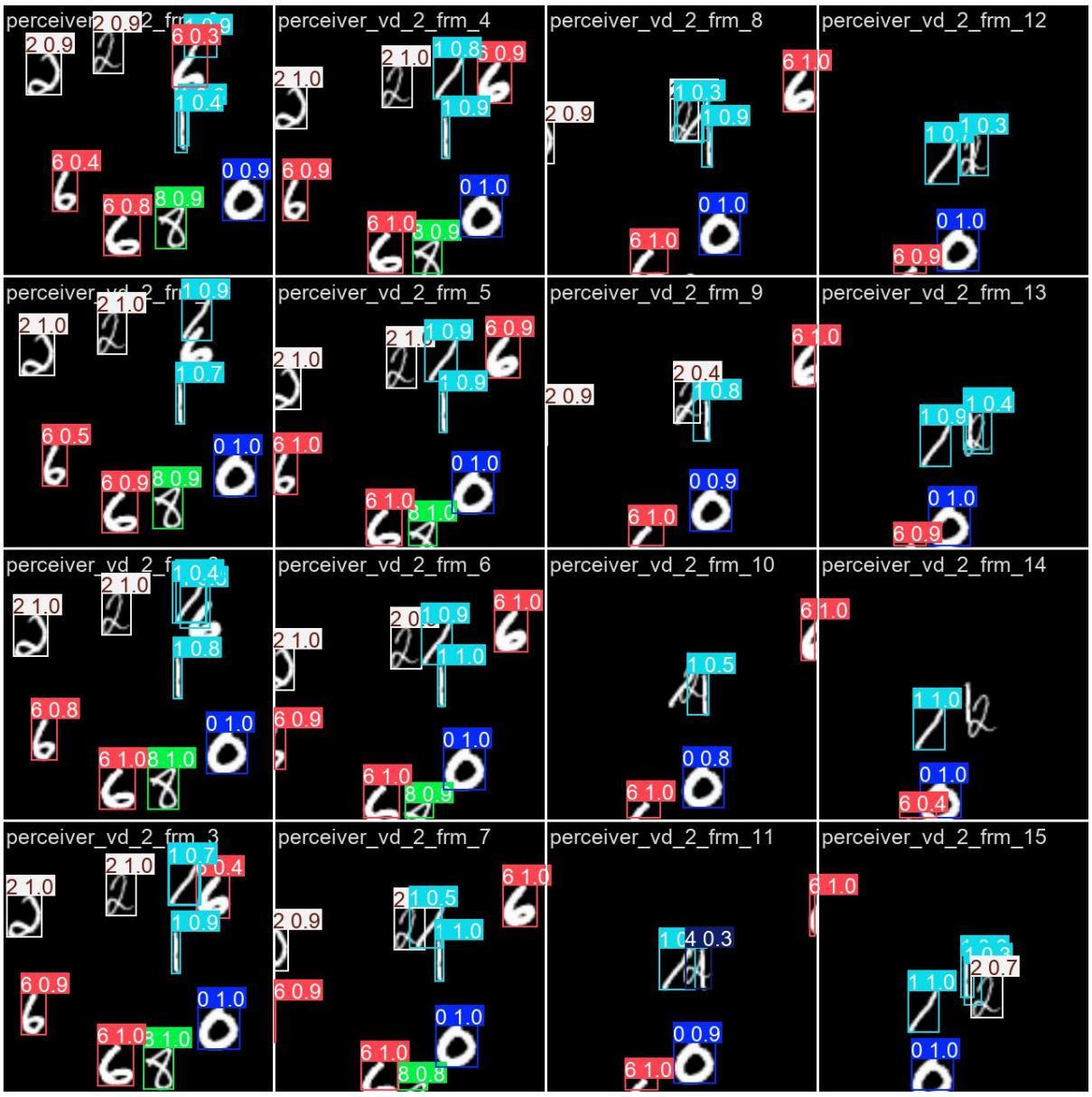

Figure 12. Visualization of RPerceiver predictions across sequential frames. Bounding boxes and confidence scores are displayed for detected digits. Notably, confidence scores can be less than 1.0 even for clear objects (e.g. digits in frame 1). Two challenging scenarios are analyzed: overlapping digits and digits near the frame border. The model shows limitations such as exhibiting false negatives with overlapping digits, as illustrated by the central cluster involving digits such as '1's and a '2' between frames 8 and 15. In contrast, predictions appear consistent between frames 5 and 13 for several digits near frame borders: the '2' and '6' on the left border, the '6' and '8' near the bottom border, and the '6' on the right border.

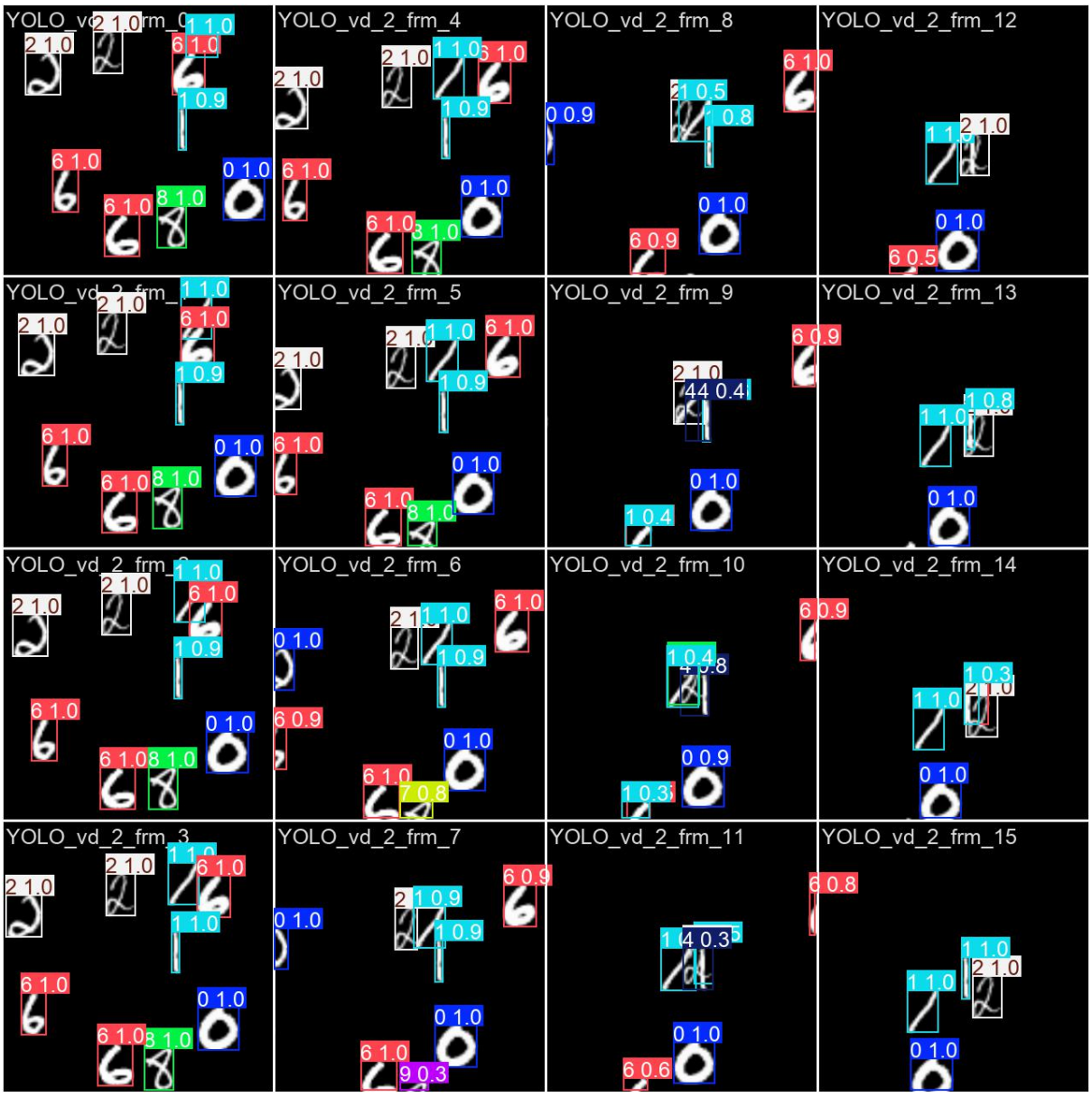

Figure 13. Visualization of YOLOv8n [47] predictions across sequential frames. Bounding boxes and confidence scores are displayed for detected digits. Notably, confidence scores are typically high (often 1.0) for relatively clear objects (e.g., digits in frame 1). Two challenging scenarios are analyzed: overlapping digits and digits near the frame border. The model appears to handle overlapping digits relatively effectively, with a few false negatives in the central cluster involving digits such as '1's and a '2' between frames 8 and 15. In contrast, predictions appear inconsistent between frames 5 and 13 for several digits near frame borders: the '2' on the left border, the '6' and '8' near the bottom border.

Table 1. Comparison with the baseline still image detector YOLOv8n [47] on the detection-moving-mnist-easy test split. RPerceiver achieves slightly better mAP_{50} and mAP_{75}, but shows worse mAP_{50-95} results. One initial hypothesis was that YOLOv8n's superior performance stemmed from its fine-tuning on a pretrained model, which could have provided an inherent advantage. However, experiments conducted with a non-pretrained YOLOv8n did not confirm this hypothesis. However, RPerceiver achieves these results with significantly fewer parameters and lower computational cost.

Model	**mAP**$_{50}$	**mAP**$_{75}$	**mAP**$_{50-95}$	**Params (M)**	**GFLOPS**
YOLOv8n	96.2	94.1	**92.3**	3	8.1
RPerceiver	**96.9**	**94.4**	90.3	**1**	**1.2**

To quantify the observations regarding model performance in challenging scenarios, as illustrated in Figures 12 and 13, the models were specifically evaluated on filtered subsets of the ground truth data. These subsets isolate two challenging conditions: overlapping objects (defined as pairs of ground truth objects with an Intersection over Union (IoU) greater than 10%) and border objects (defined as ground truth objects whose bounding boxes intersect with the frame border). These subsets represent 10.4% and 11.5% of the entire test set. The resulting performance metrics for these specific cases are presented in Table 2.

Table 2. Comparative analysis of YOLOv8n [47] and RPerceiver on specific scenarios within the detection-moving-mnist-easy test split: object overlaps ('Overlap') and proximity to image borders ('Border'). Postprocessing ('Post') is applied to both models. The data indicates that the still-image detector YOLOv8n achieves higher accuracy on overlapping objects. In contrast, RPerceiver significantly outperforms YOLOv8n on border cases, supporting the hypothesis that it effectively leverages temporal information from the video sequence.

Model	**Overlap**			**Border**		
	mAP$_{50}$	mAP$_{75}$	mAP$_{50-95}$	mAP$_{50}$	mAP$_{75}$	mAP$_{50-95}$
YOLOv8n	**87.5**	**82.4**	**75.7**	76.7	73.7	69.9
RPerceiver	84.0	68.8	61.0	**95.5**	**91.0**	**84.7**

4.4 Ablation Study

This ablation study evaluates the robustness of the RPerceiver and RPerceiverMM models against complete sensor failure and non-deterministic sensor input. The evaluation employs the center point prediction task. Two primary configurations were experimented with: `single-view` and `multi-view`. First, the `single-view` configuration utilized a single camera sensor, representing a basic object detection task using video input. Second, the `multi-view` setting involved processing information from multiple cameras to perform object detection within a unified spatial representation derived from these inputs.

Three distinct evaluation procedures were considered: `default`, `shuffle`, and `blind`. Additionally, a combination of the latter two procedures was evaluated.

`default` This procedure represents the normal operational regime where all sensors function as expected without any induced faults or input shuffle.

`shuffle` In this procedure, the sensor inputs are randomly permuted within each time step. Consequently, the model receives inputs from the sensors in a random order for that specific time step. Shuffling occurs only for sensor inputs within the same time step. This procedure is applicable only to the `multi-view` setup.

`blind` This procedure simulates a complete sensor(s) failure scenario where the input from camera sensor(s) becomes unavailable. The `blind` procedure is implemented by dropping the input stream from the sensor(s) after a midpoint time step, t_{half}. Consequently, information from the sensor(s) is unavailable to the model for the second half of the sequence.

It is important to note that the distinct terms "`dropout`" for the training procedure described in Section 3.3 and "`blind`" for the evaluation procedure were chosen intentionally, even though they represent a similar goal of removing the input data. The `blind` procedure can be understood as a specific case of the `dropout` mechanism, where the probability of dropping a sensor's input is set to 1.0, effectively "blinding" the model to that sensor's input.

The analysis begins by comparing the performance of the baseline RPerceiver (RP) model against its counterpart trained using the dropout procedure (RP (d)), as described in Section 3.3. This comparison is conducted under the `single-view` configuration using the `default` evaluation procedure. The results of this evaluation are presented in Table 3.

Table 3. Comparison of RPerceiver (RP) and RPerceiver trained with dropout (RP (d)) under the `single-view` configuration and `default` evaluation procedure. The '(d)' denotes training with dropout. Results are presented based on the number of digits in the input sequence. Metrics shown are Average Displacement Error (ADE), calculated over the second half of the sequence, and Final Displacement Error (FDE), calculated for the final frame. The results indicate that the baseline RPerceiver consistently outperforms the RPerceiver trained with dropout, although the difference is not very large, achieving lower error rates for both ADE and FDE across all tested sequence complexities.

Model	**1 digit**		**2 digits**		**4 digits**		**8 digits**		**10 digits**	
	ADE	**FDE**	**ADE**	**FDE**	**ADE**	**FDE**	**ADE**	**FDE**	**ADE**	**FDE**
RP	**0.717**	**0.725**	**0.794**	**0.793**	**0.958**	**0.934**	**1.520**	**1.382**	**3.273**	**2.611**
RP (d)	0.745	0.793	0.838	0.882	1.034	1.065	1.656	1.583	3.626	3.12

Next, the same models, RPerceiver (RP) and RPerceiver trained with dropout (RP (d)), are compared. This comparison is conducted under the `single-view` configuration using the `blind` evaluation procedure. The results for this `blind` test are detailed in Table 4.

Table 4. Comparison of RPerceiver (RP) and RPerceiver trained with dropout (RP (d)) under the `single-view` configuration and `blind` evaluation procedure. The '(d)' denotes training with dropout. Results are presented based on the number of digits in the input sequence. Metrics shown are Average Displacement Error (ADE), calculated over the second half of the sequence, and Final Displacement Error (FDE), calculated for the final frame. The results under the `blind` condition demonstrate a significant advantage for the RPerceiver trained with dropout (RP (d)). This model maintains substantially lower error rates compared to the baseline RP, which exhibits a sharp degradation in performance under the `blind` evaluation procedure.

Model	**1 digit**		**2 digits**		**4 digits**		**8 digits**		**10 digits**	
	ADE	**FDE**	**ADE**	**FDE**	**ADE**	**FDE**	**ADE**	**FDE**	**ADE**	**FDE**
RP	42.210	34.754	36.087	28.83014	32.146	24.397	31.32972	23.730	33.334	25.935
RP (d)	**1.541**	**1.101**	**1.907**	**1.364**	**2.658**	**1.905**	**5.084**	**3.679**	**8.198**	**6.627**

For the `multi-view` configuration, a tile augmentation technique is introduced to the dataset pipeline. This process divides each frame into a specified grid of views, partitioning the

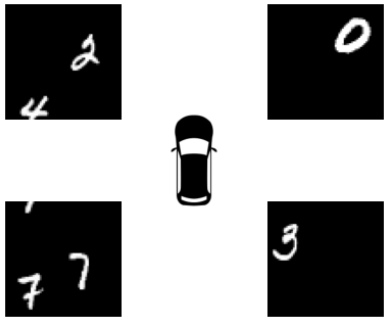

Figure 14. Bird Eye View (BEV) of the frame raster divided into grid.

corresponding ground truths as well. In the experiments, a 2×2 grid configuration is utilized. This effectively treats the tiled frame as a simplified spatial layout, akin to a Bird's-Eye View (BEV) where each tile represents a spatial region. Figure 14 illustrates an example of a frame divided using this grid approach.

The ablation study is now extended to the `multi-view` configuration, evaluating the RPerceiverMM (RPMM) model and its variants trained with training procedures from Section 3.3: shuffle training (RPMM (s)), dropout training (RPMM (d)), and combined shuffle and dropout training (RPMM (d, s)). These models are compared against the baseline RPMM under four distinct evaluation procedures: `default`, `shuffle`, `blind`, and a combined `blind` and `shuffle` scenario. The performance metrics for these multi-view evaluations are presented in Table 5.

Table 5. Comparison of RPerceiverMM (RPMM) variants under the `multi-view` configuration across different evaluation procedures: `default`, `shuffle`, `blind`, and combined `blind, shuffle`. Model notations indicate training procedures: '(s)' for shuffle training, '(d)' for dropout training, and '(d, s)' for combined dropout and shuffle training. Metrics shown are Average Displacement Error (ADE), calculated over the second half of the sequence, Final Displacement Error (FDE), calculated for the final frame, and the number of model parameters in millions (Params (M)). The results highlight trade-offs: baseline RPMM performs best under `default` conditions. While these training procedures introduce a small performance penalty in the `default` evaluation, they provide substantial improvements under performance-degrading evaluation strategies. Models trained with shuffle (s) excel in the `shuffle` evaluation, while dropout-trained models (d) show superior robustness in the `blind` scenario. The combined training (d, s) yields the most robust performance under the combined `blind, shuffle` condition, demonstrating the effectiveness of targeted training strategies for specific failure modes. Interestingly, RPMM (d) also shows notable robustness under the `shuffle` evaluation, despite not being explicitly trained for this condition.

Model	**Default**		**Shuffle**		**Blind**		**Blind, Shuffle**		**Params (M)**
	ADE	**FDE**	**ADE**	**FDE**	**ADE**	**FDE**	**ADE**	**FDE**	
RPMM	0.760	0.753	23.701	23.759	17.952	25.568	25.559	24.696	1.2
RPMM (s)	0.823	0.814	0.812	0.799	13.644	20.124	13.670	20.080	1.2
RPMM (d)	0.881	0.937	4.043	4.345	1.471	2.093	5.648	7.341	1.2
RPMM (d, s)	1.073	1.152	0.956	1.022	1.729	2.345	1.632	2.287	1.2

5. Discussion

One of the limitations discovered during this research is that the RPerceiver's precision is fairly poor at higher levels of Intersection over Union (IoU). Comparative analysis proved this, showing that YOLOv8n performed better than RPerceiver at $mAP@0.5:0.95$, despite both models achieving comparable mAP scores at easier IoU metrics. This suggests that while RPerceiver can identify objects at a basic level, its ability to accurately define their boundaries is less refined than YOLOv8n when stricter overlap criteria are applied. These findings indicate that further investigation is needed to enhance RPerceiver's precision. Potential directions for future work could involve increasing the RPerceiver model's capacity to be more comparable to that of YOLOv8n, for instance by increasing its depth of the RPerceiver or experimenting with different backbone architectures, which might improve its ability to learn finer details. This is particularly relevant as one of the findings regarding transformer based object detection model DETR [13] was its lower performance on small objects, where authors anticipated that future work could address this limitation, perhaps in a manner similar to how the development of Feature Pyramid Networks (FPN) [48] improved performance on objects of varying scales in other architectures [13].

Another limitation of this study revolves around the "detection-moving-mnist-easy" dataset. While intentionally simplified for controlled experiments, it does not fully represent all the challenges presented by real-world video data. In particular, the dataset is missing effects such as motion blur and objects entering the video frame. Therefore, another important direction for future research is to increase the complexity of the dataset. Introducing these real-world challenges could provide more nuanced insights into the comparative analysis of different object detection models and further highlight the strengths and weaknesses of architectures like RPerceiver.

Finally, an idea for a future ablation study is to investigate the core intuition behind the RPerceiver architecture for object detection. The hypothesis is that the variable N, representing a dimension of the latent array, corresponds to the number of objects the RPerceiver actively tracks, while D represents the number of features for each object, like position, size or color. As objects appear and disappear, the system must periodically initiate tracking for new objects within its N available slots, which necessitates a mechanism to monitor slot utilization. To explore this, one could incorporate metrics commonly used in multi-object tracking tasks (MOT). This could involve treating the latent array index as a tracked object ID to observe whether a detected

object is consistently tracked by the same latent array vector in time or if the object's identity shifts from one latent vector to another as the object moves through the frame scene. Such an investigation could validate the architectural design choices and provide deeper understanding of RPerceiver's internal object representation and tracking capabilities.

6. Conclusion

Two novel architectures were introduced: the Recurrent Perceiver (RPerceiver) and its multi-modal variant (RPerceiverMM). These were inspired by the Perceiver architecture's [17] ability to handle high-dimensional, multi-modal data but were adapted for sequential inputs, such as video. To facilitate this research, a novel dataset, "detection-moving-mnist-easy", was generated. This dataset provides controlled video sequences with annotations for evaluating both bounding box and center point (keypoint) prediction tasks.

The RPerceiver was compared against a strong baseline, YOLOv8n [47], on the bounding box detection task. While both models showed similar $mAP@0.5$ and $mAP@0.75$ scores, YOLOv8n performed better at $mAP@0.5:0.95$. Further evaluation showed that YOLOv8n performed better with overlapping objects when the IoU between objects was $> 10\%$. However, the RPerceiver demonstrated superior performance in handling objects near frame borders, suggesting its effectiveness in leveraging temporal information, particularly when objects have partial visibility. Notably, RPerceiver achieved competitive mAP scores with significantly fewer parameters and lower computational cost compared to YOLOv8n.

Another contribution of this work was the investigation of training strategies to enhance model resilience against sensor input degradation, simulating real-world Autonomous Driving Systems (ADS) challenges like complete sensor failure and asynchronous data arrival. `dropout` and `shuffle` training procedures were introduced and evaluated. Ablation studies using the center point prediction task demonstrated the efficacy of these strategies. In single-view scenarios, training RPerceiver with `dropout` resulted in significantly better performance under simulated complete sensor failure compared to the baseline model, although it incurred a minor performance penalty under normal conditions.

Extending the analysis to multi-view scenarios using RPerceiverMM, the benefits of targeted training procedures were further confirmed. Models trained with input `shuffling` excelled when evaluated with shuffled sensor inputs, while `dropout`-trained models showed the best robustness against simulated sensor failure. The model trained with both `dropout` and `shuffling` provided the most robust performance under the combined `blind, shuffle` evaluation, highlighting the potential for tailoring training methods to specific anticipated failure modes. While these robustness-enhancing training procedures introduced a slight performance decrease under default, non-degraded conditions, they offer substantial improvements in resilience against common sensor issues faced by ADS.

References

[1] Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. Surface Vehicle Recommended Practice. Revision April 2021, Superseding J3016 JUN2018. SAE International, Apr. 2021.

[2] Litman T. Autonomous Vehicle Implementation Predictions: Implications for Transport Planning. *Transportation Research Board 94th Annual MeetingTransportation Research Board* (2015).

[3] Yurtsever E., Lambert J., Carballo A., and Takeda K. A Survey of Autonomous Driving: Common Practices and Emerging Technologies. *IEEE Access* 8 (2020), pp. 58443–58469. DOI: 10.1109/ACCESS.2020.2983149. https://ieeexplore.ieee.org/document/9046805/?arnumber=9046805 (03/10/2025).

[4] Han W., Khorrami P., Paine T. L., Ramachandran P., Babaeizadeh M., Shi H., Li J., Yan S., and Huang T. S. Seq-NMS for Video Object Detection. Aug. 22, 2016. DOI: 10.48550/arXiv.1602.08465. arXiv: 1602.08465 [cs]. http://arxiv.org/abs/1602.08465 (03/27/2025). Pre-published.

[5] Kang K., Ouyang W., Li H., and Wang X. Object Detection from Video Tubelets with Convolutional Neural Networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016, pp. 817–825. DOI: 10.1109/CVPR.2016.95. arXiv: 1604.04053 [cs]. http://arxiv.org/abs/1604.04053 (03/27/2025).

[6] Kang K., Li H., Yan J., Zeng X., Yang B., Xiao T., Zhang C., Wang Z., Wang R., Wang X., and Ouyang W. T-CNN: Tubelets With Convolutional Neural Networks for Object Detection From Videos. *IEEE Transactions on Circuits and Systems for Video Technology* 28.10 (Oct. 2018), pp. 2896–2907. DOI: 10.1109/TCSVT.2017.2736553. https://ieeexplore.ieee.org/document/8003302 (03/27/2025).

[7] Lu Y., Lu C., and Tang C.-K. Online Video Object Detection Using Association LSTM. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[8] Xiao F. and Lee Y. J. Video Object Detection with an Aligned Spatial-Temporal Memory. July 2018. DOI: 10.48550/arXiv.1712.06317. arXiv: 1712.06317 [cs]. (04/07/2025).

[9] Bahdanau D., Cho K., and Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate. May 19, 2016. DOI: 10.48550/arXiv.1409.0473. arXiv: 1409.0473 [cs]. http://arxiv.org/abs/1409.0473 (05/12/2025). Pre-published.

[10] Guo C., Fan B., Gu J., Zhang Q., Xiang S., Prinet V., and Pan C. Progressive Sparse Local Attention for Video Object Detection. Aug. 16, 2019. DOI: 10.48550/arXiv.1903.09126. arXiv: 1903.09126 [cs]. http://arxiv.org/abs/1903.09126 (05/12/2025). Pre-published.

[11] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., and Polosukhin I. Attention Is All You Need. Aug. 1, 2023. arXiv: 1706.03762 [cs]. http://arxiv.org/abs/1706.03762 (06/28/2024). Pre-published.

[12] Russakovsky O., Deng J., Su H., Krause J., Satheesh S., Ma S., Huang Z., Karpathy A., Khosla A., Bernstein M., Berg A. C., and Fei-Fei L. ImageNet Large Scale Visual Recognition Challenge. Jan. 30, 2015. DOI: 10.48550/arXiv.1409.0575. arXiv: 1409.0575 [cs]. http://arxiv.org/abs/1409.0575 (05/05/2025). Pre-published.

[13] Carion N., Massa F., Synnaeve G., Usunier N., Kirillov A., and Zagoruyko S. End-to-End Object Detection with Transformers. May 28, 2020. arXiv: 2005.12872 [cs]. http://arxiv.org/abs/2005.12872 (07/04/2024). Pre-published.

[14] Zhu X., Su W., Lu L., Li B., Wang X., and Dai J. Deformable DETR: Deformable Transformers for End-to-End Object Detection. Mar. 17, 2021. arXiv: 2010.04159 [cs]. http://arxiv.org/abs/2010.04159 (07/05/2024). Pre-published.

[15] Zhou Q., Li X., He L., Yang Y., Cheng G., Tong Y., Ma L., and Tao D. TransVOD: End-to-End Video Object Detection with Spatial-Temporal Transformers. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.6 (June 1, 2023), pp. 7853–7869. DOI: 10.1109/TPAMI.2022.3223955. arXiv: 2201.05047 [cs]. http://arxiv.org/abs/2201.05047 (10/04/2024).

[16] Wang H., Tang J., Liu X., Guan S., Xie R., and Song L. PTSEFormer: Progressive Temporal-Spatial Enhanced TransFormer Towards Video Object Detection. Sept. 6, 2022. DOI: 10.48550/arXiv.2209.02242. arXiv: 2209.02242 [cs]. http://arxiv.org/abs/2209.02242 (05/14/2025). Pre-published.

[17] Jaegle A., Gimeno F., Brock A., Zisserman A., Vinyals O., and Carreira J. Perceiver: General Perception with Iterative Attention. June 22, 2021. arXiv: 2103.03206 [cs, eess]. http://arxiv.org/abs/2103.03206 (06/28/2024). Pre-published.

[18] Tampuu A., Matiisen T., Semikin M., Fishman D., and Muhammad N. A Survey of End-to-End Driving: Architectures and Training Methods. *IEEE Transactions on Neural Networks and Learning Systems* 33.4 (Apr. 2022), pp. 1364–1384. DOI: 10.1109/TNNLS.2020.3043505. https://ieeexplore.ieee.org/document/9310544/ (05/13/2025).

[19] Matos F., Bernardino J., João Durães, and Cunha J. A Survey on Sensor Failures in Autonomous Vehicles: Challenges and Solutions. *Sensors* 24.16 (16 Jan. 2024), p. 5108. DOI: 10.3390/s24165108. https://www.mdpi.com/1424-8220/24/16/5108 (04/02/2025).

[20] Park J., Delgado R., and Choi B. W. Real-Time Characteristics of ROS 2.0 in Multiagent Robot Systems: An Empirical Study. *IEEE Access* 8 (2020), pp. 154637–154651. DOI: 10.1109/ACCESS.2020.3018122. https://ieeexplore.ieee.org/document/9172073 (05/05/2025).

[21] Fischler M. and Elschlager R. The Representation and Matching of Pictorial Structures. *IEEE Transactions on Computers* C-22.1 (Jan. 1973), pp. 67–92. DOI: 10.1109/T-C.1973.223602. https://ieeexplore.ieee.org/document/1672195/?arnumber=1672195 (04/03/2025).

[22] Hinton G. E. and Salakhutdinov R. R. Reducing the Dimensionality of Data with Neural Networks. *Science* 313.5786 (July 28, 2006), pp. 504–507. DOI: 10.1126/science.1127647. https://www.science.org/doi/10.1126/science.1127647 (04/03/2025).

[23] LeCun Y., Bengio Y., and Hinton G. Deep Learning. *Nature* 521.7553 (May 2015), pp. 436–444. DOI: 10.1038/nature14539. https://www.nature.com/articles/nature14539 (06/28/2024).

[24] Krizhevsky A., Sutskever I., and Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc., 2012. https://proceedings.neurips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html (04/03/2025).

[25] Girshick R., Donahue J., Darrell T., and Malik J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014 IEEE Conference on Computer Vision and Pattern Recognition. June 2014, pp. 580–587. DOI: 10.1109/CVPR.2014.81. https://ieeexplore.ieee.org/document/6909475 (04/03/2025).

[26] Jiao L., Zhang R., Liu F., Yang S., Hou B., Li L., and Tang X. New Generation Deep Learning for Video Object Detection: A Survey. *IEEE Transactions on Neural Networks and Learning Systems* 33.8 (Aug. 2022), pp. 3195–3215. DOI: 10.1109/TNNLS.2021.3053249. https://ieeexplore.ieee.org/document/9345705/?arnumber=9345705 (09/04/2024).

[27] Liu M. and Zhu M. Mobile Video Object Detection with Temporally-Aware Feature Maps. Mar. 28, 2018. DOI: 10.48550/arXiv.1711.06368. arXiv: 1711.06368 [cs]. http://arxiv.org/abs/1711.06368 (04/07/2025). Pre-published.

[28] Liu W., Anguelov D., Erhan D., Szegedy C., Reed S. E., Fu C.-Y., and Berg A. SSD: Single Shot MultiBox Detector. *European Conference on Computer Vision* (2015). DOI: 10.1007/978-3-319-46448-0_2.

[29] Hochreiter S. and Schmidhuber J. Long Short-Term Memory. *Neural Computation* 9.8 (1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[30] Ballas N., Yao L., Pal C., and Courville A. Delving Deeper into Convolutional Networks for Learning Video Representations. Mar. 2016. DOI: 10.48550/arXiv.1511.06432. arXiv: 1511.06432 [cs]. (04/08/2025).

[31] Howard A. G., Zhu M., Chen B., Kalenichenko D., Wang W., Weyand T., Andreetto M., and Adam H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. Apr. 17, 2017. DOI: 10.48550/arXiv.1704.04861. arXiv: 1704.04861 [cs]. http://arxiv.org/abs/1704.04861 (05/15/2025). Pre-published.

[32] Shi X., Chen Z., Wang H., Yeung D.-Y., Wong W.-k., and Woo W.-c. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. Version 2. Sept. 19, 2015. DOI: 10.48550/arXiv.1506.04214. arXiv: 1506.04214 [cs]. http://arxiv.org/abs/1506.04214 (12/28/2024). Pre-published.

[33] Itti L., Koch C., and Niebur E. A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20.11 (Nov. 1998), pp. 1254–1259. DOI: 10.1109/34.730558. https://ieeexplore.ieee.org/document/730558 (05/14/2025).

[34] Dai J., Qi H., Xiong Y., Li Y., Zhang G., Hu H., and Wei Y. Deformable Convolutional Networks. *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017 IEEE International Conference on Computer Vision (ICCV). Oct. 2017, pp. 764–773. DOI: 10.1109/ICCV.2017.89. https://ieeexplore.ieee.org/document/8237351 (05/14/2025).

[35] Lin J., Mao X., Chen Y., Xu L., He Y., and Xue H. D^2ETR: Decoder-Only DETR with Computationally Efficient Cross-Scale Attention. Mar. 2, 2022. DOI: 10.48550/arXiv.2203.00860. arXiv: 2203.00860 [cs]. http://arxiv.org/abs/2203.00860 (05/14/2025). Pre-published.

[36] Srivastava N., Mansimov E., and Salakhutdinov R. Unsupervised Learning of Video Representations using LSTMs. 2016. arXiv: 1502.04681 [cs.LG]. https://arxiv.org/abs/1502.04681.

[37] Lecun Y., Bottou L., Bengio Y., and Haffner P. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE* 86.11 (Nov. 1998), pp. 2278–2324. DOI: 10.1109/5.726791. https://ieeexplore.ieee.org/document/726791 (05/14/2025).

[38] Stewart R. and Andriluka M. End-to-End People Detection in Crowded Scenes. July 8, 2015. arXiv: 1506.04878 [cs]. http://arxiv.org/abs/1506.04878 (08/07/2024). Pre-published.

[39] Redmon J., Divvala S., Girshick R., and Farhadi A. You Only Look Once: Unified, Real-Time Object Detection. May 9, 2016. arXiv: 1506.02640 [cs]. http://arxiv.org/abs/1506.02640 (07/31/2024). Pre-published.

[40] Lin T.-Y., Goyal P., Girshick R., He K., and Dollár P. Focal Loss for Dense Object Detection. Version 2. Feb. 7, 2018. DOI: 10.48550/arXiv.1708.02002. arXiv: 1708.02002 [cs]. http://arxiv.org/abs/1708.02002 (01/16/2025). Pre-published.

[41] Rezatofighi H., Tsoi N., Gwak J., Sadeghian A., Reid I., and Savarese S. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. Apr. 15, 2019. DOI: 10.48550/arXiv.1902.09630. arXiv: 1902.09630 [cs]. http://arxiv.org/abs/1902.09630 (05/03/2025). Pre-published.

[42] Kuhn H. W. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly* 2.1–2 (1955), pp. 83–97. DOI: 10.1002/nav.3800020109. https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800020109 (05/03/2025).

[43] Everingham M., Van Gool L., Williams C. K. I., Winn J., and Zisserman A. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision* 88.2 (June 1, 2010), pp. 303–338. DOI: 10.1007/s11263-009-0275-4. https://doi.org/10.1007/s11263-009-0275-4 (04/28/2025).

[44] Lin T.-Y., Maire M., Belongie S., Bourdev L., Girshick R., Hays J., Perona P., Ramanan D., Zitnick C. L., and Dollár P. Microsoft COCO: Common Objects in Context. Feb. 21, 2015. DOI: 10.48550/arXiv.1405.0312. arXiv: 1405.0312 [cs]. http://arxiv.org/abs/1405.0312 (04/28/2025). Pre-published.

[45] Loshchilov I. and Hutter F. Decoupled Weight Decay Regularization. Jan. 4, 2019. DOI: 10.48550/arXiv.1711.05101. arXiv: 1711.05101 [cs]. http://arxiv.org/abs/1711.05101 (05/04/2025). Pre-published.

[46] Redmon J. and Farhadi A. YOLO9000: Better, Faster, Stronger. Dec. 25, 2016. DOI: 10.48550/arXiv.1612.08242. arXiv: 1612.08242 [cs]. http://arxiv.org/abs/1612.08242 (05/04/2025). Pre-published.

[47] Jocher G., Qiu J., and Chaurasia A. Ultralytics YOLO. Version 8.0.0. Jan. 2023. https://github.com/ultralytics/ultralytics.

[48] Lin T.-Y., Dollár P., Girshick R., He K., Hariharan B., and Belongie S. Feature Pyramid Networks for Object Detection. Apr. 19, 2017. arXiv: 1612.03144 [cs]. http://arxiv.org/abs/1612.03144 (07/31/2024). Pre-published.

7. Acknowledgements

I express my sincere gratitude to my supervisor, Tambet Matiisen, for his invaluable guidance, support, reviews, and strategic direction for the research. His unwavering support and insightful guidance were crucial throughout this work. I also extend my thanks to the University of Tartu for providing an excellent research environment, including access to essential research journals and High-Performance Computing (HPC) resources, which were crucial for running experiments on state-of-the-art hardware. Finally, I am deeply grateful to my family for their unwavering support and encouragement throughout my studies, which has always motivated me to give my best.

Appendices

A. Training Hyperparameters

Table 6. Hyperparameters used for training the RPerceiver model for bounding box prediction task.

Hyperparameter	**Value**	**Description**
backbone	cnn	Backbone type
batch_size	1	Batch size for training
bbox_loss_coef	5	L1 box coefficient
eos_coef	0.1	Relative classification weight of the no-object class
epochs	32	Number of training epochs
giou_loss_coef	2	GIoU box coefficient
learning_rate	0.0001	Learning rate
learning_rate_backbone	0.0001	Learning rate
num_frames	20	Number of frames.
object_detection	True	Use object detection prediction head
resize_frame	320	Resize frame to this size
scheduler_step_size	18,28	Scheduler step size
set_cost_bbox	5	L1 box coefficient in the matching cost
set_cost_class	1	Class coefficient in the matching cost
set_cost_giou	2	giou box coefficient in the matching cost
weight_decay	0.01	Weight decay for optimizer
weight_loss_bce	1	Weight loss binary cross entropy
weight_loss_center_point	5	Weight loss center point

Table 7. Hyperparameters used for training the RPerceiverMM model for center point task.

Hyperparameter	**Value**	**Description**
backbone	cnn	Backbone type
batch_size	1	Batch size for training
bbox_loss_coef	None	L1 box coefficient
eos_coef	None	Relative classification weight of the no-object class
epochs	21	Number of training epochs
giou_loss_coef	None	GIoU box coefficient
learning_rate	0.0001	Learning rate
learning_rate_backbone	0.0001	Learning rate
num_frames	12	Number of frames.
object_detection	None	Use object detection prediction head
resize_frame	None	Resize frame to this size
scheduler_step_size	18	Scheduler step size
set_cost_bbox	None	L1 box coefficient in the matching cost
set_cost_class	None	Class coefficient in the matching cost
set_cost_giou	None	giou box coefficient in the matching cost
weight_decay	0.01	Weight decay for optimizer
weight_loss_bce	1	Weight loss binary cross entropy
weight_loss_center_point	5	Weight loss center point

B. Model Hyperparameters

Table 8. Hyperparameters used for configuring RPerceiver model.

Hyperparameter	Value	Description
enc_layers	1	Number of layers in Perceiver encoder
enc_nheads_cross	1	Number of cross-attention heads
hidden_dim	128	Latent dimension size
max_freq	10	Maximum frequency for Fourier encoding
nheads	1	Number of latent self-attention heads
num_freq_bands	6	Number of frequency bands for Fourier encoding
num_queries	16	Number of latents, or induced set points, or centroids
self_per_cross_attn	1	Number of self-attention blocks per cross-attention block

Table 9. Hyperparameters used for configuring RPerceiverMM model.

Hyperparameter	Value	Description
enc_layers	1	Number of layers in Perceiver encoder
enc_nheads_cross	1	Number of cross-attention heads
hidden_dim	128	Latent dimension size
max_freq	10	Maximum frequency for Fourier encoding
nheads	1	Number of latent self-attention heads
num_freq_bands	6	Number of frequency bands for Fourier encoding
num_queries	16	Number of latents, or induced set points, or centroids
self_per_cross_attn	1	Number of self-attention blocks per cross-attention block

C. Train Procedures

The Python code snippet below details the implementation of the `shuffle` and `dropout` procedures within the model's forward pass method. The `shuffle` mechanism, activated when `self.apply_shuffle` is true, randomizes the processing order of sensor views (`view_indices`) within each timestep using `get_shuffled_order`,

mimicking potential real-world asynchronicity. The `dropout` strategy is applied conditionally based on the timestep: dropout is only considered for timesteps in the second half of the input sequence (i.e., where `timestep >= midpoint`, with `midpoint` being the sequence's halfway point). For these eligible timesteps, a probabilistic check (`should_drop_view(dropout_probability)`) determines if the current view (`drop_this_view`) is actually dropped. When a view is dropped, the code bypasses both the feature extraction via the `backbone` and the `cross_attention` step within the perceiver layers for that view. However, the `self_attention` mechanism within each layer is executed unconditionally. The model's core state, represented by `latent_array`, is updated iteratively within the loop processing each view: after every perceiver layer's self-attention step, the `latent_array` is updated. If the view was not dropped, this update incorporates information fused via cross-attention before the self-attention step; otherwise, the self-attention refines the `latent_array` based solely on the state propagated from processing the previous view. Finally, after processing all views for a timestep, a prediction is generated from the updated `latent_array`.

```
def forward(self, sequence_of_frames, dropout_probability):
    latent_array = self.initial_learned_hidden_state
    all_predictions = []
    midpoint = len(sequence_of_frames) // 2
    for timestep, frame_views in enumerate(sequence_of_frames):
        should_apply_dropout = (timestep >= midpoint)

        view_indices = list(range(len(frame_views)))
        if self.apply_shuffle:
            view_indices = get_shuffled_order(view_indices)

        for view_index in view_indices:
            view_data = frame_views[view_index]
            drop_this_view = should_apply_dropout
            if drop_this_view:
                drop_this_view = should_drop_view(dropout_probability)

            extracted_features = None
            if not drop_this_view:
                extracted_features = self.backbone(view_data)
```

```
            for perceiver_layer in self.perceiver.layers:
                if not drop_this_view:
                    latent_array = perceiver_layer.cross_attention(
                        q=latent_array,
                        kv=extracted_features,
                        sensor_id=view_index,
                    )

                latent_array = perceiver_layer.self_attention(
                    qkv=latent_array,
                )

        prediction = self.detection_head(latent_array)
        all_predictions.append(prediction)

    return all_predictions
```

License

Non-exclusive licence to reproduce the thesis and make the thesis public

I, **Maksim Ploter**,

(author's name)

1. grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the digital archives of the University of Tartu until the expiry of the term of copyright, my thesis,

   Eyes Wide Shut: Analyzing Object Detection Performance Under Degraded Sensor Input Scenarios,

   (title of thesis)

   supervised by Tambet Matiisen.

   (supervisor's name)

2. grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright;

3. am aware of the fact that the author retains the rights specified in points 1 and 2;

4. confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Maksim Ploter

15/05/2025