Imitation learning in multi-sensor self-aware autonomous systems

UNIVERSITÀ DEGLI STUDI
DI GENOVA

Universidad
Carlos III de Madrid

Abrham Shiferaw Alemaw

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in
Electrical Engineering, Electronics and Automation

Department of Electrical, Electronic, Telecommunications Engineering and Naval Architecture (DITEN)
University of Genoa

Department of Systems Engineering and Automation (DISA)
University Carlos III of Madrid

Advisors:
Carlo Regazzoni
David Martín Gómez

Defense Month: April 2025

Imitation learning in multi-sensor self-aware autonomous systems

Abrham Shiferaw Alemaw

Joint Doctorate in Interactive and Cognitive Environments
JD-ICE

XXXVII cycle

Acknowledgements

This PhD Thesis has been developed in the framework of, and according to, the rules of the Joint Doctorate in Interactive and Cognitive Environments JD-ICE with the cooperation of the following Universities:

Università degli Studi di Genova (UNIGE)

DITEN - Dept. of Electrical, Electronic, Telecommunications Engineering and Naval Architecture

ISIP40 - Information and Signal Processing for Cognitive Telecommunications

Supervisor: Prof. Carlo Regazzoni

UNIVERSITÀ DEGLI STUDI
DI GENOVA

Universidad Carlos III de Madrid (UC3M)

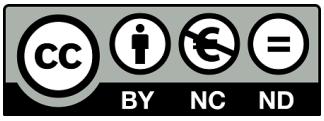
IEEA - Doctoral Program in Electrical Engineering, Electronics and Automation

DISA - Dep. of Systems Engineering and Automation

ISL- Intelligent System Laboratory

Supervisor: Prof. David Martín Gomez

Universidad
Carlos III de Madrid



Acknowledgements

I would like to thank the people that make this thesis possible, with their discussion, support, and collaboration.

First of all, I am deeply grateful to my supervisors Carlo Regazzoni, David Martín Gómez, and Lucio Marcenaro for their unwavering support in everything I have accomplished. I extend my gratitude to Professor Carlo Regazzoni for his tireless effort in guiding and inspiring me in those challenging but also interesting times. The interesting weekly meetings which always trigger my mind for scientific discussions, elaborations, the professional criticism and many more, I say thank you again. I am grateful to Prof. David Martín Gómez for his impeccable support and guidance during my stay in UC3M. Designing new scenarios, unwavering support and availability to do many experiments for each scenario, indeed, I relay thank you. I extend my sincere thanks to Prof. Lucio Marcenaro for his invaluable support and availability for discussion. My sincere gratitude also goes to Prof. Pamela Zontone, for the discussions, the support that you did in this year is so amazing and invaluable to me.

My special thanks to all of my colleagues at University of Genoa and at the University Carlos III of Madrid, who provided me a friendly research environment.

I am eternally grateful to my many friends for everything you did to ease my life. You are actually my family.

Last but not least, I would like to thank my family for their encouraging support and love. Thank you Belu.

I always remember you Eme.

Published and Submitted Papers

1. G. Slavic, **A. S. Alemaw**, L. Marcenaro, C. Regazzoni, D. Martín "A Kalman Variational Autoencoder Model assisted by Odometric Clustering for Video Frame Prediction and Anomaly Detection", IEEE Transactions on Image Processing.

   (This journal paper is partly included in the thesis, in Chapters 6, 7 and 8. The material from this source included in this thesis is not singled out with typographic means and references.)

2. **A. S. Alemaw**, G. Slavic, Hafsa Iqbal, D. Martin, L. Marcenaro, C. Regazzoni, "A Data-Driven Approach for the Localization of Interacting Agents via a Multi-Modal Dynamic Bayesian Network Framework", IEEE International Conference on Advanced Video and Signal-Based Surveillance, (AVSS 2022).

   (This conference paper is wholly included in the thesis, in Chapter 6. The material from this source included in this thesis is not singled out with typographic means and references.)

3. **A. S. Alemaw**, G. Slavic, Pamela Zontone, L. Marcenaro, D. Martin, C. Regazzoni, ''Modeling Interactions between Autonomous Agents in a Multi Agent Self-Awareness Architecture", IEEE Transactions on Multimedia.

   (This journal paper is wholly included in the thesis, in Chapter 7. The material from this source included in this thesis is not singled out with typographic means and references.)

4. **A. S. Alemaw**, Pamela Zontone, L. Marcenaro, Pablo Marin, D. Martin, C. Regazzoni, ''Integrated Learning and Decision Making for Autonomous Agents through Energy based Bayesian Models", IEEE International Conference on Information Fusion (FUSION 2024)

(This conference paper is wholly included in the thesis, in Chapter 8. The material from this source included in this thesis is not singled out with typographic means and references.)

Summary

The fusion of different sensory information in an integrated way for an Artificial Intelligence (AI) model is the most advancing research direction in building a self-aware agent. Single data modalities are short in representing and exploiting the representation in any real scenario. Information fusion plays a vital role in a holistic understanding of a situation in addition to sensor fail-safe applications. Nowadays, training an AI model involves experience modeling from expert-generated data. These datasets are vast, highly non-linear, and high dimensional. Integrating these highly complex and dynamic experiences robustly is challenging. Understanding different structures in higher dimensional sensory input, also fusing them with lower dimensional information, has been a primary objective in the progress of powerful generative models.

Recent advancements in sensory information integration for the development of explainable AI models are progressing through the combination of probabilistic self-expressive Bayesian models and Deep Learning approaches. Dynamic Bayesian Networks (DBNs) are unified general probabilistic representations and inference mechanisms for time-series domains that can be used to introduce explainability in an AI model through their causal-effect relations. Integrating DBNs with Variational Autoencoders (VAEs), to represent higher dimensional data, makes them powerful and scalable, and can be used for self-supervised learning of higher dimensional data distributions.

Autonomous driving is an area where the research direction is advancing to integrate explainable AI models to facilitate the realization of full autonomy. Increasing convenience to free up time, increasing the mobility of people with disabilities, improving environmental health, minimizing human driving errors, and minimizing the economic impact through car sharing and carpooling, make the current AI research trend focus more on generalized AI models for Self-Driving cars.

This thesis introduces a novel Multi-Agent Self-Awareness Architecture (MASAA) that integrates generative World Model (WM), Experience Models (Multi-Modal Perception, MMP), Active First-Person Model, Cost model (Multi-level Anomalies) through the Short-Term Memory Module (Multi-Agent Coupled Markov Jump Particle Filters, MAC-MJPFs).

In the MASAA framework, a set of modules are introduced to enable an agent to localize itself while interacting with neighboring agents simultaneously.

One of the main goals of MASAA is to enhance the generalizability and interpretability of an AI model. To this end, this thesis introduces a novel methodology that integrates multi-sensorial data from proprioceptive and exteroceptive sensors of an agent, coupled in a Hierarchical DBN model in an Active Inference framework. A lower-dimensional unsupervised learning stage, considering both odometry and action modalities, is carried out by applying Null Force Filtering (NFF) and a modified Growing Neural Gas (GNG) clustering algorithm, thus producing lower-dimensional knowledge. A self-supervised higher-dimensional video modality learning stage using VAEs, guided by the learned lower-dimensional vocabularies, creates integrated multi-sensorial inference knowledge. An online model-based active learning in continuous and discrete state spaces and action spaces for decision-making in the Active Inference framework is developed. These representation and decision-making models are evaluated using localization and interaction benchmarks.

Resumen

La fusión de distintas fuentes de información sensorial de una forma integrada para crear un modelo de Inteligencia Artificial (IA) es la línea de investigación más avanzada en la construcción de un agente autoconsciente. Las modalidades de datos individuales son insuficientes para representar y explotar la representación en cualquier escenario real. La fusión de información desempeña un papel vital en una comprensión holística de una situación, además de las aplicaciones a prueba de fallos de sensores. Hoy en día, el entrenamiento de un modelo de IA implica el modelado de experiencias a partir de datos generados por expertos. Estos conjuntos de datos son muy extensos, extremadamente no lineales y de elevada dimensionalidad. Integrar de forma sólida estas experiencias altamente complejas y dinámicas es un desafío. Comprender las diferentes estructuras de la información sensorial de alta dimensionalidad y fusionarla con la información de baja dimensionalidad, ha sido un objetivo principal en el desarollo de potentes modelos generativos.

Los avances recientes en la integración de información sensorial para el desarrollo de modelos explicables de IA están progresando a través de la combinación de modelos bayesianos autoexpresivos y enfoques de aprendizaje profundo. Las Redes Bayesianas Dinámicas (DBN) son representaciones probabilísticas generales unificadas y mecanismos de inferencia para dominios de series temporales que se pueden utilizar para introducir explicabilidad en un modelo de IA a través de sus relaciones causa-efecto. La integración de DBNs con autocodificadores variacionales (VAEs), para representar datos de dimensiones superiores, los hace potentes y escalables, y se pueden utilizar para el aprendizaje autosupervisado de distribuciones de datos de dimensiones superiores.

La conducción autónoma es un área en la que la dirección de la investigación está avanzando para integrar modelos explicables de IA que faciliten la consecución de la autonomía total. Aumentar la comodidad para disponer de más tiempo, mejorar la movilidad de las personas con discapacidad, mejorar la salud ambiental, minimizar los errores de conducción humana y minimizar el impacto económico a través del uso compartido del automóvil hacen que la tendencia actual de investigación de IA se centre más en modelos generalizados de IA para automóviles autónomos.

Esta tesis presenta una nueva arquitectura de autoconciencia de múltiples agentes (MASAA) que integra un modelo generativo del mundo (WM), modelos de experiencia (percepción multimodal, MMP), un modelo activo en primera persona y un modelo de costes (anomalías multinivel) a través del módulo de memoria de corto plazo (filtros de partículas de salto de Markov acoplados a múltiples agentes, MAC-MJPF). En el marco MASAA, se introduce un conjunto de módulos para permitir que un agente se localice a sí mismo mientras interactúa con agentes vecinos simultáneamente.

Uno de los principales objetivos de MASAA es mejorar la generalización e interpretabilidad de un modelo de IA. Con este fin, esta tesis presenta una metodología novedosa que integra datos multisensoriales de sensores propioceptivos y exteroceptivos de un agente, acoplados en un modelo DBN jerárquico en un marco de inferencia activa. Se lleva a cabo una etapa de aprendizaje no supervisado de menor dimensión, considerando odometría y modalidades de acción, aplicando un filtrado de fuerza nula y un algoritmo de agrupamiento de gases neuronales en crecimiento (GNG) modificado, produciendo así conocimiento de menor dimensión. Una etapa de aprendizaje de modalidad de video de mayor dimensión autosupervisada que utiliza VAE, guiada por los vocabularios de menor dimensión aprendidos, crea conocimiento de inferencia multisensorial integrado. Se desarrolla un aprendizaje activo basado en modelos en línea en espacios de estados continuos y discretos y espacios de acción para la toma de decisiones en el marco de inferencia activa. Estos modelos de representación y toma de decisiones se evalúan utilizando puntos de referencia de localización e interacción.

Contents

Published and Submitted Papers vi

List of Figures xvii

List of Tables xxiv

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Outline and contributions . . . 5

2 Self-Driving and State-space modeling 8
- 2.1 History of Self-Driving . . . 8
- 2.2 Levels os Autonomy in Self-Driving . . . 9
- 2.3 State-space models . . . 10
  - 2.3.1 Definition . . . 10
  - 2.3.2 Generalized state-space models . . . 11
- 2.4 Generative models . . . 14
- 2.5 Dimensionality Reduction and Latent Variable models . . . 14
  - 2.5.1 Autoencoders . . . 15
  - 2.5.2 Variational Inference in Latent Variable Models . . . 16
  - 2.5.3 Variational Autoencoders (VAE) . . . 19
  - 2.5.4 Dynamical Variational Autoencoders . . . 20
- 2.6 Summary and discussion . . . 21

3 Representation and inference models 22
- 3.1 Probabilistic Graphical Models (PGMs) . . . 22
- 3.2 Bayesian Networks . . . 23
- 3.3 Dynamic Bayesian Networks . . . 24
- 3.4 State estimation models . . . 26

3.4.1 Kalman Filter . . . 27
3.4.2 Particle Filter . . . 29
3.4.3 Markov Jump Particle Filter . . . 30
3.5 Unsupervised Clustering . . . 31
3.5.1 Self Organizing Map (SOM) . . . 31
3.5.2 GNG . . . 32
3.6 Anomaly Signals . . . 33
3.7 Summary and discussion . . . 33

4 Self-awareness and Self-Expressive Systems 35
4.1 Definition and characteristics . . . 35
4.2 Self-awareness Levels . . . 36
4.3 Self-awareness Architecture . . . 37
4.3.1 Sensing Model . . . 38
4.3.2 World Model . . . 38
4.3.3 Multi-Modal Perception Model . . . 40
4.3.4 Memory Model . . . 40
4.3.5 Cost Model . . . 41
4.3.6 Actor Model . . . 41
4.3.7 Actuation Model . . . 42
4.3.8 Incremental Learning Model . . . 42
4.4 Reinforcement Learning . . . 42
4.5 Imitation Learning . . . 44
4.6 Active Inference . . . 45
4.7 Summary and discussion . . . 48

5 Datasets 49
5.1 iCab . . . 51
5.2 KITTI . . . 52
5.3 CARLA . . . 52
5.4 Heterogeneous Agents . . . 53
5.5 Summary . . . 55

6 A Data-Driven Approach for the Localization of Interacting Agents via a Multi-Modal Dynamic Bayesian Network Framework 56
6.1 Introduction . . . 56
6.2 Multi-Modality Learning Framework . . . 59

6.2.1 Learning Lower Dimensional Vocabulary . . . 59
6.2.2 Higher Dimensional Vocabulary Learning . . . 61
6.3 Multi-modality Interaction Modeling . . . 63
6.4 Used Datasets . . . 65
6.5 Results . . . 66
6.6 Comparison with state-of-the-art trajectory estimation methods . . . 73
6.7 Summary . . . 73

7 Modeling Interactions between Autonomous Agents in a Multi-Agent Self-Awareness Architecture 75
7.1 Introduction . . . 76
7.2 Component Analysis of the system model . . . 79
7.3 Learning the World Model . . . 83
7.4 Learning the Multi-Modal Perception Model . . . 84
7.5 Online Modal Interaction . . . 88
7.6 Employed Datasets . . . 90
7.7 Learned H-DBN Models . . . 94
7.7.1 World Model H-DBN . . . 95
7.7.2 Multi-Agent CG-KVAE H-DBN . . . 95
7.7.3 Single-Agent CG-KVAE H-DBN . . . 96
7.8 Localization and Interaction Algorithm . . . 97
7.9 Resampling and Anomaly Signals . . . 101
7.9.1 Anomaly Signals . . . 102
7.9.2 Particle resampling . . . 104
7.10 Results . . . 104
7.11 Comparison with state-of-the-art trajectory estimation methods . . . 111
7.12 Summary . . . 112

8 Incremental Learning and Decision Making in Continuous Action and State Space Models for Autonomous Vehicles 113
8.1 Introduction . . . 114
8.2 Method Description: Active Learning and Inference . . . 117
8.2.1 Odometry Learning . . . 117
8.2.2 Action Learning . . . 117
8.2.3 Video Learning . . . 119
8.2.4 Active Learning and Inference . . . 121
8.3 Multi-Level Messages . . . 125

8.4 Online Decision Making 126
8.4.1 Variational Free energy 128
8.4.2 Expected Free energy 129
8.4.3 Action prediction and selection 131
8.5 Online Incremental Learning and Decision Making 131
8.6 Results 135
8.7 Summary 139

9 Conclusion and Future Work **141**
9.1 Conclusions 141
9.2 Open questions and Future Work 144

Bibliography **147**

List of Figures

2.1 Single dimensional trajectory depicted across time. 12
2.2 Multi dimensional trajectory representing generalized states. 13
2.3 Autoencoder 16
2.4 Variational autoencoder (VAE) 17

3.1 Dynamic Bayesian Network (DBN). A two-time-slice Dynamic Bayesian network (2TDBN), with two levels of hierarchy, consists of variables such as x_{t+1} and $\tilde{z}_{t+1}$ whose parents are variables $\tilde{z}_t$ and $\tilde{z}_{t+1}$. 24
3.2 Hierarchal DBN (H-DBN). A two-time-slice Dynamic Bayesian network (2TDBN) with three levels of hierarchy. Blue arrows show inter-slice connections. Red arrows show intra-slice connections. Dotted black lines show time evolution in DBNs while solid black lines represent links for messages passing. 25

4.1 MASAA: Cognitive architecture for autonomous agents. Arrows 1 to 5 are processes, based on model output errors, for correcting (changing) the corresponding model as an incremental learning process. The sensing module is a hardware module interfacing with the external world by translating physical stimuli into an initial internal representation. It produces a time series of different observations for both the WM and MMP modules. The WM processes the incoming lower dimensional data and produces vocabularies representing the agent's self-state in a context. The MMP module uses the WM vocabulary as a guiding attentive knowledge. The integrated WM representation is used as an encoded collection memory of sensorial active experiences of the agent, to be used when the agent has to perform experienced tasks. The learned WM Situation model is applied under a First-Person viewpoint as a result of embedded explicit relations between external and internal variables. The short-term memory module represents the communication module, which is the interaction medium using Markov Jump Particle Filtering. The Active First-Person module is an online process that selects actions based on learned policies and transfers action commands to the Actuation module. © 2024 IEEE. 39

5.1 Kolb's Experiential Learning Cycle, learning is the process whereby knowledge is created through the transformation of experience. 50

5.2 (a) iCab (intelligent Campus automobile), electric golf carts navigate autonomously within the UC3M campus(a). (b) KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute), public datasets for use in mobile robotics and autonomous driving. (c) Simulation platform supports flexible specification of sensor suites, environmental conditions, and full control of all static and dynamic actors. 50

5.3 iCab: iCab 2 overtaking iCab1, odometry trajectories. Five experiments, where experiments 1 to 3 for training, experiment 4 for validation and experiment 5 for testing. 51

5.4 KITTI, odometry trajectories of six sequences. 52

5.5 CARLA Simulation for interacting agents. 53

5.6 CARLA: Agent 2 overtaking Agent 1, odometry trajectories. Eight experiments, where experiments 01 to 04 are for training, experiments 05 and 07 are for validation, and experiments 06 and 08 are for testing. 54

5.7 (a) Drone type used in the interaction experiment. (b) iCab and drone interaction single image taken by iCab camera sensor. 54

6.1 Block diagram of the proposed method. This architecture can support more than two autonomous agent but here we used it for unmanned vehicles u_1 and u_2. © 2022 IEEE. 60
6.2 Multi-sensory-multi-agent learned Hierarchical Dynamic Bayesian Network (H-DBN) situation model. © 2022 IEEE. 64
6.3 iCab clustered positional overtaking trajectories of the interacting agents. Arrows indicate the direction of motion of vehicles encoded in $\tilde{s}_{k,u_i}$ and different colors represent the different clusters. The filled cyan colors represents the mean position of each cluster. © 2022 IEEE. 66
6.4 CARLA clustered positional overtaking trajectories of the interacting agents. Arrows indicate the direction of motion of vehicles encoded in $\tilde{s}_{k,u_i}$ and different colors represent the different clusters. The filled black colors represents the mean position of each cluster. © 2022 IEEE. 67
6.5 (a) KLD, (b) Reconstruction, and (c) Total VAE losses from the iCab validation dataset of iCab2, for different content related latent state (a) dimensions: $a \in \left(\mathbb{R}^{10}, \mathbb{R}^{20}, \mathbb{R}^{30}, \mathbb{R}^{40}, \mathbb{R}^{50}\right)$. 67
6.6 (a) Predicted trajectories of iCab agent u_2, only from video modality. (b) Localization of iCab agent u_1, from u_2 based on $r_{d_t}^{u_{12}}$. © 2022 IEEE. 68
6.7 (a) Predicted trajectories of iCab agent u_1, only from video modality. (b) Localization of iCab agent u_2, from u_1 based on $r_{d_t}^{u_{21}}$. © 2022 IEEE. 68
6.8 (a, b) Show the smooth predicted trajectories of itself (blue) and of the interacting agents (red) using the iCab dataset. © 2022 IEEE. 69
6.9 (a) Predicted trajectories of CARLA agent u_2, only from video modality. (b) Localization of CARLA agent u_1, from u_2 based on $r_{d_t}^{u_{12}}$. © 2022 IEEE. . . 69
6.10 (a) Predicted trajectories of CARLA agent u_1, only from video modality. (b) Localization of CARLA agent u_2, from u_1 based on $r_{d_t}^{u_{21}}$. © 2022 IEEE. . . 70
6.11 (a, b) Show the smooth predicted trajectories of itself (blue) and of the interacting agents (red) utilizing the CARLA dataset. © 2022 IEEE. 70
6.12 (a) Predicted GSs of u_1 (itself), (b) u_2 (interacting). The trajectories are smoothed to magnify the positional and velocity information. © 2022 IEEE. 71
6.13 (a) smoothed trajectory of the relative distance vector $r_{d_t}^{u_{12}}$, (b) Predicted GSs of u_2 (itself) magnified to show trajectories of position and velocity. © 2022 IEEE. 71
6.14 (a) Smoothed trajectory of u_1 (interacting) depicting position and velocity trajectories. (b) Relative distance vector trajectories $r_{d_t}^{u_{21}}$, illustrating the relative positions and relative velocities. © 2022 IEEE. 72

6.15 (a) KLD, (b) Reconstruction, and (c) Total VAE losses from the CARLA validation dataset of agent2, for different content related latent state (a) dimensions: $a \in \left(\mathbb{R}^{10}, \mathbb{R}^{20}, \mathbb{R}^{30}, \mathbb{R}^{40}, \mathbb{R}^{50}\right)$. 72

7.1 Single-agent H-DBN. A self-localization generative model that connects higher and lower dimensional sensory input variables, to produce a combined self-knowledge. © 2025 IEEE. 80
7.2 Simplified version of the self-awareness (MASAA) architecture that is currently implemented. The cost module, which is not included in this simplified MASAA version, is implicitly included in the short-term memory. The sensing component transmits odometry and video data to the WM and MMP components, respectively. First, the WM learns vocabulary representing the odometry data. Second, this vocabulary is used to guide (focus) the learning of video data according to the odometry module. Last, WM and MMP communicate through the short-term memory to apply the learned knowledge online (testing phase). © 2025 IEEE. 81
7.3 MASAA: Self-representation learning and inference frameworks. A learned vocabulary of proprioceptive information (controlled hallucination) drives exteroceptive information learning. © 2025 IEEE. 85
7.4 Multi-Agent H-DBN. It illustrates the interaction which models the interrelation between the higher and lower dimensional vocabularies, and also represents multi-agent interaction knowledge through a relative distance vector for the overtaking interaction. © 2025 IEEE. 90
7.5 Flow chart model of multi-modal MJPF: a detailed description, including stages of the filtering process, are described in algorithm 2. © 2025 IEEE. . 92
7.6 Training trajectory: overtaking interaction takes place in a starting point of a curved zone. The light-blue oval shape represents the overtaking area. © 2025 IEEE. 94
7.7 Training trajectory: overtaking interaction takes place in a curved area. The light-blue oval shape represents the overtaking area. © 2025 IEEE. 95
7.8 Testing trajectory: Overtaking is performed in straight area under the condition of heavy rain. The light-blue oval shape represents the overtaking area. © 2025 IEEE. 96
7.9 Testing trajectory. This scenario is performed to test traffic light anomaly and light rain in the evening time of the day. The light-blue oval shape represents the overtaking area. © 2025 IEEE. 97

7.10 Clustered positional trajectories of iCab - WM. The red arrow represents mean velocity of each cluster. Mean position of each cluster is represented by the filled black colored points. The difference from fig. 6.3 of Chapter 6 is due to the fusion of second-order motion parameters in the clustering algorithm to improve the WM vocabulary. © 2025 IEEE. 98
7.11 Clustered positional trajectories of CARLA - WM. Mean position and mean velocity are represented by the black doted points and red arrows respectively. As in fig. 7.10 second-order motion parameters are fused with the GSs in the clustering algorithm. © 2025 IEEE. 99
7.12 (a)KLD, (b)Reconstruction, and (c)Total VAE losses from the KITTI validation dataset of agent2, for different content related latent state (a) dimensions: $a \in \left(\mathbb{R}^{10}, \mathbb{R}^{20}, \mathbb{R}^{30}, \mathbb{R}^{40}, \mathbb{R}^{50}\right)$ 100
7.13 Anomaly signals: Exp:06. The highest anomaly signal is when the agent passes under a bridge. The light variability and limited number of such video frames in the training data results in higher video reconstruction anomaly. © 2025 IEEE. 102
7.14 Anomaly signals: Exp:08. The light variability and the red traffic light introduction in the testing dataset results in a higher anomaly signal. © 2025 IEEE. 103
7.15 iCab trajectory. WM guided MMP learned trajectory from video sequences. © 2025 IEEE. 106
7.16 Carla trajectory. WM guided MMP learned trajectory from video sequences. © 2025 IEEE. 107
7.17 Kitti trajectory. WM guided MMP learned trajectory from video sequences. © 2025 IEEE. 108
7.18 (Transition probability Matrices. These probability matrices are used to govern the learning of higher dimensional data, i.e., to assign the most probable WM states to each video sequence. © 2025 IEEE. 109
7.19 (Predictions of the trajectories corresponding to all particles for testing experiments. The light blue oval shape represents the overtaking zone. © 2025 IEEE. 109

8.1 Integrated self-representation learning and decision-making from exteroceptive information driven by learned vocabulary of proprioceptive information. Lower-dimensional learning stage is represented as $S.1$, higher-dimensional learning stage is represented as $S.1$, and $S.3$ is online learning, inference and decision making process. © 2024 IEEE. 116

8.2 Clustered trajectory of an overtaking agent showing the output of odometry cluster information. © 2024 IEEE. 118
8.3 Learned trajectory from stage 2 of fig. 8.1 (video learning). Cluster guided trajectory © 2024 IEEE. 119
8.4 Learned HC-DBN. The agent now has a combined vocabularies of action, odometry, and video, that are mapped by the configurator network. © 2024 IEEE. 124
8.5 Expert agent action sequences and predicted action sequences of an ego agent. © 2024 IEEE. 124
8.6 Expert agent action sequences and updated action sequences of an ego agent. © 2024 IEEE. 125
8.7 Multi-level anomaly signal measurement models: A) anomaly signal indicator at the observations and the predicted generalized states; B) anomaly signal indicator between predictive message; C) anomaly signal indicator at cluster level between predictive and diagnostic messages; 127
8.8 Preferred expert agent action sequences vs learned action sequences of an ego agent. 128
8.9 Explaining different sequences of actions in terms of anomaly signals. Online performance signal measured from the video latent state information. © 2024 IEEE. 130
8.10 Anomaly signal detection. Combined video and odometry anomaly signals. Reconstruction anomaly is related to the video content and video anomaly is regarding the motion aspect of the video. 132
8.11 Anomaly signal characterization. Characterizing anomaly helps to know the threshold of new and learned environments. The orange line is the content related video encoding and the light blue signal is motion related anomaly signal of the video encoding. 133
8.12 Training trajectories from Carla simulator. The red arrow shows the intended trajectory that the agent follows to both exploit and explore learned and new regions in the environment. 134
8.13 Experienced regions and possible exploration zones. These zones are based on a testing data containing positional sequences that are taken as time steps. 135

8.14 Anomaly signal thresholding. The threshold used here is by taking the mean of the anomaly signal and by scaling it with the standard error ($\mu + 2.5\sigma$). The blur experiment is to check how the threshold is affected if we minimize the luminosity of each video frame. The legends *anomaly blur th* and *anomaly th* are short forms for anomaly blur threshold and anomaly threshold respectively. These signals are computed for the content related latent states of the VAE encodings. 136
8.15 MMP generalizability. The input images are used to test the generalizability of the learned network. 137
8.16 MMP incremental learning video reconstruction anomaly. The signals show how different transfer learning approaches affects the MMP output. 138
8.17 Odometry prediction anomaly in the continuous GSs. Exploration zones are shown to indicate how the anomaly signals are varying in each hierarchical generative states. 138
8.18 Filter performance anomaly in learned and explored zones. Exploration zones are shown to indicate how the anomaly signals are varying in each hierarchical generative states. 139

List of Tables

1.1 Summary of the contributions of this thesis. Visual summary of the developed systems are presented in Chapter 8, fig. 8.1: stage 1 includes the lower dimensional odometry and action vocabulary learning; stage 2 the higher dimensional vocabulary learning; and stage 3 the incremental online active learning stage. 5

I Analysis of latent state size influence on the relation of odometry and video vocabularies 68

II Table displays the quantitative analysis of the proposed method based on Root Mean Square Error (RMSE) measurement. We perform two different experiments to test our model, and obtained consistent results. © 2022 IEEE. 72

I Table displaying the temperature values according to eq. (7.6), and the corresponding RMSE error between the ground truth generalized states and the learned generalized states. The RMSE values shown here are the mean of the generalized state RMSE errors. From this, the best temperature value is between 10 and 25. Hence, $n = 15$ is used in our learning models. 100

II Analysis of latent state size influence on the relation of odometry and video vocabularies. 101

III Table displays the quantitative analysis of the proposed method based on Root Mean Square Error (RMSE) measurements. 105

IV Table displaying the testing parameters for MAC-MJPF. The temperature value in the testing scenario is according to eq. (7.6). 105

V MAC-MJPF Complexity Analysis. MAC-MJPF performs best between 50 and 500 particles, with low mean RMSE of generalized states and within acceptable time. Time is in minutes. 110

VI Table displays the quantitative analysis of the overtaking model and following model, tested with overtaking data set, based on Root Mean Square Error (RMSE) measurement. We used the generalized training data as ground truth to compute the errors. 110

Nomenclature

Acronyms / Abbreviations

AE Auto Encoder

AFPM Active First-Person Model

AI Artificial Intelligence

AIF Active Inference

ALVINN Autonomous Land Vehicle in a Neural Network

AM Autobiographical Memory

ANN Artificial Neural Network

AV Autonomous Vehicle

BC Behavior Cloning

BN Bayesian Network

CA Collective Awareness

CDS Cognitive Dynamic System

CG-KVAE Cluster-Guided Kalman Variational Autoencoder

CNN Convolutional Neural Network

CVAE Conditional Variational Autoencoder

DAgger Data Aggregation

DAS Driver Assistance System

DBN Dynamic Bayesian Network

DGM Directed Graphical Model

DL Deep Learning

DNN DeepNeural Network

DVAE Dynamical Variational Autoencoder

EFE Expected Free Energy

EKF Extended Kalman Filter

ELBO Evidence Lower BOund

FPM First-Person Model

FPV First-Person Viewpoint

GDBN Generalized Dynamic Bayesian Network

GE Generalized Error

GM Generative Model

GMM Gaussian Mixture Model

GNG Growing Neural Gas

GO Generalized Observation

GPS Global Positioning System

GPU Graphics Processing Unit

GS Generalized State

GT Ground Truth

H-DBN Hierarchical Dynamic Bayesian Network

HI Human Intelligence

HMM Hidden Markov Model

iCab Intelligent Campus Automobile

IID Independent and Identically Distributed

IL Imitation Learning

IMM Interacting Multiple Model

IMU Inertial Measurement Unit

IRL Inverse Reinforcement Learning

KF Kalman Filter

KITTI Karlsruhe Institute of Technology and Toyota Technological Institute

KL Kullback-Leibler

KLD Kullback-Leibler-Divergence

KLDA Kullback-Leibler Divergence Abnormality

KNN K-nearest-neighbours

KVAE Kalman Variational Autoencoder

LGSSM Linear Gaussian State Space Model

LiDAR Light Detection and Ranging

LSTM LongShort-Term Memory

M-GNG Modified Growing Neural Gas

MAC-MJPF Multi-Agent Coupled Markov Jump Particle Filter

MAH-DBN Multi Agent Hierarchical Dynamic Bayesian Network

MASAA Multi-Agent Self-Awareness Architecture

MDP Markov Decision Processes

MJPF MarkovJumpParticle Filter

ML Machine Learning

MLP Multi-Layer Perceptron

MMP Multi-Modal Perception

MMP-MA Multi-Modal Perception for Multi-Agent

MMP-SA Multi-Modal Perception for Single Agent

MSE Mean Square Error

NFF Null Force Filter

NN Neural Network

ODE Ordinary Differential Equation

ORAD On-Road Automated Driving

PF Particle Filter

PGM Probabilistic Graphical Model

POMDP Partially observable Markov Decision Processes

RL Reinforcement Learning

RMSE Root Mean Square Error

RNN Recurrent Neural Network

SA Self-awareness

SEA Society of Automotive Engineers

SLAM Simultaneous Localization and Mapping

SLDS Switching Linear Dynamical System

SM Situation Model

SOM Self Organizing Map

SSM State Space Models

UAV Unmanned Aerial Vehicle

UC3M University Carlos III of Madrid

UGM Undirected Graphical Model

UGV Unmanned Ground Vehicle

UKF Unscented Kalman Filter

VAE Variational Auto Encoder

VFE Variational Free Energy

WM World Model

WM-MA World Model for Multi-Agent

WM-SA World Model for Single Agent

Chapter 1

Introduction

1.1 Motivation

Structured knowledge building for autonomous systems, that learn from experience and current situations, focuses on the linkage between input expert data and its associated learning mechanisms. It involves the integration of multi-sensorial information sources representing internal and external situations (Esterle et al., 2023; Regazzoni et al., 2020). Building representation schemas and transferring this representation into knowledge that enables decision-making is the goal of an AI model. These models should couple environmental situations with self-representation in the environment to become a functioning system. Considering autonomous vehicles, to transform the decision-making ability of a self-driving agent, an explicit integration of internal system information with the outside world is mandatory to have a holistic understanding of the situation. However, the environment where the autonomous vehicle operates is high-dimensional and shared with different erratic users, which challenges the modeling process. On top of this, the dynamicity introduces many variabilities that affect the decision-making process. Hence, the learning strategy must accommodate these challenging conditions in time-series domains.

In time-series problem domain, machine learning algorithms are the focal research areas. Data-driven algorithms (Kingma and Welling, 2014; LeCun et al., 2015), in the machine learning domain, are the most promising advancements in this research direction to achieve general artificial intelligence. Self-supervised approaches are among this family of methodologies that are gaining more focus to achieve generalizability and interpretability when performing a particular activity or task. Learning different and complicated hidden structures, and associating them with lower dimensional knowledge to execute tasks, is the goal of self-supervised learning algorithms.

Integrating higher-dimensional hidden structural knowledge with lower-dimensional models is a self-awareness behavior that allows one to understand oneself and the surrounding environment. Autonomous agents, like self-driving vehicles, must aim to integrate sensing, perceiving, interpreting, planning, and decision-making capabilities to be characterized as self-aware agents. Self-awareness, from a sensorial integration point of view, seeks to understand the general state of the external environment jointly with the internal coherency of the autonomous agent generalized states (Esterle et al., 2023; Regazzoni et al., 2020). Self-awareness models ought to be discriminant and generative in the face of learned behavior deviations. Probabilistic interaction models equip these models to be generalizable and measurable when deviations occur due to the anomalous incidents present in the dynamic environment. A probabilistic system should maintain an agent at equilibrium by preserving it in a highly probable state. Maintaining an agent in equilibrium is the idea behind a Bayesian brain (Da Costa et al., 2020; Friston et al., 2017; Fountas et al., 2020), which models the world around it to generate predictions and then draws inferences about incoming sensory information. It has a powerful linkage in determining the causes of the outcomes. The Bayesian brain concept is a condition of a specific world that produces some sensory observation, establishing posterior beliefs about the causes and processing the observation data through a generative model (the brain) using likelihoods and prior information. Interestingly, modeling artificial autonomous agents in a Bayesian brain framework makes it plausible to generalize and reason the task the agent has executed.

Generative models have structures that accommodate different and interrelated distributions (Slavic et al., 2023; Fraccaro et al., 2017). Probabilistic Graphical Models (PGMs) are suited for this type of joint distribution, because of their property in representing such Bayesian knowledge. In autonomous driving, the generated and the observed information are typically time-series data. From the family of PGMs, Dynamic Bayesian Networks (DBNs) are well-suited to deal with the causal-effect relations in Bayesian-based Learned models.

The objective of this thesis is to study and develop a unified Multi-Agent Self-Awareness Architecture (MASAA) for autonomous systems in a Bayesian framework. The architecture integrates different and interrelated aspects of the agent's self-aware functionalities, from the sensory stimuli to the actuation models. We integrate Dynamic Variational Autoencoders (VAEs) with lower dimensional World Model (WM) vocabularies to represent the Situation Model as Multi-Modal Perception (MMP). The integrated WM and MMP knowledge is used as a First-Person Model in the decision-making stage through Multi-Agent Coupled Markov Jump Particle Filters (MAC-MJPFs).

MASAA integrates models to enhance the generalizability and interpretability of a data-driven model by building a Generalized state in Hierarchical Dynamic Bayesian Networks (H-

DBNs). The approach is motivated by Lecun's general AI (LeCun, 2022), Seth's Controlled hallucination theory (Seth, 2021), Friston's Active Inference (Friston et al., 2015) and the self-awareness framework (Regazzoni et al., 2020). These H-DBNs are multi-level representation, inference, and decision-making networks learned from video, odometry, and action state variables combined hierarchically. Each network represents different aspects of the generative model. The World Model (WM) is a lower dimensional vocabulary built through the odometry and action generalized state DBNs that are low dimensional.

The odometry generalized state contains the generalized coordinates, which represent a set of parameters specifying the dynamics of an agent as a time derivative of its state, in addition to the state itself. From the positional stationary points, we apply the Null Force Filter (Iqbal et al., 2021) to have first-order derivative (velocity) information that governs the motion dynamics of an agent. These four-dimensional Generalized States are combinations of exteroceptive and proprioceptive information, further processed by a Modified Growing Neural Gas (M-GNG) algorithm (Fritzke, 1994; Iqbal et al., 2019a) to obtain a semantic category (cluster), thus creating a hierarchical DBN model called Odometry Vocabulary Network.

The second WM vocabulary is the Action Vocabulary Network, learned from the output of a controller having continuous throttle, steering angle, and brake information. The Action Vocabulary Network is explicitly proprioceptive information. As in the Odometry Vocabulary Network, the sequences of actions are further processed through the M-GNG algorithm, generating the Action Vocabulary Network. The Odometry and Action Vocabularies guide the learning of the Video Vocabulary Network through a probabilistic attention vector. The Video Vocabulary Network is explicitly exteroceptive information.

Two different attention vectors train the Video Vocabulary network using a Cluster-Guided Kalman Variational Autoencoder (CG-KVAE). These vectors are probabilistic distance measures assigning corresponding video latent information to the unsupervised Odometry and Action Networks, linking motion-component video latent states with the WM vocabularies. The content-related VAE encoding outputs are used for incremental learning purposes, since they can discriminate anomalies in the external environment. These video vocabularies are the learned representations that constitute the MMP model.

Integration between the generated interrelated knowledge should be abstracted to drive the perceptual and decision-making processes simultaneously. This approach has to ensure the prediction of the optimal actions that minimize the prediction errors in the perceptual setup, according to the active inference framework. The central idea behind active inference is to jointly minimize Variational Free Energy (VFE) and Expected Free Energy (EFE) (Smith et al., 2022). Based on this concept, we build an auxiliary higher-level Network in the

H-DBNs model that integrates these three (Odometry, Action, and Video) vocabularies. This dictionary abstracts the label-based relations between each vocabulary network.

We hypothesize that achieving general artificial intelligence for specific tasks across diverse situations is possible through the dynamic fusion of multi-sensory information within a self-awareness framework according to the principle of free energy minimization. We propose MASAA and validate it with the active inference framework. In active inference perception and learning can be understood by minimizing a quantity known as variational free energy (VFE). Action selection, planning, and decision-making can be understood by minimizing the expected free energy (EFE), which quantifies the VFE of various actions based on expected future outcomes. This framework should account for the generative nature of developed models across various situations, driven by the specific goals that an artificial agent aims to achieve.

Developing a general cognitive self-awareness model is an immense process that can learn from different experiences and perform tasks in an explainable way. This thesis aims to build Integrated learning and decision-making models for a particular self-driving agent in a novel self-awareness architecture. The fusion of multi-sensorial data to learn representation and inference models that rule the incremental learning process from self-action is developed. Understanding self-state (self-awareness) and robustly interacting with the environment (situation awareness) are modeled to test and assert this hypothesis. This thesis aims to answer the following research questions:

1. Which data-driven framework can be developed based on self-awareness models?
2. How can an artificial agent be as close as self-aware?
3. Can an agent learn explainable decision-making models?
4. How does an agent measure self-aware model predictions and decisions?

The main contributions of this thesis are summarized in table 1.1.

Table 1.1 Summary of the contributions of this thesis. Visual summary of the developed systems are presented in Chapter 8, fig. 8.1: stage 1 includes the lower dimensional odometry and action vocabulary learning; stage 2 the higher dimensional vocabulary learning; and stage 3 the incremental online active learning stage.

Chapters	**Contribution Summary**
Chapter 6	A data driven approach in a Bayesian framework is learned to localize interacting agents. Learned multi-sensorial modalities, i.e., odometry and video are represented as MAH-DBN. A modified MJPF that integrates multi-agent video and odometry vocabularies is developed.
Chapter 7	Introduces a novel cognitive architecture (MASAA) where all it's components learn hierarchy of representation. Bayesian generative and predictive models for a generalizable overtaking interaction is developed. We build the best dynamic coupling between experience modeling from higher dimensional sensor information, and attention focusing from lower dimensional sensor information, to minimize the prediction error while doing joint tasks (self-localization and interaction).
Chapter 8	We consider the learning and decision-making processes in the active stage using Coupled Markov Jump Particle Filters to model perception and action selection procedures through VFE and EFE. We model both continuous and discrete action states. Action representation, inference, and selection developed as a process of inference where an agent selects actions by inferring a distribution over control states by using particle filters (estimating discrete action states) and Kalman filters (inferring continuous action states). Online incremental learning is developed utilizing signals from anomaly detection and characterization.

1.2 Outline and contributions

In autonomous agents, being aware of oneself and understanding the external environment in sequential state estimation and interaction modeling is a challenging domain. The dynamic change in the environment makes it very difficult to generalize situations (different maneuverings), as we do not perfectly know the environment. In this thesis, from Chapter 2 to Chapter 4, we present background studies that are the basis of this thesis. In these chapters, we will

also assess state-of-the-art methods. The second part of the thesis, Chapter 6 to Chapter 8, describe the work that has been presented in three publications, based on the novel models developed for the integrated learning and decision-making processes in autonomous systems.

Chapter 2 briefly discusses the background of self-driving cars. It then discusses state-space models for autonomous systems, by defining the Generalized state-space models and how they represent the motion dynamics of a physical agent in a generative model. These generative models are probabilistic models integrating random variables and latent variables to represent the generative process as a hidden structure of the data that the generative process created. We will distinguish and briefly discuss the relations between the generative process and the generative model. We will describe dimensionality reduction and different latent variable models.

Chapter 3 provides an in-depth analysis of representation and inference frameworks. We will introduce models for sequential data representation and inference, and discusses how the generative model is represented and inferred in H-DBNs. The main discussion points covered are Bayesian Networks, Dynamic Bayesian Networks, State estimation models, Clustering and Anomaly Signals.

Chapter 4 presents the integrated learning and decision-making self-awareness model. We will briefly analyze and describe self-awareness and self-awareness architectures. In this chapter we also discuss learning algorithms that can realize the self-awareness models. Reinforcement Learning, Imitation Learning and the Active Inference framework will be introduced.

Chapter 5 describes how the experimental datasets are generated and used. It further shows the integration process for a unified representation of different experiences of autonomous agents. The datasets contain public, private, and simulation-based data. It will cover the synchronization process used to accommodate the sequential nature of models.

Chapter 6 presents the work discussed in (Alemaw et al., 2022), which introduces multi-modal interaction for collaborative agents modeling overtaking interaction using a relative distance vector learned from the expert trajectory. We model the interaction of video and odometry modalities by assuming that they share sensorial information between them. The goal is to estimate the Generalized States of an ego agent and the relative state of an interacting agent.

Chapter 7 discusses the work presented in (Alemaw et al., 2025) that introduces Multi-Agent Self-Awareness Architecture (MASAA) for Autonomous agents. We build a general self-awareness architecture demonstrated to handle multi-agent and single-agent knowledge representation and inference networks.

Chapter 8 presents the work discussed in (Alemaw et al., 2024) which models the incremental capability of the MASAA architecture, when the target distribution of the learned environment is new. It incorporates online incremental learning and decision-making in the Active Inference framework.

Chapter 9 finally summarizes the main contributions of this thesis. It also paves the way for open questions and potential research paths.

Chapter 2

Self-Driving and State-space modeling

Autonomous driving is an area that focuses on enabling to operate independently, without the intervention of a human driver. How simple it is to drive an automobile and a limited set of possible actions to drive a car fascinate people and inspire the idea of transferring these capabilities (accelerate, steer, and brake) to an algorithm, which seems a less complicated task.

However, autonomous driving is a complex task that demands a complete awareness of dynamic situations in the order of microseconds. Driving autonomously should possess a cognitive capacity, rich sensory input, and robust algorithms. These capabilities are challenging and complex to model in artificial agents (Janai et al., 2021; Andreas Geiger, 2022). In this chapter, we will discuss the history of Self-Driving, the challenges, ways of representing them, and possible approaches to solving them.

2.1 History of Self-Driving

The first remotely operated car was introduced in 1925 by Houdina Radio Control. Houdina Radio Control first demonstrated the radio-controlled "American Wonder" on New York City streets (The Evening News, 1925). This invention uses remote human drivers, which is far from self-driving.

After several decades, in the 1960s and 1970s, autonomous driving efforts focused more on vehicles that could move with the help of devices or electronic equipment embedded in the roads (Wikipedia contributors, 2024). It requires permanent installation of electromagnetic equipment in the road network, which differs from today's technology, primarily using different sensory inputs to build a self-driving model inside the agent.

In the 1980s, the interest in autonomous navigation based on installed equipment diminished, and the focus shifted to vision-based systems. Navlab (Wallace et al., 1985) and

ALVINN (Pomerleau, 1988) were projects developed to use this technology. Navlab digitizes road images, calculates deviation from the center line, and steers the vehicle autonomously. ALVINN used a neural network to perform road following based on front camera images and a laser range finder.

Around 2000, the introduction of GPS and IMU technologies accelerated the development of autonomous vehicles through their integration. Models using these technologies become more accurate in locating a vehicle. GPS determines the position of a car worldwide, while the IMU determines the velocities. These technologies have a significant impact on today's autonomous navigation system development. Following these advancements, the Darpa Grand Challenge (Buehler et al., 2007) was launched in 2005 to motivate researchers to work in groups and advance autonomous driving, and five teams managed to end the route by driving autonomously. In this competition, one team introduced LiDAR technology to scan the environment 360 degrees, and then LiDAR sensors started to be manufactured by Velodyne LiDAR technology company for commercial use. These sensors transformed the vision-based self-driving systems, making them more powerful.

As technology progresses, major technological corporations begin to investigate this field. From 2009 to 2015, many companies promised to commercialize and become pioneers in providing commercial services. Companies like Google, Volkswagen, Mercedes-Benz, Tesla Motors, and other university-based institutions were actively engaged in self-driving projects.

The technological advance in Machine Learning, particularly in Deep Learning (LeCun et al., 2015), revolutionized problem-solving methods in different research areas. Deep Learning brought a significant advantage to autonomous driving by improving its performances. DL attracts many researchers to work in the self-driving research area. Hardware advances, especially the advancement of GPU, revolutionized Deep Learning based approaches. The current self-driving research approach is towards advancing GPU-based Deep Learning methodologies.

2.2 Levels os Autonomy in Self-Driving

The Society of Automotive Engineers (SEA) (On-Road Automated Driving (ORAD) Committee, 2016) categorizes the levels of autonomy into six levels of driving automation, ranging from no driving automation (Level 0) to full driving automation (Level 5):

- **Level 0:** No Driving Automation
- **Level 1:** Driver Assistance
- **Level 2:** Partial Driving Automation

- **Level 3:** Conditional Driving Automation
- **Level 4:** High Driving Automation
- **Level 5:** Full Driving Automation

From Level 0 to Level 2, the driver needs to monitor the vehicle. From Level 3 to Level 5, the car is considered a non-monitored driving (Janai et al., 2021; Fernandez-Matellan et al., 2024; Andreas Geiger, 2022).

To expand further, Level 0 is a vehicle driven by a human driver. Level 1 is Driver Assistance Systems (DAS). Level 2 is Partial Automation, where the car performs lateral and longitudinal actions independently for a short period. The human driver can temporarily remove his hands from the steering wheel while remaining alert to take over. Level 3 is Conditional Automation. The capabilities are similar to those of Level 2, and in addition it includes a performance assessment capability to request the human driver to resume control. Level 4 is High Automation. Vehicles at this level can asses all situations and can act independently. This system is termed Eyes Off and Hands Off since it can operate independently in well-defined scenarios. In new situations where the environment is not well-defined, the human driver must actively assess the situation and take over the driving task. Level 5 is the highest and the last level according to SEA (On-Road Automated Driving (ORAD) Committee, 2016), which is FULL Automation. The system can perform the driving task entirely without the intervention of a human driver.

2.3 State-space models

In autonomous driving, a driving system has to deal with many sensors from low-dimensional to high-dimensional sensors to determine the state of itself and the the surrounding environment. The sensory inputs can be collected from radar, GPS, LiDAR, or Video sensing modalities. These sensory data need to be further processed according to the tasks that the autonomous agent is tackling.

2.3.1 Definition

A State space is the set of all possible configurations of a physical system state, where each state of the system corresponds to a unique point in the world where the physical system exists (Kalman, 1960; Terman and Izhikevich, 2008; Bernhard and Deschamps, 2019).

Based on the represented physical system, a state space could be finite, finite-dimensional, or infinite-dimensional. If it is finite it can be expressed in fewer points. If it is finite-

dimensional, can be a smooth manifold consisting of infinite number of points, which is often called a phase space. If it is infinite-dimensional, which allows the system to be represented with infinitely many degree of freedom. In this work, we only consider the finite-dimensional state spaces. We refer to degree of freedom as an expression regarding the dimension of the system phase space, that is the number of variables needed to completely describe the system.

In autonomous driving, we are interested in mapping higher-dimensional state space sensory input to low-dimensional action states like steering, accelerating, and braking for a finite-dimensional object (cars) having finite-degree of freedom. More specifically, we are interested in optimal state estimates of an autonomous agent that can be expressed using first and second order statistical estimations (Kalman, 1960).

2.3.2 Generalized state-space models

The evolution of a dynamical system corresponds to a trajectory in space. Initial states determines various trajectories, as different starting points result in different trajectories. When a state of a dynamical system is specified by a single or one-dimensional $x \in \mathbb{R}^1$, it corresponds to physically meaningful states of the system expressed in terms of time intervals. One dimensional systems are characterized by the ordinary differential equation (ODE):

$$x' = f(x) \tag{2.1}$$

where $x' = d_x/d_t$, is the derivative of the state variable x with respect to time t. One-dimensional systems can also be represented by the iterated mapping from the previous time instant state, as it can also be seen in fig. 2.1

$$x_{t+1} = f(x_t) \tag{2.2}$$

where the state at time $t+1$ is a function of the state at time t. Phase planes typically arise in the context of two-dimensional ODEs, which can be written:

$$x' = f(x,y) \tag{2.3}$$

where $f(x,y)$ is a smooth function. The two general variables can describe, for example position and velocity of an object as in fig. 2.2. The velocity vector of the trajectory $[x(t),y(t)]$ at a point p, e.g., $[x(p),y(p)]$ is given by (x',y'), which is the first order derivative of the position vector. The positional information is the observable variable and the velocity information is latent state variable that model the state space of a dynamic object. The

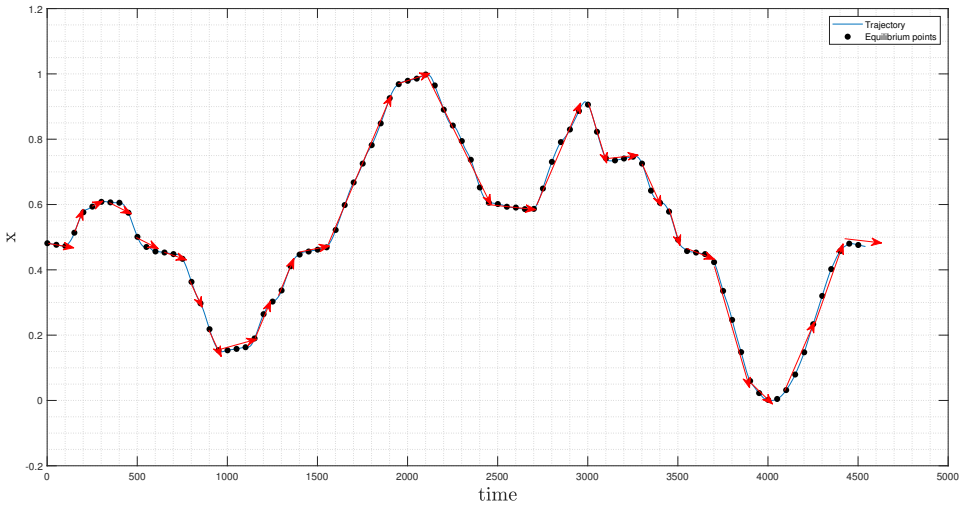


Figure 2.1: Single dimensional trajectory depicted across time.

iterated mapping for these generalized variables can be given by:

$$x_{t+1} = f(x_t, y_t) \tag{2.4}$$

State space models (SSMs) are a class of probabilistic graphical models (PGMs) that describe the probabilistic evolution of latent state variables and the observed measurements (Koller and Friedman, 2009). A SSM provides a general framework for analyzing deterministic and stochastic dynamical systems. Hidden Markov models (HMMs) and latent process models are other terms to describe SSMs. Kalman filter (Kalman, 1960) is the most powerful and well-studied SSM that defines an optimal algorithm for inferring linear Gaussian systems.

State space modeling is a process of learning a pattern of a generative process, to optimally estimate the hidden states given the observed sequences of data, using Bayes's theorem (Bayes and Price, 1763). In a Bayesian approach, generalized state space formulation, the state and cumulative observations up to time t, the posterior probability distribution of the state (z_t)

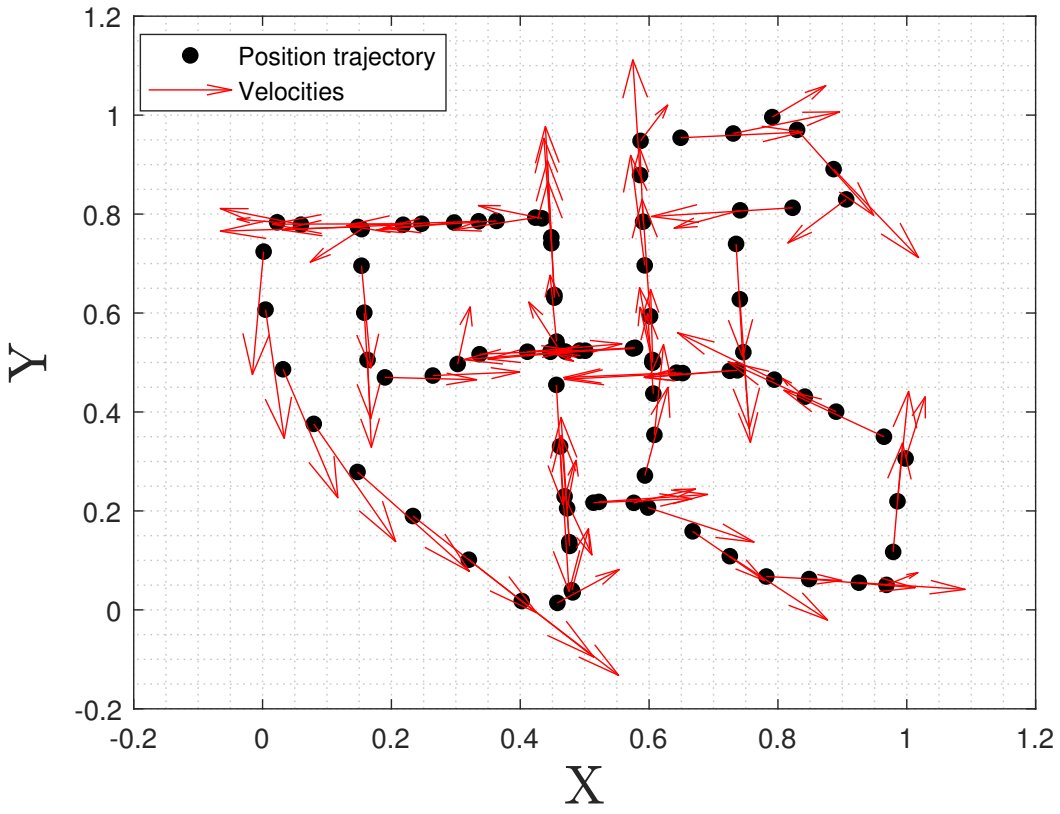


Figure 2.2: Multi dimensional trajectory representing generalized states.

conditioned on the observations are given by:

$$\begin{aligned} p(z_t|x_{0:t}) &= \frac{p(z_t|x_{0:t})}{p(x_{0:t})} \\ &= \frac{p(z_t|x_{0:t-1})p(x_{0:t}|z_t,x_{0:t-1})}{p(x_t|x_{0:t-1})} \\ &= \frac{p(z_t|x_{0:t-1})p(x_t|z_t,x_{0:t-1})}{p(x_t|x_{0:t-1})} \end{aligned} \tag{2.5}$$

where z_t denote the state and $x_{0:t}$ denote the cumulative observations up to time t. Eq. (2.5) assumes conditional independence between each observation. The one-step state prediction can be given by:

$$p(z_t|x_{0:t-1}) = \int p(z_t|z_{t-1})p(z_{t-1}|x_{0:t-1})dz_{t-1} \tag{2.6}$$

where $p(z_t|z_{t-1})$ is the state transition and the observation equation is denoted as $p(x_t|z_t,x_{0:t-1})$. Eqs. 2.5 and 2.6 are the fundamental relations in Bayesian state space modeling.

2.4 Generative models

Generative models from the Bayesian state modeling perspective, are learned or formulated representations abstracting a relation of how observations (sensory inputs) are generated by objects or events, outside of the physical system, that can not be directly known (Koller and Friedman, 2009; Smith et al., 2022). Each generative model contains a set of possible hidden states (z), priors over those states $p(z)$, a set of possible observations or outcomes (x), and a likelihood specifying how hidden states generate observations $p(x|z)$. Inference can be seen as a model inversion, since updating beliefs from prior to posterior beliefs is a way of inverting the likelihood from $p(x|z)$ to $p(z|x)$. It is a way of mapping causes to consequences and to use it to infer causes from consequences.

Models can include multiple sets of states representing different state spaces, which generate sets of observable outcomes. In multi-state modeling, the generative model is defined in terms of joint distributions $(p(x,z))$, which is a probabilistic distribution over all the possible combinations of states and observations. As the set of states and observations increase, the dimensionality of the state will increase. This factors affect the system to be computationally intensive.

In state space modeling, the generative process and the generative model are distinct in that the generative process is the actual (veridical) process that describes the causes of sensory inputs. The generative model is a belief-based process which might be a suboptimal process in representing the generative process.

2.5 Dimensionality Reduction and Latent Variable models

State space modeling has to deal with spaces of high dimensionality consisting of many input variables. Let us consider video data that are inputted to a self-driving car at each fraction of seconds. The car has to transform the image input to control signal. This involves processing thousands of voxels and outputting a steering, acceleration and braking data. This process is computationally intensive. The use of raw images in Bayesian approach requires the generation of many predictions and updates to estimate states. In probabilistic models, the central idea is to evaluate the posterior distribution $p(z|x)$ of the latent variables z given the observed variables x, which practically is cumbersome to evaluate or compute expectations with respect to this distribution (Bishop, 2006). In this case, dimensionality reduction is a mandatory preprocessing step to execute any action (task).

Dimensionality reduction is the process of converting high dimensional input data to a low dimensional representation, such that similar input features are mapped to nearby points

governed by a specific mathematical rule between samples in the input space (Hadsell et al., 2006).

2.5.1 Autoencoders

Autoencoders are artificial neural networks introduced to learn hidden state representations instead of directly mapping input layers to output layers Rumelhart et al. (1986). Their main characteristic is to compress higher dimensional input data into a lower dimensional latent state, capable of generating the input data with minimum loss. From the family of autoencoders, sparse, denosing, and contractive autoencoders are used effectively in representation learning. They are used efficiently for classification problem in machine learning (Vincent et al., 2010; Hinton et al., 2011; Baldi, 2011). Autoencoders and their variants can compress linear or nonlinear datasets efficiently, which makes them very attractive for many problem domains. As it can be seen in fig. 2.3, the encoder reduces the input data to a few key features as latent state (called bottleneck). The bottleneck layer holds these lower dimensional features. The decoder layer reconstructs the original input data with minimum loss.

Training an autoencoder focuses on minimizing the difference between the original data and the reconstructed version of the original data. Let us say we are training an autoencoder for image data. The aim is to improve the ability of the decoder to accurately reconstruct the original data from the encoded bottleneck (latent) layer. The accuracy can be measured by using loss functions like the Mean Square Error (MSE) between each pixel of the original (x) and the reconstructed $(\tilde{x})$ image data:

$$\mathscr{L}(x,\tilde{x}) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \tilde{x}_i)^2 \tag{2.7}$$

where N is the total number of pixels of the image. The smaller the error, the better the representation and inference network, since the learning process progresses by minimizing this error.

In autoencoders, since the latent state is compact and not well-organized, new content generation is very noisy and most of the time meaningless. This is because autoencoders are trained to encode a data in a latent space for a sole purpose of reconstructing them with minimum loss, without considering the latent space organization. To enable new content generation with a regularized latent state as Variational autoencoders were introduced (Kingma and Welling, 2013; Rezende et al., 2014), explained in detail in section 2.5.3 after discussing variational inference 2.5.2.

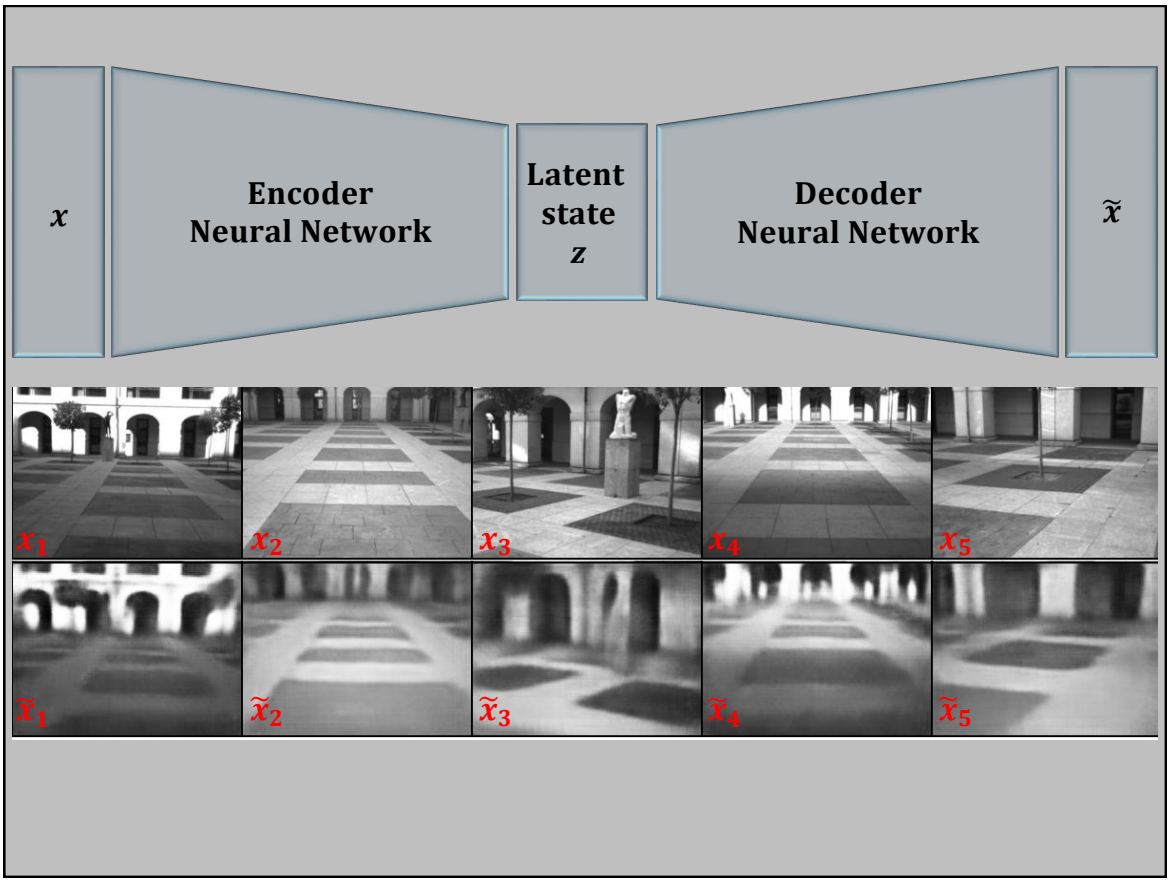


Figure 2.3: Autoencoder

2.5.2 Variational Inference in Latent Variable Models

One of the core problems of modern statistics is approximating difficult-to-compute probability densities. This problem is a fundamental issue in Bayesian statistics, which frames all inferences about unknown quantities as calculations about the posterior (Blei et al., 2017). Modern Bayesian statistics relies on variational methods to approximate this type of distributions. Variational inference is an approximation process of an intractable posterior distribution to perform a model inversion that can be solved in a computationally efficient manner (Smith et al., 2022). It is accomplished by introducing an approximate posterior distribution through simple assumptions (for example the independent and identically distributed (IID) assumption of latent states) to resemble the true posterior distribution $p(z|x,\theta)$:

$$p(z|x,\theta) = \frac{p(x,z|\theta)}{p(x|\theta)} = \frac{p(x|z,\theta)p(z|\theta)}{\int p(x,z|\theta)dz} \tag{2.8}$$

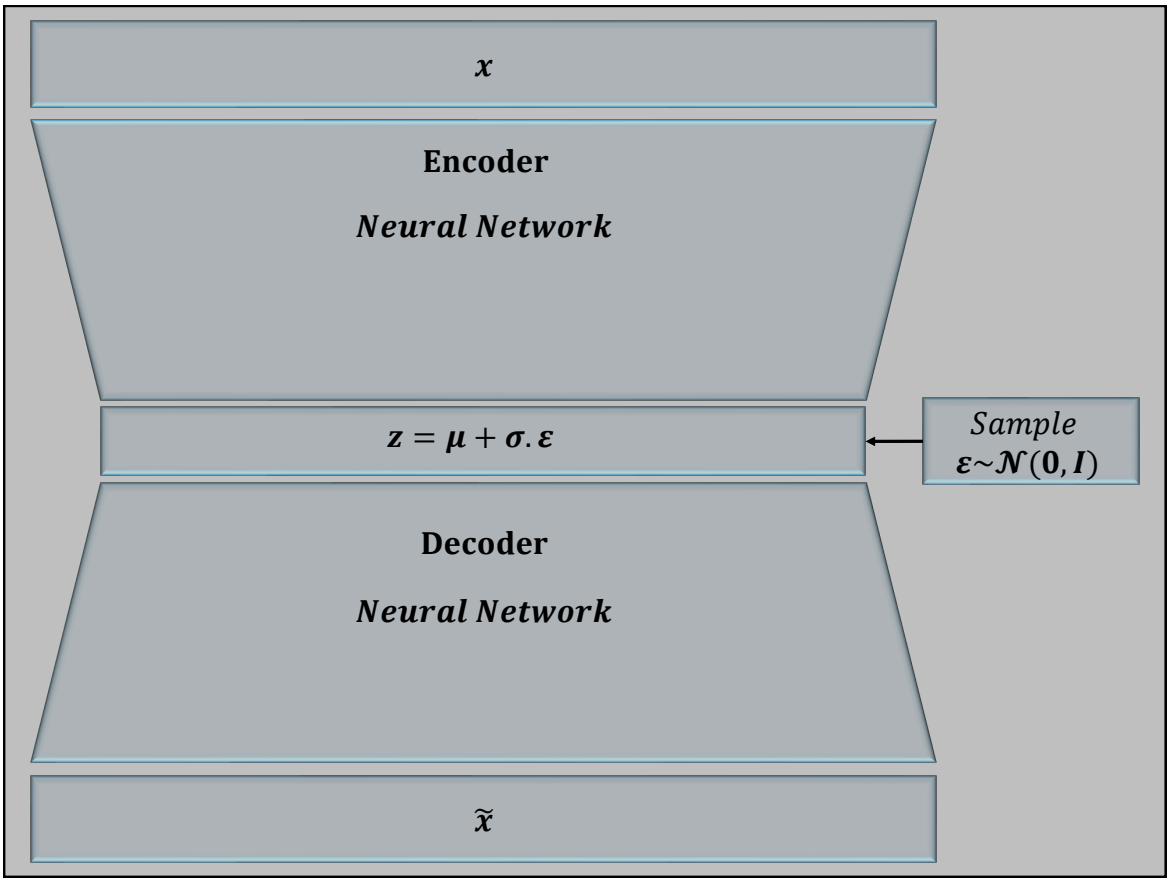


Figure 2.4: Variational autoencoder (VAE)

The posterior is factorized into the generative model ($p(x,z|\theta)$) and the data likelihood ($p(x|\theta)$) using Bayes rule. Looking the normalizing term ($\int p(x,z|\theta)dz$), as the dimensionality of the latent state (z) increases, the integral in continuous or the sum in discrete (as the integral becomes a sum) state distributions become cumbersome to compute.

To estimate the data from the latent state we can start from the notion of Variational Inference:

$$\begin{aligned} p(x|\theta) &= \int p(x,z|\theta)dz \\ &= \int p(x|\theta)p(z|x,\theta)dz \\ &= \int p(z|\theta)p(x|z,\theta)dz \end{aligned} \tag{2.9}$$

In eq. (2.9) since the latent state is influencing the data to be estimated, z needs to be marginalized. Let us introduce an approximate distribution $q(z)$ over the latent space:

$$\begin{aligned}\ln p(x|\theta) &= \int q(z) \ln \left(p(x|\theta) \frac{q(z)}{q(z)} \right) dz - - - - - - - - - - - - - (1) \\ &= \int q(z) \ln \frac{p(x,z|\theta)}{q(z)} dz + \int q(z) \ln \frac{q(z)}{p(z|x,\theta)} dz - - - (2)\end{aligned} \tag{2.10}$$

In eq. (2.10), it uses the property of log-likelihood as $\ln p(x) = 1 \times \ln(p(x) \times 1)$. Since $q(z)$ is a density, $\int q(z)dz = 1$ and $q(z)/q(z) = 1$. Eq. (2.10) line (1), is basically a cleaver expansion of $p(x|\theta)$. The second line is using an identity $p(x) = \frac{p(z,x)}{p(z|x)}$, i.e., the marginal distribution is the same as the joint distribution over the posterior distribution. The second term in eq. (2.10) line (2), represents the KL divergence:

$$KL(q(z) \| p(z|x,\theta)) \geq 0 \tag{2.11}$$

where KL divergence is a similarity measure which is always greater or equal to zero (0 if $q(z) \equiv p(z|x,\theta)$).

Considering the first term in eq. (2.10) line (2), it is noticeable that $\ln p(x|\theta)$ does not depend on $q(z)$. Hence, $\ln p(x|\theta)$ is constant with respect to $q(z)$, this implies $\ln \frac{p(x,z|\theta)}{q(z)} dz \leq \ln p(x|\theta)$. This is due to the non-negative nature of the KL divergence in eq. (2.11), whatever the value of $\ln \frac{p(x,z|\theta)}{q(z)} dz$ is, we need to add the KL term to get to $\ln p(x|\theta)$.

$$\begin{aligned}\int q(z) \ln \frac{p(x,z|\theta)}{q(z)} dz &= \mathbb{E}_{q(z)}[\ln p(x,z|\theta)] + H(q) \\ &= \mathscr{L}_{ELBO}(q,\theta)\end{aligned} \tag{2.12}$$

where $H(q)$ is the entropy. $\mathscr{L}_{ELBO}(q,\theta)$ is the evidence lower bound or variational lower bound because it is lower than the log-likelihood of the data ($\ln p(x|\theta)$). It also called variational free energy (VFE).

The introduction of $q(z)$ allows us to approximate the true posterior distribution ($p(z|x,\theta)$). One way to achieve this would be to minimize the KL divergence (see eq. (2.11)). The minimization of this function with respect to the approximate distribution $q(z)$ is a variational approach:

$$\begin{aligned}\ln p(x|\theta) &= \mathscr{L}_{ELBO}(q,\theta) + KL(q(z) \| p(z|x,\theta)) \\ \Rightarrow \underset{q(z)}{\operatorname{argmin}} KL(q(z) \| p(z|x,\theta)) &= \underset{q(z)}{\operatorname{argmax}} \mathscr{L}_{ELBO}(q,\theta)\end{aligned} \tag{2.13}$$

The main point in eq. (2.13), is thus minimizing the KL divergence with respect to $q(z)$ is implicitly maximizing the ELBO, which gives a flexibility to choose to maximize the ELBO or to minimize the KL, according to the situation or the model that we are developing.

The ELBO does not explicitly contain the posterior, but can implicitly approximate it:

$$\begin{aligned}\mathscr{L}_{ELBO}(q,\theta) &= \int q(z) \ln \frac{p(x,z|\theta)}{q(z)} dz \\ &= \int q(z) \ln \frac{p(x|z,\theta)p(z)}{q(z)} dz \\ &= \mathbb{E}_{q(z)}[\ln p(x,z|\theta)] - KL(q(z)\|p(z))\end{aligned} \tag{2.14}$$

where the expectation $\mathbb{E}_{q(z)}[\ln p(x,z|\theta)]$ of the generative model states how z influences x, which can be handled with a neural network; $q(z)$ is the approximate distribution that we already have, and $p(z)$ is the prior which can be chosen randomly. Hence, both the the expectation and the KL divergence do not require the posterior distribution which has a flexibility due to the duality of eq. (2.13) to choose one of the approximation strategies.

Neural Network based architectures, for example Variational auto encoders, can learn both $p(x|z,\theta)$ and $q(z|x,\phi)$ with parameters θ and ϕ for x and z. The output can be for example $q(z|x,\phi) = \mathscr{N}(z|\mu_\phi(x),\Sigma_\phi(x))$, i.e., Gaussian distributions with nonlinear dependence on the input x. Then to get z, we can sample from this distribution.

2.5.3 Variational Autoencoders (VAE)

Variational autoencoders are a family of autoencoders in compressing data into lattent lower dimensional state, but in addition it regularizes the latent state representation to be a normal distribution that enables new content generation. Instead of encoding to a latent point like in autoencoders, VAE encodes its input as a standard normal distribution.

Let us consider an image raw data x which has a distribution $p(x)$, then VAE creates another lower dimensional latent state z with a distribution $p(z)$. $p(z|x)$ is the posterior distribution that the encoder layer creates. The decoder decodes the likelihood distribution from the encoded latent state $p(z|x)$. The latent distribution $p(z)$ is not exactly known, which makes the computation intractable. To solve this, (Kingma and Welling, 2014) come up with an idea to regularize the distribution by forcing the distribution to be a normal distribution $p(z) \sim \mathscr{N}(0,1)$. Still the true posterior $p(z|x)$ is not known, which needs to be approximated using the Gaussian distribution $q(z|x) \approx p(z|x)$, where $q(z|x) = \mathscr{N}(\mu,\sigma)$. The goal of training the encoder network of VAE is to estimate these parameters of the Gaussian distribution, and the decoder network reconstructs the original data from the sampled approximate distribution $q(z|x)$.

In VAE we have two objectives: good reconstruction and regularized latent state for new content generation. These two objectives in the VAE network are handled through a loss function:

$$\mathscr{L}(x) = \mathbb{E}_{q(z|x)}[\log p(x|z)] - KL(q(z|x), p(z)) \tag{2.15}$$

where $\mathbb{E}_{q(z|x)}[\log p(x|z)]$ is the data consistency term, also called reconstruction term, and $KL(q(z|x), p(z))$ is the latent space regularization, which is the Kulback-Leibler divergence (Li and Turner, 2016) between the encoded distribution ($q(z|x) = \mathscr{N}(\mu, \sigma)$) and a standard Gaussian ($p(z) = \mathscr{N}(0,1)$).

In the decoding process there is a sampling process which samples from $\mathscr{N}(\mu, \sigma)$ to the latent state z, which makes it difficult to back-propagate the loss in the network. The sampling process should be adjusted through a reparametrization trick. Instead of sampling from the $\mathscr{N}(\mu, \sigma)$, a random variable $\varepsilon \sim \mathscr{N}(0,1)$ can be introduced to handle the randomness outside of the network, which is scaled and shifted by the variance and mean of the approximate distribution ($z = \mu + \sigma \cdot \varepsilon$). This process makes the the back-propagation possible.

A deep neural network VAE, takes data point x_i as input and output the mean and diagonal covariance matrix of its corresponding Gaussian variational approximation, i.e $q_\phi(z_i|x_i)$ through parameter learning. The loss in eq. (2.15), can be calculated based on parameters ϕ and θ which are the encoding ($q_\phi(z|x)$) and decoding ($p_\theta(x|z)$) parameters of the VAE network.

VAEs performance may be suboptimal due to the approximation introduced. For practical and computational reasons the network is forced to maximize the ELBO, so we approximate the true posterior using the normal distribution which affects the tightness of the bound. Other challenges are conditioning VAEs to constraint the latent space to produce a specific data. To address this issues many variants of VAE are being studied. Conditional variational auto-encoder (CVAE) Sohn et al. (2015) introduced for class specific image generation, by introducing class level loss functions. The β VAE (Higgins et al., 2017) approach, introduces a tunable parameter to enhance reconstruction and disentanglement in higher dimensional images. For sequential data KVAE (Fraccaro et al., 2017) introduce two latent states by further disentangling the latent state in to content related and motion related latent states to estimate the motion dynamics of a bouncing ball having only positional information.

2.5.4 Dynamical Variational Autoencoders

VAEs were initially developed without including temporal correlation, meaning that each data vector and the corresponding latent vector were processed independently from other

sequences. Consequently, studies regarding VAEs were extended and applied to many complex Deep Bayesian Networks to model sequential data exhibiting temporal correlation. These Deep Dynamic Bayesian Networks embedded in the VAE framework are the so-called Dynamical VAEs (DVAEs), which are VAEs including a temporal model for sequential data. As described in (Girin et al., 2021), the DVAE model must contain two fundamental relationships. The first is decoding link where z_t is a parent of x_t which depicts the hierarchical nature of DVAE. The second is temporal link. At least one element in $z_{1:t-1}$ or in $x_{1:t-1}$ is parent to either z_t or x_t.

In processing higher dimensional data, this thesis uses the KVAE (Fraccaro et al., 2017) as a base for disentangling different latent states.

2.6 Summary and discussion

In this chapter we overviewed the history of self-driving vehicles (section 2.1). A steady advance in self-driving technology has been taking place over the past few decades. The current level of autonomy is between 3 and 4, and many big companies, particularly Waymo and Tesla, are advancing this technology (section 2.2). Further we mention the technology advancements in self-driving state space modeling. In section 2.3, section 2.3.2, and section 2.4 we have shown how the state-space models are modeled in generalized coordinate systems. Moreover in section 2.4, we have shown how a latent variable drives the observation and how this relation is modeled in a probabilistic modeling approach.

In state-space modeling, dimensionality reduction is a mandatory step for representation and inference purposes, as detailed in section 2.5. Artificial Neural Network based encoders and decoders for higher dimensional sensory input are also introduced in section 2.5.1 and section 2.5.3. These encoder decoder architectures are used in representation and inference of latent variable models as discussed in section 2.5.2. A careful structured neural network can map a latent variable (z) onto a distribution of the input data x with learnable parameter θ, which is the generative model ($p(x|z,\theta)$). These neural networks also have capacity to learn how to map the input data (x) on to a distributions over a latent variable z with parameters ϕ, that is the recognition model ($q(z|x,\phi)$). These parameters are learned through the process of back propagation of the loss function ($\mathscr{L}_{ELBO}(q,\theta) \equiv \mathscr{L}_{ELBO}(\phi,\theta)$).

Inference and representation models can be represented in causal effect models that are very intuitive and explainable models, and are called Dynamic Bayesian Networks. These models are the focus of the next chapter.

Chapter 3

Representation and inference models

A representation system exhibits clear semantics that are independent of the algorithms used to operate on the representations. A well-studied and structured representation system forms a clear boundary between knowledge and reasoning (Koller and Friedman, 2009). This boundary gives the flexibility to develop a general suite of algorithms that can be exploited for any model in self-driving, medical diagnosis, natural language processing, or cognitive telecommunications domains. It is also possible to improve this general model for a specific application domain without having to modify the reasoning algorithm constantly.

3.1 Probabilistic Graphical Models (PGMs)

The true state of the world is always challenging to know as observations are partial (only part of the world is observable). The noisiness of observations makes it very difficult to model the exact state of the environment. Further, having noise-free observations is not enough to determine the true state of the environment with certainty. The non-deterministic nature of the world, model limitations, and partial observability are the dominant factors of environmental uncertainty (Koller and Friedman, 2009).

Uncertainty forces reasoning systems to be probabilistic. Reasoning systems are models demonstrated by observations (measurements) that can happen because of many factors. Probabilistic models allow us to model complex systems with causal effect relations about the outcomes and the events that cause them. These systems have interrelated and dynamic components that can be represented by random variables, where the value of each variable defines a property of their state at that particular time. The probabilistic model's task is to reason out probabilistically the values of these variables conditioned on each other. In probabilistic models, these variables can be dealt with a joint distribution, enabling them to

estimate their posterior distributions. Probabilistic models are ideal to represent and describe this complex probabilistic relations.

Probabilistic graphical models are graph-based representation systems that are suitable for compactly encoding a complex distribution over a high-dimensional space (Koller and Friedman, 2009). Graphical representation models can be categorized either directed or undirected graphs. Directed graphs have parent and child (source and target) relationships. Bayesian networks are examples of directed representations having causal effect relationships. In the case of undirected graphs, they are graphs that do not have causal effect relations. Markov networks are examples of undirected graphs, representing interdependency rather than causality. This thesis exclusively uses directed graphs in a Bayesian setup.

A Bayesian representation provides transparent (explainable) models that build understandable semantics and properties, enabling an agent to generalize well in similar configurations. They also allow inference algorithms to compute posterior probabilities agent state spaces. The hierarchical nature of Bayesian networks provide a powerful model approximations by representing past experiences through data-driven approaches. Representation, learning, and inference for decision-making are underlying components in developing intelligent agents. Bayesian models are capable of handling knowledge in an explainable form.

3.2 Bayesian Networks

Bayesian networks are based on the Bayes model (Bayes and Price, 1763) which can be represented as a directed acyclic graph (DAG), where the nodes are random variables and the edges are links influencing one node on another in the direction of the link as in fig. 3.1. They satisfy the local conditional independence of child nodes given the parent nodes (Koller and Friedman, 2009).

Bayesian networks can represent direct, indirect, and common causal effects over a joint distribution considering a fixed set of random variables. Bayesian probabilistic models are effective in modeling fixed sets of variables. To deal with dynamic and complex state spaces in these networks, it is necessary to introduce an additional set of rules, that allows them to keep track of time series information. In a temporal setting, the environment must be represented by variables that account for temporal changes. Sensory readings should be linked to track changes in each random variable, which needs a single compact temporal model that provides a template for the entire class of distributions in the domain, which also handles dynamic changes. For these problems, Dynamic Bayesian Networks are ideal for modeling a single compact model that optimally represents the system's dynamics.

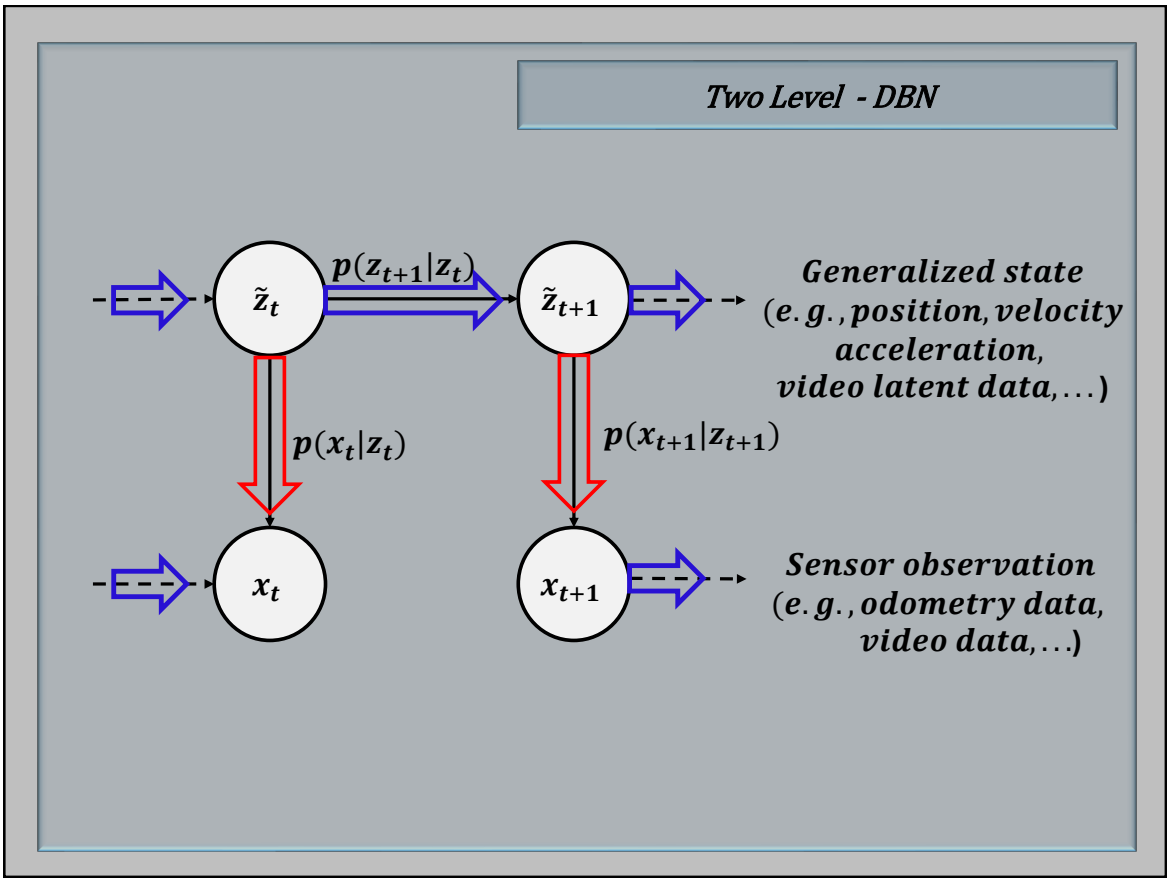


Figure 3.1: Dynamic Bayesian Network (DBN). A two-time-slice Dynamic Bayesian network (2TDBN), with two levels of hierarchy, consists of variables such as x_{t+1} and $\tilde{z}_{t+1}$ whose parents are variables $\tilde{z}_t$ and $\tilde{z}_{t+1}$.

3.3 Dynamic Bayesian Networks

Dynamic Bayesian networks are extensions of Bayesian networks based on the assumption that, if event a causes another event b, then event b can not cause event a (that is the directed acyclic assumption). Bayesian networks for time series are directed acyclic graphs that should follow forward in time. These networks generally operate on the Markov assumption that a random variable x_{t+1} at time $t+1$, directly depends only on the random variable x_t at time t. More specifically, it assumes every information, to estimate the next time step, is stored in the current time step, ignoring previous time step information until t.

In self-driving car state estimation, if we only have location and observation information, it is not possible to say that the location at $t+1$ is independent of the previous locations, given only the location at time t, as the previous locations contain the information about the direction of motion and the speed of the agent. However, if the velocity is also considered in

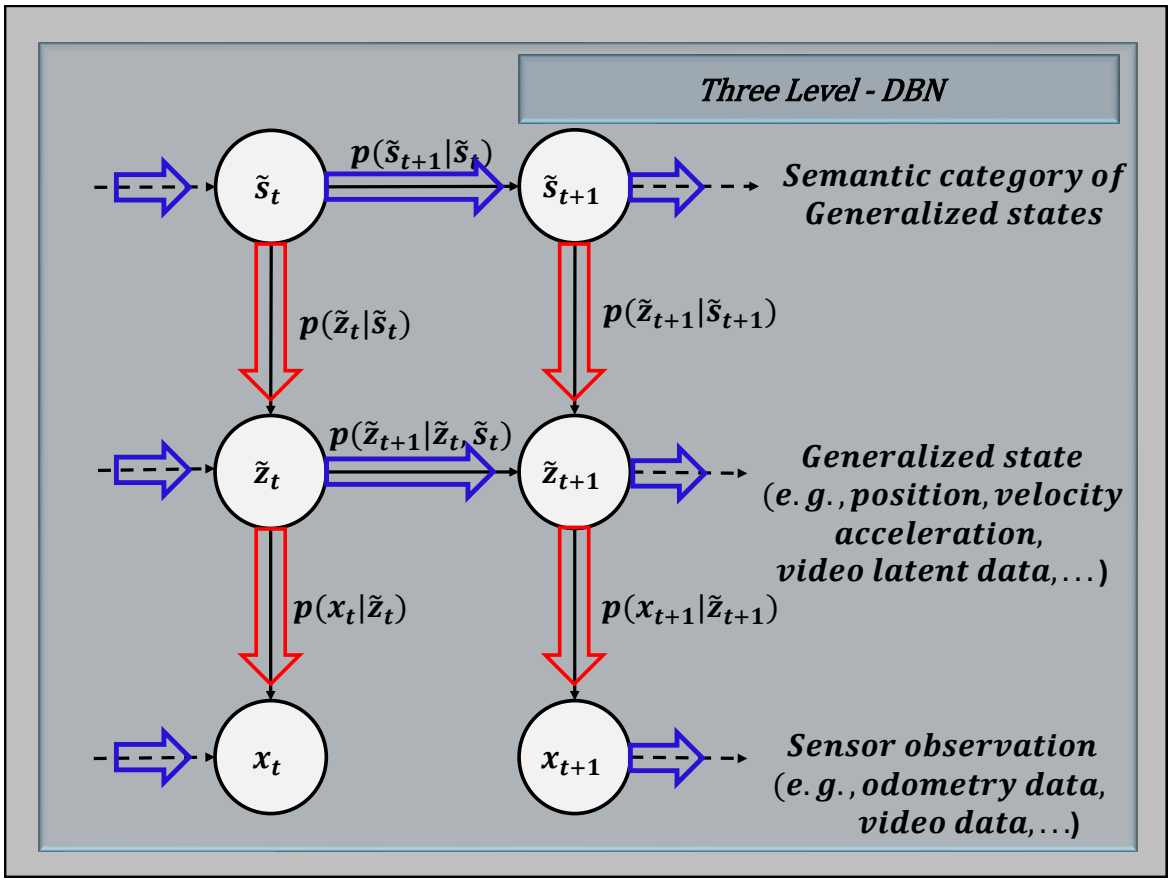


Figure 3.2: Hierarchal DBN (H-DBN). A two-time-slice Dynamic Bayesian network (2TDBN) with three levels of hierarchy. Blue arrows show inter-slice connections. Red arrows show intra-slice connections. Dotted black lines show time evolution in DBNs while solid black lines represent links for messages passing.

the random variable vector as a Generalized variable, the Markov assumption can be satisfied. Hence the introduction of Generalized variables into the state space model is a reasonable approximation of Markov assumption (Murphy, 2002; Koller and Friedman, 2009).

The Markov model can give a more abstract form to model the random variables as hidden states that generate the observations. The sequence of these generalized random variables (states) can follow a Markov process. Kalman filter (Kalman, 1960) is a model of this type that is optimal in linear-Gaussian state space models.

State estimation in DBNs can be performed in intra and inter-time slice message passing schemes (connections) (Chai et al., 2014). Intra-slice connections are the connections within time slices with different random variables, whereas inter-slice connections are the

connections between time slices related to the same random variables in consecutive time steps(see also figs. 3.1 and 3.2).

3.4 State estimation models

In sequential state-space models, a sequence of D-dimensional observation vectors $x_1, x_2, ..., x_T$, are modeled by assuming that each sequence vector at time step x_t, is generated from a K-dimensional hidden state random variable z_t. These sequences of observations can define a first-order Markov process (Beal et al., 2001) from time t to time T:

$$p(z_t, x_t) = p(z_1)p(x_1|z_1)\prod_{t=2}^{T} p(z_t|z_{t-1})p(x_t|z_t)) \tag{3.1}$$

The state transition probability $p(z_t|z_{t-1})$ can be changed into a function as:

$$z_t = f_t(z_{t-1}) + \omega_t \tag{3.2}$$

where f_t is a deterministic transition function determining the time evolution of the state z, and ω_t is a zero mean random noise vector. The observation probability can also be converted to a function as:

$$x_t = h_t(z_t) + \upsilon_t \tag{3.3}$$

The properties of f_t and h_t determine the state space model. If both functions are linear and time-invariant, and the distributions of the state and observation noises (ω_t, υ_t) are Gaussian variables, the model becomes a linear-Gaussian state space model:

$$\begin{aligned} z_t &= Az_{t-1} + \omega_t \\ x_t &= Hz_t + \upsilon_t \end{aligned} \tag{3.4}$$

where A is state transition matrix and H is the observation matrix. The Bayesian network corresponding to this model, shown in fig. 3.1, includes a sequence of nodes z_t, each of which is a parent of the corresponding x_t.

Different filtering methods can be applied in Bayesian state estimation, depending on the characteristics of the state being represented and inferred. For linear Gaussian state-space models, Kalman Filters (KF)are optimal. When the linear assumption is not holding anymore, an Unscented Kalman Filter (UKF) (Wan and Van Der Merwe, 2000; Julier and Uhlmann, 2004), Extended Kalman Filter (EKF) (Jazwinski, 1970), Particle Filter (Elfring et al., 2021), or a combination of these filters can be applied.

In this thesis, Kalman Filters, Particle Filters, and Coupled Markov Jump Particle Filters are used for inference and decision-making algorithms in a scenario with interacting autonomous agents. We will briefly discuss them in the following sections.

3.4.1 Kalman Filter

A linear dynamic system evolving linearly over time with Gaussian noise is called a Kalman Filter. This system is a DBN where all the variables are continuous and their dependencies are linear and Gaussian (Welch and Bishop, 1994; Koller and Friedman, 2009). Kalman Filters model the dynamics of moving objects given noisy measurements. The model is generally mathematically defined as:

$$\begin{aligned} p(z_{t+1}|z_t) &= \mathcal{N}(Az_t; Q) \\ p(x_{t+1}|z_{t+1}) &= \mathcal{N}(Hz_{t+1}; R) \end{aligned} \tag{3.5}$$

where z is an n-dimensional vector representing state variables, x is an m-dimensional vector of observation variables. A is an $n \times n$ matrix defining the dynamic (process or transition) model. Q is an $n \times n$ matrix defining the Gaussian noise associated with the system dynamics. H is an $n \times m$ matrix defining the linear observation model, and R is an $m \times m$ matrix defining the Gaussian noise associated with the observations.

In simple localization, considering lower dimensional data (position and velocity), z denoting an agent's current position in (x, y) dimension and a variable v representing its velocity in (x, y), first order approximation can be a modeled as: $p(z'|z, v) = \mathcal{N}(z + v\triangle; \sigma_z^2)$ and $p(v'|v) = \mathcal{N}(v; \sigma_v^2)$, where $\triangle$ is the length of time step.

In eq. (3.5), the state transition model can be modified if there is a linear control input u_t as:

$$p(z_{t+1}|z_t) = (x_t, u_t = u) = \mathcal{N}(Az_t + Bu; Q) \tag{3.6}$$

where B is $N_{\times}N_u$ matrix, N_u is the number of the control inputs, commonly it is a column vector.

The Kalman Filter model for a dynamic agent can be modeled in a Generalized State form, where $z_{t+1} = (x_{t+1}, y_{t+1}, \dot{x_{t+1}}, \dot{y_{t+1}})$ representing the position and velocity of an object. Considering a constant velocity:

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \dot{x_{t+1}} \\ \dot{y_{t+1}} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \triangle & 0 \\ 0 & 1 & 0 & \triangle \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_t \\ y_t \\ \dot{x_t} \\ \dot{y_t} \end{pmatrix} + v_t \tag{3.7}$$

where $\triangle$ is the sampling period, $v_t \sim \mathcal{N}(0,Q)$ is the noise, and Q is a covariance matrix:

$$Q = \begin{pmatrix} Q_x & Q_{x,y} & 0 & 0 \\ Q'_{x,y} & Q_y & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.8}$$

Q_x is the variance of the noise in the x direction, Q_y is the variance of the noise in the y direction, and $Q_{x,y}$ is the cross-covariance. $Q'_{x,y}$, the prime symbol is the transpose of the matrix. Eqs. 3.7 and 3.8 define the prediction step of the Kalman Filter algorithm.

The update step of Kalman Filter is based on exteroceptive observation, the position of the object, which needs to be transformed from $2D$ to $4D$ using the observation matrix H in eq. (3.5):

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \dot{x_{t+1}} \\ \dot{y_{t+1}} \end{pmatrix} + w_{t+1} \tag{3.9}$$

where $w_{t+1} \sim \mathcal{N}(0,R)$.

In the update step we correct the dynamic state based on the Kalman Gain:

$$\begin{aligned} K_{t+1} &= v_t H'(H v_t H' + R)^{-1} \\ z_{t+1} &= z_t + K_{t+1}(x_{t+1} - H z_t) \\ v_{t+1} &= v_t - K_{t+1} H v_t \end{aligned} \tag{3.10}$$

where K_{t+1} is the Kalman Gain, x_{t+1} is the observation, v_{t+1} is the velocity, and $z_{t+1} = [x_{t+1} \;\; v_{t+1}]'$ is the dynamic state variable.

Kalman Filter models assume the state to be uni-modal, since it is a linear state space modeling. However, in the real world, state spaces are more complex and do not only evolve linearly. Commonly state spaces include nonlinear behaviors and also non-Gaussian having Gaussian mixture models. The nature of complex state dynamics forces the representation and inference framework to use approximation models, to make them reliable systems. These models that have different dynamics, can be approximated to piecewise-linear systems, called switching linear dynamical systems, which will be discussed in section 3.4.3.

3.4.2 Particle Filter

Particle-based methods approximate the joint distribution as a set of instantiations to the variables in the model (Arulampalam et al., 2002; Koller and Friedman, 2009; Elfring et al., 2021; Murphy, 2023). These particles should be designed to fit well the representation of the overall probability distribution.

In recursive non-linear state space estimation, the main idea is to represent the posterior distribution by a set of random samples (or particles) that are associated with weights and estimates based on the samples. Mathematically, taking a random measure $\{z^i_{0:t}, w^i_t\}^{N_s}_{i=1}$ that characterizes the posterior pdf ($p(z_{0:t}|x_{1:t})$), where $\{z^i_{0:t}, i = 0,\ldots,N_s\}$ is a set of support points with associated weights $\{w^i_t, i = 1,\ldots,N_s\}$, and $z_{0:t} = \{z_k, k = 1,\ldots,t\}$ is the set of all states up to time t. The weights are normalized such that $\sum_i w^i_t = 1$. Using the weights the posterior density can be approximated as:

$$p(z_{0:t}|x_{1:t}) \approx \sum_{i=1}^{N_s} w^i_t \delta(z_{0:k} - z^i_{0:k}) \tag{3.11}$$

In eq. (3.11) the true posterior $p(z_{0:t}|x_{1:t})$ is approximated with a discrete weighted approximation. The weights can be chosen according to the importance sampling approach (Arulampalam et al., 2002; Doucet et al., 2000).

Importance sampling procedure is based on the assumption that $p(z) \propto \pi(z)$ ($\propto$ symbol, refers proportionality), where $p(z)$ is a probability density from which it is difficult to draw a sample, but $\pi(z)$ can be evaluated up to proportionality of $p(z)$. Taking $z_i \sim q(z), i = 1,\ldots,N_s$, as samples from the proposal function q, called importance density, a weighted approximation to the density p is given by:

$$\begin{aligned} p(z) &\approx \sum_{i=1}^{N_s} w^i \delta(z - z^i), \\ w^i &\propto \frac{\pi(z^i)}{q(z^i)} \end{aligned} \tag{3.12}$$

where w^i is the normalized weight of particle i. If the samples are drawn from the importance density $q(z_{0:t}|x_{1:t})$, then the weights are defined as:

$$w^i_t \propto \frac{p(z^i_{0:t}|x_{1:t})}{q(z^i_{0:t}|x_{1:t})} \tag{3.13}$$

In sequential state estimation, to approximate the future time step states $p(z_{0:t+1}|x_{1:t+1})$, from the current approximation $p(z_{0:t}|x_{1:t})$, we can have samples $z^i_{0:t+1} \sim q(z_{0:t+1}|x_{1:t+1})$ from the proposal function, by taking each of the existing samples $z^i_{0:t} \sim q(z_{0:t}|x_{1:t})$ and by

augmenting it with the new state $z^i_{t+1} \sim q(z_{t+1}|z_{0:t}, x_{1:t+1})$. Then the weight needs to be updated as:

$$w^i_{t+1} = w^i_t \frac{p(x_{t+1}|z^i_{t+1})p(z^i_{t+1}|z^i_t)}{q(z^i_{t+1}|z_{0:t}, x_{1:t+1})} \tag{3.14}$$

In sequential importance sampling, after some iterations, only one particle will have a weight close to 1 and all the other particles will have negligible weights (degeneracy problem). This implies that a large computational effort is used to update nonrelevant particles. Measuring the effective sample size indicates if there is a degeneracy problem:

$$N_{eff} = \frac{1}{\sum_{i=1}^{N_s}(w^i_t)^2} \tag{3.15}$$

where w^i_t is the normalized weight (eq. (3.14)). If N_{eff} is very small compared to N_s, there is a degeneracy problem. The degeneracy problem can be solved by resampling based on a threshold or through a good choice of importance density (Arulampalam et al., 2002; Elfring et al., 2021).

Particle filters require many samples to get a good approximation. Their efficiency can be improved by combining them with linear filters like Kalman Filter (section 3.4.1). The combination of particles and Kalman filters in a unified manner is suitable for switching linear dynamical systems. This combination can indeed handle discrete state estimation through particle filters and continuous state estimation using Kalman Filters. This integration is called the Markov Jump linear dynamical system (Markov Jump Particle Filter).

3.4.3 Markov Jump Particle Filter

In Sections 3.4.1 and 3.4.2, we briefly discussed state-space models for linear and nonlinear dynamic systems, respectively. Kalman Filters are optimal in linear Gaussian dynamical systems, whereas Particle Filters are good approximations in nonlinear dynamical systems. Since world states are a mixture of these dynamics, addressing them in a unified and robust manner improves the abstraction level. Bayesian hybrid representation and inference schemes are solutions for unified explainable hybrid systems. These hybrid models include discrete and continuous variables (Koller and Friedman, 2009; Baydoun et al., 2018).

A conditional linear Gaussian DBN (fig. 3.2), where the continuous variables have a conditional (piecewise-linear) Gaussian dynamics, and the discrete variables have no continuous parents, is called a switching linear dynamical system. Hence, changes in the discrete state cause the linear continuous dynamics to switch from discrete state to discrete state. The switching model is represented through switching variables that will have the

same value until there is a change (switch) in the discrete states. In this type of state space modeling, the Markov Jump Particle Filters (MJPFs) (Baydoun et al., 2018) are used for discrete and continuous state estimation.

MJPFs are state estimation models in H-DBN models through the combination of Particle Filters for the discrete states, and Kalman Filters for the continuous state estimations. For each particle estimating the switching variables, the Kalman estimation performs the prediction and update steps described in section 3.4.1. In MJPF models, the system can switch between a set of discrete modes. While within a fixed mode, the system evolves using standard linear Gaussian dynamics where the equations governing the dynamics in different modes are different. We can view this type of system as a DBN that introduces a discrete variable s that encodes the mode, allowing s to be the parent of the continuous variables (see fig. 3.2).

The third-level variable representing the discrete states is commonly created considering a learning algorithm that can cluster the trajectory of a dynamic agent based on probabilistic distances. In this thesis, we applied an unsupervised clustering algorithm to define higher-level variables. In the following section, we will briefly discuss clustering algorithms.

3.5 Unsupervised Clustering

Unsupervised clustering is a process that allows an agent to learn similar structures (features) from data. These features should give a concise view of the data having certain common semantics. Clustering simplifies the large number of distinct sequences of points defining trajectories, by classifying them into a reduced number of categories, having different trajectories globally. Commonly clustering methods are based on distance metrics like the Euclidean distance. The process involves an iterative strategy to group similar observations. From the family of unsupervised clustering, we introduce the self-organizing map (SOM) (Kohonen, 1998) and the Growing Neural Gas (GNG) (Fritzke, 1994) clustering algorithms that are in line with this thesis.

3.5.1 Self Organizing Map (SOM)

Self-organizing map (SOM) (Kohonen, 1998) is an unsupervised clustering algorithm based on a grid of artificial neurons whose weights are learned from the training input data. SOM clusters the input data into a 2D grid space. The number of clusters in SOM is predefined. SOM is a powerful tool to visualize high-dimensional clustered data in a 2D space. It performs a nonlinear projection from high-dimensional data onto a regular (usually) two-dimensional grid while neurons try to approximate the density of the input data.

The SOM approach allows the mapping of multidimensional data onto a lower-dimensional model using a competitive learning approach. In a competitive learning scenario, nodes compete to represent subsets of input data, leading to the formation of meaningful clusters within the dataset. Each data point in SOM competes for representation by calculating its distance from each neuron (in a grid).

Based on a predefined number of nodes (number of clusters), the SOM algorithm measures a distance vector between each input sample and each neuron's weight, and assigns the input vector by selecting the best matching unit based on the minimum distance between that input vector and the neurons. To avoid the need for a predetermined number of nodes that create the topology, other clustering techniques like GNG (Fritzke, 1994) are now getting more attention.

3.5.2 GNG

The Growing Neural Gas (GNG) (Fritzke, 1994) is an incremental algorithm where number of clusters are learned than predefined. The creation of new nodes is determined based on the training data dynamics. GNG training dynamics creates, in the output space, a structure that can be considered as a graph, with neurons being vertexes (Costa and Oliveira, 2007) growing eventually like a gas diffusion.

GNG starts with a minimal neuron structure that is incremented during training until it reaches a maximum number of nodes. The training process can be stopped by setting a specific condition manually. A carefully chosen discriminator, or group of discriminators (e.g., the distance from the cluster center, the covariance of clusters), allows one to decide when to stop the algorithm. Altering the process may produce sub-optimal results. To set stopping conditions, the user must know the characteristics of the input data in detail.

GNG learns a topological structure based on the input vectors that perceptual systems aim to create an optimized number of models (Ozasa et al., 2023). This clustering method is advantageous since it can be adjusted to match a particular rule when grouping each data point. For example, when applying GNG to 4D vectorial data, representing position and velocity in both x and y directions, it can be adjusted to make the topologies oriented to the positional data or velocity data. During topology creation, GNG dynamically changes the structure of each cluster by adding and removing nodes and edges according to the input state space model.

Learning-based topology and flexibility to optimize some characteristics of the input data make GNG-based clustering algorithms more preferable. However, GNG has many parameters that should be analyzed for different scenarios, making them more complex algorithms.

3.6 Anomaly Signals

Anomalies are data points that deviate from the overall pattern of an entire data set (Slavic et al., 2022). In many scenarios, manually labeling an entire data set to train a model to detect anomalies is unrealistic, especially when the relevant data have many more normal samples than abnormal ones. In those scenarios, anomaly detection, based on semi-supervised or unsupervised learning is a more viable solution (Chandola et al., 2009; Zhang et al., 2023).

Supervised Anomaly Detection, detects anomalies by assuming the availability of training datasets that are pre-labeled as normal and abnormal. In such cases to detect the anomaly, a predictive model for normal versus anomaly classes can be applied.

Semi-supervised Anomaly Detection, assumes that the training data are only labeled for normal behavior, so the detection is generated from the learned model, by comparing it with the normal behavior. Since these methods do not require labels for the anomaly class, the approach used is to train the model for the labeled class corresponding to the normal behavior and use this model to identify anomalies in the testing data.

Unsupervised Anomaly Detection techniques used in this mode do not require training data, under the assumption that the majority of the instances in the dataset are normal. Such detection techniques assume that the test data contains a small number of anomalies so that the model learned during training is robust enough to detect these few anomalies.

3.7 Summary and discussion

In this chapter, we discussed how state space models that are composed of higher dimensional and lower dimensional variables are compactly represented in the Bayesian framework. The conditional independence assumption in directed graphs allows us to derive a distribution through the chain rule for Bayesian networks. This independence assumption helps to factorize the probability distribution as a representation of, a set of independent samples that hold for the modeled state space. This class of models can be designed with the assumption of Linear Gaussian State-Space Models, called Kalman filters (Kalman, 1960) as discussed in section 3.4.1.

Bayesian networks can capture complex interactions between interrelated objects with the notion of prior knowledge by providing a richer understanding that substantially improves the quality of predictions. We have discussed the general framework for defining the probabilistic model. We described representation levels for temporal representations that allow the specification of models over general object-relational domains. Probabilistic relational models allow a considerably richer set of dependencies to be encoded easily, but

at the cost of both conceptual and computational complexity that requires dimensionality reduction, especially for higher dimensional state spaces.

Through interaction modeling, DBNs allow probability distributions over probability spaces that are significantly richer in modeling uncertainty about the interaction network interrelationships between agents, as it will be the main focus in chapter 6. These interaction models need special attention, especially when inference algorithms are applied. Hierarchical inference for interacting agents is challenging when states are nonlinear or piecewise linear. The learned interaction network is complex and densely connected due to multiple interactions between variables. Consequently, it is necessary to optimize inference and decision making algorithms in these interaction models.

Chapter 4

Self-awareness and Self-Expressive Systems

Self-awareness in autonomous agents can be expressed as a capacity to adopt human cognition as a reference to understand the environment and make decisions. Simon Haykin, in his cognitive dynamic systems book (Haykin, 2012), discussed the four fundamental capabilities that an autonomous agent must have to be a cognitive dynamic agent. As a first capacity, a cognitive dynamic system (CDS) has a perceptor and actuator that allow an agent to perceive the world (perceptor) and to control itself based on the information from the perceptor (actuator) that generate a cycle of perception and action, called the perception-action cycle. The second capacity is memory, which deals with changes in the environment through learning and storing the knowledge acquired incrementally. Attention is the third capability in the CDS's selective focusing, based on the task in mind. The fourth and last capacity, according to Simon Haykin, is intelligence. Intelligence is a unified view of the other three capabilities that includes the correction mechanism of feedback at multiple levels. This feedback information is also refereed anomaly signal (Slavic et al., 2022). These cognitive capabilities of a dynamic system that understands the environment and itself, from exteroceptive and proprioceptive sensory signals, make an agent self-aware from the signal processing point of view (Regazzoni et al., 2020).

4.1 Definition and characteristics

Self-awareness (SA) is a characteristic of an agent that is aware of itself and the environment accumulated through experience. It is a learnable behavior that minimizes surprise in making intelligent decisions in uncertain environments. Self-aware systems must be generative in

predicting the future state of the self and the environment, which makes them self-reflective observers (Regazzoni et al., 2020).

The four characteristics (perception-action cycle, memory, attention, and intelligence) (Haykin, 2012) define a cognitive embodied representation, inference, and decision-making knowledge unified to improve autonomy in artificial agents.

Perception of the environment is equivalent to state estimation that receives information and models it for adaptable knowledge representation and inference mechanisms. In perception stage, initialization, model creation, inference, anomaly detection, and interfacing to control are core characteristics. An autonomous systems having these characteristics can operate in highly dynamic, interactive, and uncertain environments (Regazzoni et al., 2020).

Each self-awareness function can be modeled as a subsystem of integrated models to produce self-awareness capabilities (Lewis et al., 2015). Each subsystem must interact within the other internal subsystems and also with external biological or artificial agents. These interaction models require a unified modeling of understanding, estimating, and acting functionalities. In addition, asserting reliability and security in uncertain situations, including system failures, are crucial functionalities that a robust system has to provide.

Self-awareness in terms of the sources of knowledge, from internal or external sensors, underlines the notion of private and public self-awareness (Chen et al., 2014; Regazzoni et al., 2020). The private aspect is linked to a processing unit that processes the internal phenomena of the agent, while the public part is associated with the external knowledge to the agent's self.

The subsequent sections focus on levels, architectures, learning, and decision-making frameworks of self-awareness.

4.2 Self-awareness Levels

Self-awareness can be categorized into five main levels (Chen et al., 2014; Lewis et al., 2015):

Stimulus-aware: Knowledge of stimuli in responding to an event that is originated from internal or external states of the system. Stimulus awareness is a prerequisite for all the other levels of self-awareness, and it can be private, public, or both.

Interaction-aware: It is an extension of stimulus awareness that includes knowledge of the interaction between subsystems and external agents. Interaction awareness is based on external and internal events, making it public or private self-awareness.

Time-aware: Time awareness is related to memory in preserving experience and the capacity to estimate future states. Time-awareness is both private and public, since the agent needs to keep track of itself and the external environment simultaneously.

Goal-aware: Preferences, objectives, and constraints are in this category. A subsystem can be designed implicitly to reach a particular goal, but to be goal-aware, it needs to reason about it. Differentiating states from goal states, understanding strategy (utility), and discriminating between long-term and short-term objectives, are benchmarks in this type of awareness. Goal awareness is flexible in changing the goal states or objectives. Linking goal awareness with time or interaction awareness can reason out about goals related to external agents and likely future goals. Goal awareness can be private, public, or both self-awareness types.

Meta-self-aware: Meta-self-awareness is a high level of awareness capable of understanding the cost and benefits of maintaining a certain level of awareness. It allows an agent to update the levels of awareness by changing algorithms and the degree of complexity in realizing the awareness levels. Through contentious monitoring of its internal system, it can change goals dynamically. Its main objective is maintaining the internal coherence of internal processes, which makes it private self-awareness.

An architecture designed or learned for a cognitive system must address the levels and capabilities of self-awareness. An architecture can be heuristically designed as in (Lewis et al., 2015) or can be learned from expert data (Alemaw et al., 2025) that can be augmented heuristically for costs related to intrinsic components which are immutable (not trainable) costs (LeCun, 2022). MASAA (Alemaw et al., 2025) is a Deep Learning-based architecture that we developed inspired by Joint Embedding Predictive Architecture (LeCun, 2022) and controlled hallucination theory (Seth, 2021).

4.3 Self-awareness Architecture

An architecture is a blueprint for an object that communicates its interrelated subsystems to function effectively and efficiently. It must provide detailed functions of subsystems and their interdependency through input-output linkages. The integration model should link each subsystem robustly, to naturalize decision-making and anomaly detection processes.

The development of a self-awareness architecture can be started using the Global Workspace Theory (GWT) (Signa et al., 2021). In GWT, the first phase is about the external stimuli. It activates the functionalities of the sensorial memory of the agent generating perception, where interrelations of perceptions are processed in a workspace, producing a perception model. According to this theory, the second phase is the consciousness phase.

Collections of perceptions (multi-modal perception) are fed to the Global Workspace to select the best perceptual signal as a content of consciousness. The third phase, which is the last stage, is the action selection procedure. Action selection is a strategy that brings preferred outcomes based on a criterion set by the Global Workspace. The selected action pattern (policy) will be stored in sensory-motor memory to activate the behavior that closes the current perception-action cycle and initiates the next cycle.

Another self-awareness architecture has been presented in (Chen et al., 2014). It is an architecture pattern based on problem-solution by encompassing cognitive capabilities in a given context. The focus in this architecture is associated with capabilities interrelated through interaction awareness. Interaction with other neighboring agents is based on learned experiences, without the need of a communication protocol.

In the following subsections, we briefly introduce the learning-based self-awareness architecture (fig. 4.1) components, that are building blocks of the MASAA (Alemaw et al., 2025) architecture, where we use the previous architectural viewpoints as a baseline. We will discuss in detail the MASAA architecture in 7.

4.3.1 Sensing Model

The sensing model is an interface between the hardware and the software components of an agent. The physical system of a cognitive agent is equipped with many different (proprioceptive and exteroceptive) sensors that must harmonize with the Cognitive Dynamic System (CDS). The CDS is the mind of the cognitive agent that needs to be updated through data and control information (Regazzoni and Pitas, 2019) coming from the physical sensors.

The sensing module provides information to the CDS from lower-dimensional (GPS, Odometry, IMU) and higher-dimensional (Video, LiDAR) sensors.

4.3.2 World Model

MASAA aims to build a cognitive knowledge from low-level understanding to higher-level semantic meaning and reasoning schemas. The world model is the start of the learning process. It is a vocabulary, from lower-dimensional sensory inputs, that creates low-level cognition. MASAA is a learning-based architecture where the World Model (WM) is learned from data, following an unsupervised machine-learning approach.

The WM can process different and interrelated information collected from the sensing model. As demonstrated in (Alemaw et al., 2024), odometry and control data are used to build the WM vocabularies. Equipped with both vocabularies (knowledge sets), WM can represent and infer the external and internal states of an autonomous agent. In the WM, agents can

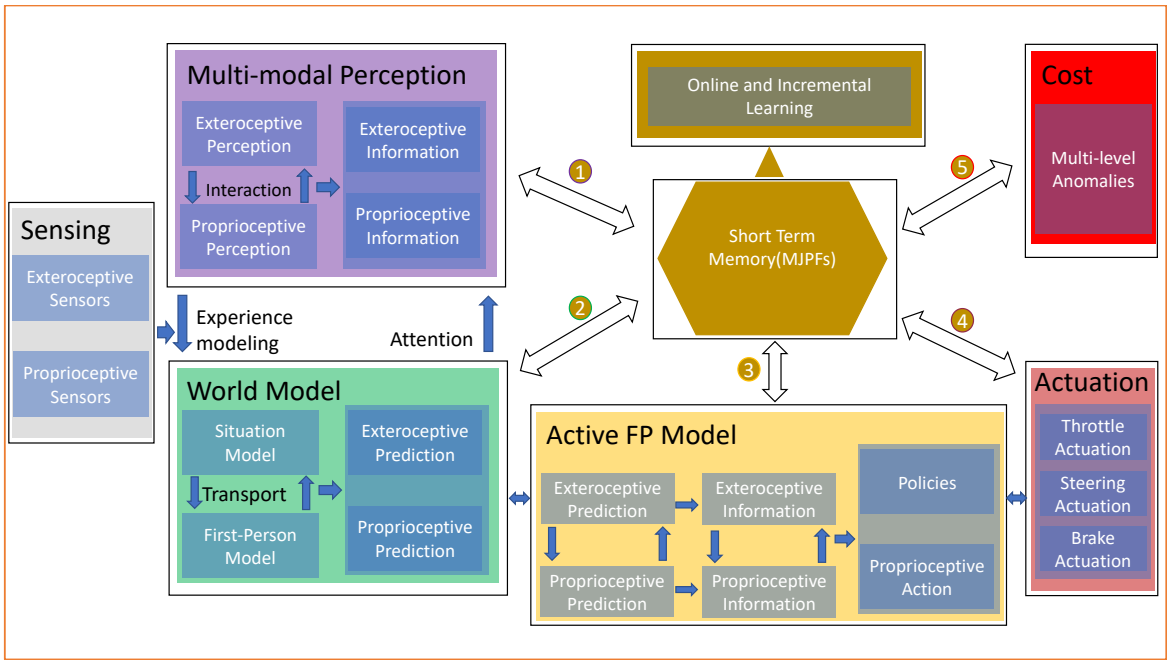


Figure 4.1: MASAA: Cognitive architecture for autonomous agents. Arrows 1 to 5 are processes, based on model output errors, for correcting (changing) the corresponding model as an incremental learning process. The sensing module is a hardware module interfacing with the external world by translating physical stimuli into an initial internal representation. It produces a time series of different observations for both the WM and MMP modules. The WM processes the incoming lower dimensional data and produces vocabularies representing the agent's self-state in a context. The MMP module uses the WM vocabulary as a guiding attentive knowledge. The integrated WM representation is used as an encoded collection memory of sensorial active experiences of the agent, to be used when the agent has to perform experienced tasks. The learned WM Situation model is applied under a First-Person viewpoint as a result of embedded explicit relations between external and internal variables. The short-term memory module represents the communication module, which is the interaction medium using Markov Jump Particle Filtering. The Active First-Person module is an online process that selects actions based on learned policies and transfers action commands to the Actuation module. © 2024 IEEE.

predict future sequences of their actions and select the best actions. They can reason, plan, and explore new environments and avoid dangerous situations (obstacles) through a new set of actions.

Reasoning, obstacle avoidance, and exploring new environments are coordinated tasks that demand higher-dimensional knowledge (section 4.3.3), incremental learning ability (section 4.3.8), knowledge to detect anomaly signals (section 4.3.5), avoid dangerous situations through best policy (section 4.3.6), and model incrementality (section 4.3.8).

4.3.3 Multi-Modal Perception Model

Processing higher-dimensional data has good characteristics and undesirable behaviors. The more samples you have that cover a larger area of interest, the better the accuracy in state estimation. Increasing samples and coverage requires extensive resources to process and learn salient features from vast data. There should be a common ground to balance accuracy and computational resources with a feasible and robust solution.

We build the Multi-Modal Perception (MMP) component of MASAA from higher-dimensional data learned according to a controlled hallucination theory (Seth, 2021), by using the WM vocabulary-based attention vector guiding the learning of the MMP model. The MMP model in (Alemaw et al., 2024) takes video data as a higher-dimensional sensory input. However, the MMP model can also include other sensor inputs like LiDAR to build multiple perception models learned according to the rules of the WM model.

The guiding vector from the WM vocabulary plays an important role in building a perceptual memory that enables the cognitive agent to infer future states based on the information preservation principle, thus preserving only salient features. The first MMP vocabularies are learned from expert demonstrations in an offline mode. This process creates initial knowledge about the external environment. This knowledge has a mapping with the semantics of the WM model, producing combined vocabulary (perceptual salient memory). These combinations in the actor module (section 4.3.6) enable the agent to navigate the learned environment.

4.3.4 Memory Model

Simon Haykin (Haykin, 2012) grouped the memory modules in the perception-action cycle, into perceptual, executive, and working memories for cognitive radar and radio systems. The perceptual memory is related to the receiver, helping the receiver to estimate the target. The executive memory guides the transmit-waveform selector in the transmitter to select the best waveforms to transmit. The working memory, reciprocally couples the executive and perceptual memories. The working memory enables the radar system to predict the consequences of actions taken by the radar system.

In line with Simon Haykin's memory modeling, MASAA builds a short-term memory module, realized through the Markov process, to coordinate higher-order (H-DBN levels) vocabularies (can be taken as memories) from each model of the architecture, and communicate each model through Markov Jump Particle Filters (MJPFs). The short-term memory module is a medium for changing situation models to first-person (Active First-Person) models.

4.3.5 Cost Model

The cost module is a general discomfort measure for an autonomous agent. Particularly in MASAA, it quantifies the degree of mismatch between predictions and observations. MASAA's cost module can also include a set of signals assessing the agent. The cognitive artificial agent tries to minimize these costs. In each H-DBN level, there are methods to measure the level of discomfort (anomaly signals) of an agent. These measures are mainly probabilistic distance measures.

Cost in learning is an optimization process of an objective function to select an action that maintains preferred observations in the future by minimizing a quantity known as expected free energy (EFE). The referred observations are the ones that a goal-aware agent wants to be in (reach) by minimizing a quantity known as variational free energy (VFE). This unified approach, based on both VFE and EFE minimization principles, is the procedure for learning and acting in active inference (Friston et al., 2015). To understand perception and learning, an agent has to minimize VFE, while for action selection, planning, and decision making the agent needs to perform EFE minimization coherently.

Representation, inference, and decision-making can be achieved through different machine learning algorithms. Reinforcement Learning (section 4.4), Imitation Learning (section 4.5), and Active Inference (section 4.6) are the main algorithms that enable an agent to become a cognitive agent. In MASAA, any learning algorithm can be applied to learn each component. Hence, we discuss these learning algorithms in the following sections. However, MASAA in this thesis, is described and tested through explainable models, by integrating explainable Imitation Learning and active inference methodologies.

4.3.6 Actor Model

As introduced in (LeCun, 2022), the actor module computes alternative policies or proposals for sequences of actions. From these, it selects the best one and outputs it to the effectors (actuators). In MASAA, the actor generates sequences of actions to be used by the WM for future state estimation, and feeds them to the Cost module. The Cost module will measure the degree of compliance based on both EFE and VFE, for each proposed action sequence. The actor module then selects the lower-cost actions and outputs the first action (if some sequences have a similar low cost) to the effectors.

In this module, the learned situation models are applied under a first-person viewpoint as a result of embedded explicit relations between external and internal variables. It is an online process that selects actions based on learned policies and transfers action commands to the Actuation module.

4.3.7 Actuation Model

The Actuation model is an interface between the hardware and the software components of the cognitive agent. This model makes the physical system evolve instantly, according to the commands transferred to it. It may include feedback information to let the agent know the physical states of its actuators.

4.3.8 Incremental Learning Model

An architecture should be scalable to introduce a new set of rules as new situations are encountered, and to update the existing ones. The incremental learning module interfaces with both the Cost and the short-term memory modules to update each vocabulary if new experiences are faced in the online learning and decision-making processes. In online incremental learning, the current knowledge can be updated through retraining the entire model, or through transferring model parameters and fine-tuning them. Another choice is to keep different models for each scenario for the new environmental setup.

Incrementality in self-awareness helps cognitive agents to actively memorize generative and discriminative meta-knowledge in their Autobiographic Memory (AM), by processing sensorial experiences (Regazzoni et al., 2020). The updated AM is used to infer internal and external perceptions that allows them to generate optimal policies. In this way, the perception-action cycle starts from initial models and builds incremental models from new experiences incrementally.

4.4 Reinforcement Learning

Learning from interaction with the environment without an explicit teacher is the foundational idea of reinforcement learning (RL) (Sutton and Barto, 2018). RL focuses on learning goal-oriented algorithms through interaction with the environment. It is a learning algorithm that maps situations to actions through a trial-and-error approach to maximize a numerical reward signal, by trying many actions and discovering actions that yield the most rewarding signal. Hence, trial and error, and learning through reward functions, are the main characteristics of reinforcement learning.

The reinforcement learning problem is formalized according to Markov decision processes in solving sensing and goal-oriented action in observable Markov Decision Processes (MDPs) or partially observable MDPs (POMDPs).

According to Sutton and Barto (Sutton and Barto, 2018), RL is not a supervised or unsupervised machine learning paradigm. RL is a third category of machine learning

frameworks. Supervised learning is a category of learning algorithms which uses a training set of labeled data from an expert demonstration. Since supervised learning tries to categorize based on a decision boundary, it is called a categorical-based learning approach. Its goal is to make good generalizations in its decisions, in unseen situations at the training process. One of the limitations of supervised learning is its impracticality in obtaining examples of desired behaviors that are both correct and representative of all the situations in which the agent has to act.

Unsupervised learning is an approach to learn a hidden feature (structure) from unlabeled expert demonstrations. Comparing RL with unsupervised learning, RL does not depend on labeled behavior, as well. However, it differs from unsupervised learning as it aims to maximize a reward signal instead of finding hidden structures in the data. Policy, reward, value function, and optional model of the environment are key components characterizing RL-based agents, as follows.

Policy: A rule that maps perceived states of the environment to the possible actions, in that particular environment. Policy is the main component of RL since it sufficiently determines the behavior of an agent in that environment.

Reward signal: A signal that defines the goal of an agent. It is a signal from the environment that is sent to the reinforcement learning agent to make it understand the nature of its action as a rewarding or a punishing one. The reward signal alters the policy if the reward is a punishing one, as the policy has to be changed to select other actions in that situation in the future.

Value function: Determines the long-term desirability of states through the accumulation of rewards. Even if the reward is low in a particular state, the value function cares about the total reward an agent can expect over the future, starting from the start state. The difference with reward signal is that reward signal indicates what is good in immediate desirability, whereas values are for long-term accumulation of rewards. Hence, values are estimations of rewards from the sequences of observations an agent makes.

Model of the environment: This is a particular characteristic of model-based RL systems. A model of the environment allows an agent to estimate the behavior of the environment itself (state and reward) in the future, given the current state and the selected action before it is experienced by the agent.

Reinforcement learning, in general, or specifically model-free methods, ignores evolutionary methods like imitation learning. The reason is that, even though an expert policy is there, an agent should experience a state-to-action mapping to have explicit self-experience of those states, as expert accumulated states might be wrongly perceived or misunderstood (Sutton and Barto, 2018) by the learning agent.

Learning by direct interaction does not require supervision or having complete models of the environment (which can be an advantage). However, learning from scratch (trial-and-error) is a very computationally intensive and expensive approach in many artificial agent training fields, and might lead an agent to malfunction or even to be useless. In some fields, like surgery, it create many dilemmas when we have to learn on a human body by trial-and-error methods. Moreover, the resources required to train an agent from an unresponsive object to a responding and reasoning agent, drive researchers to explore alternative paradigms like model-based RL and imitation learning techniques.

4.5 Imitation Learning

Imitation is the art of learning from demonstrations of expert data. It is a learning scheme that allows an agent to transform situation models into first-person models. Generally, imitation learning can be approached in the form of behavior cloning (BC) or inverse reinforcement learning (IRL) (Osa et al., 2018; Argall et al., 2009). Behavior cloning learns a policy through supervised learning methodologies (Bain and Sammut, 1995; Pomerleau, 1988). Inverse RL is based on unsupervised techniques where a cost function is minimized, aiming to generalize well beyond the expert-demonstrated data (Osa et al., 2018).

In autonomous driving, imitation learning aims to train an agent how to map a high-dimensional sensory input data to a low-dimensional actions (such as steering, accelerating, and breaking) (Andreas Geiger, 2022). The expert actions can be expressed as:

$$\begin{aligned} a^* &\in \mathscr{A} \\ \pi^* &: \mathscr{S} \rightarrow \mathscr{A} \end{aligned} \tag{4.1}$$

where $\mathscr{A}$ is the set of all possible actions, a^* is the optimal action from the set of actions, π^* is the best policy that the expert demonstrates in $\mathscr{S}$ (which represents all possible set of states). States can be observable or partially observable.

Imitation learning endeavors to learn a mapping function for the learning agent:

$$\pi_\theta : \mathscr{S} \rightarrow \mathscr{A} \tag{4.2}$$

From state $s_t \in \mathscr{S}$ taking action $a_t \in \mathscr{A}$, the agent's intention is to be in the next state s_{t+1} by learning a policy π_θ. This can be formalized in the form of probabilistic state transition $p(s_{t+1}|s_t, a_t)$. It can be learned by minimizing a loss between an expert policy (π^*) and

learning agent policy (π_θ) as:

$$\underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{s\sim p(s|\pi_\theta)}[\mathscr{L}(\pi^*(s), \pi_\theta(s))] \tag{4.3}$$

the state distribution $p(s|\pi_\theta)$ depends on the policy π_θ

The general imitation learning eq. (4.3), in behavior cloning (BC) is simple supervised learning, $\sum_{t=1}^{N} \mathscr{L}(a_t^*, \pi_\theta(S_t^*))$. In standard supervised learning, since mostly we make the IID (independently and identically distributed) assumption, if a policy deviates from the expert trajectory, the observation will be outside of the experience, which makes the agent drift and might lead to undesirable situations, like collision (Codevilla et al., 2019).

Data aggregation (DAgger) was introduced to alleviate BC shortcomings through the data aggregation on-policy generated new samples. The aggregation process is offline after some iterations. DAgger has a limitation if the sampling strategy for the new data is performed randomly. The model may overfit due to unbalanced datasets.

In self-driving, DAgger can be improved through critical state sampling, based on the driving scenario and replay memory buffer from on-policy data (Chen and Huang, 2017; Bicer et al., 2019; Prakash et al., 2020). The replay memory should focus on highly uncertain regions of the environment, to smooth the sudden changes in the dataset.

End-to-end driving models on the other hand have limitations in regions like cross-section roads. Conditional imitation learning introduces control input to enable an agent to input a new set of high-level actions (Codevilla et al., 2018; Abdou et al., 2019; Xiao et al., 2020).

Imitation learning can be integrated with active inference to address distribution shifts (Nozari et al., 2022). In active inference (Friston et al., 2015; Smith et al., 2022), perception and action selection are unified. The state estimation and the resulting behavior can be harmoniously coupled based on the free energy minimization principle.

4.6 Active Inference

The establishment of active inference is akin to the human brain in maintaining the body in a stable state. The brain predicts sensory inputs with the expectation of preferred outcomes and measures the outcomes by relating them with real measured sensory inputs. Mismatches in this generative process are termed surprisals (Prabhushankar and AlRegib, 2023). Mismatches (surprisals) are sensory errors or anomalies that can be handled by updating sensory input that improves perception and, through actions that can change the environment to make the agent in a preferred state. Hence, the active inference framework is constituted based on two key assumptions. The first one is the variational free energy (VFE) minimization,

which drives perception and learning. The second assumption is the expected free energy (EFE) minimization principle, that rules action selection, planning, and decision-making. EFE quantifies the VFE of various actions based on expected future outcomes (Smith et al., 2022; Alemaw et al., 2024).

Actions can be treated in two ways. In the first case, if the outcome of each action is known with certainty, the agent has to select the action that leads to the outcome that is most preferred. The challenge in this approach is that if the outcomes are large and complex, the agent must weigh different factors in determining which of the possible outcomes is the most preferable, which is not a trivial task. The second more difficult task appears when the outcome of an action is not fully determined. In this second case, the agent must take into account both the probabilities of various outcomes and the preferences of the agent between these outcomes.

The difference between simple probabilistic models and causal models is that causal models are not simply observation models that only represent the values that variables take, but also can take actions that can manipulate these variables (Koller and Friedman, 2009) as in decision theory. Decision theory, indeed, provides a formal foundation for decision-making under uncertainty and, by assigning numerical utilities to various possible outcomes, encoding the agent's preferences called maximum expected utility. Each outcome, say o, is associated with a numerical value $U(o)$, a numerical encoding of the agent's preferences whose magnitudes are meaningful. Hence, it is possible to probabilistically aggregate utilities and compute their expectations over the different likely outcomes.

The active inference framework is established based on decision theory, where the internal model of the world is maintained through the minimization of the free energy that maximizes the expected utility. Active inference agents can be seen as self-evidencing agents, as they aim to maximize expected future outcomes (Mazzaglia et al., 2021). Compared to reinforcement learning, active inference gives a natural way of representing and understanding the objective of selecting actions. However, active inference has scalability issues, and current implementations are more focused on tasks with low-dimensional sensory inputs (Nozari et al., 2022) or with a small set of discrete actions (Da Costa et al., 2020; Krayani et al., 2024).

Active inference computes VFE ($\mathscr{F}$) by minimizing prediction errors during perception, which can be seen as a measure of complexity and accuracy:

$$\begin{aligned}\mathscr{F} &= \mathbb{E}_{q(z)}\left[\ln \frac{q(z)}{p(x,z)}\right] \\ &= \mathbb{E}_{q(z)}\left[\ln q(z) - \ln p(x,z)\right] \\ &= \mathbb{E}_{q(z)}\left[\ln q(z) - \ln p(z)\right] - \mathbb{E}_{q(z)}\left[\ln p(x|z)\right] \\ &= D_{KL}\left[q(z)||p(z)\right] - \mathbb{E}_{q(z)}\left[\ln p(x|z)\right]\end{aligned} \tag{4.4}$$

Eq. (4.4) is a posterior approximation $q(z)$ of the true probability distribution $p(z|x)$, as detailed in Section 2.5.2. $D_{KL}\left[q(z)||p(z)\right]$ represents the complexity measure of the approximation process, and $\mathbb{E}_{q(z)}\left[\ln p(x|z)\right]$ is the accuracy where a hidden state z is generating an observation x. VFE can also be expressed in terms of putting an upper bound on the surprisal (log evidence) of the model:

$$\mathscr{F} = \mathbb{E}_{q(z)}\left[\ln q(z) - \ln p(z|x)\right] - \ln p(x) \tag{4.5}$$

Eq. (4.4) rearranges $\mathbb{E}_{q(z)}\left[\ln q(z)/p(x,z)\right]$ showing VFE is always greater than surprisal (i.e., it is an upper bound on surprisal) with respect to $(-\ln p(o))$, where this term represents the information gain. By changing the sign of VFE it becomes an evidence lower bound (ELBO) as it can be referred in Section 2.5.2.

Active inference, in addition to minimizing prediction error in perception, models the rule of action selection as well. To select optimal actions, the agent must consider future observations before they happen based on the current observation. A model must predict sequences of future states and observations for each possible policy, and then compute the expected free energy (EFE) associated with each sequence of future states and observations. In action selection, the assumption is that the model of the agent is biased towards its preferred outcomes, distributed according to the prior $p(x)$. Considering the approximate inference in eq. (4.4), the EFE ($\mathscr{G}$) can be written as:

$$\begin{aligned}\mathscr{G} &= \mathbb{E}_{q(x,z)}\left[\ln q(z) - p(x,z)\right] \\ &= \mathbb{E}_{q(x,z)}\left[\ln q(z) - \ln p(z|x)\right] - \mathbb{E}_{q(x)}\left[\ln p(x)\right] \\ &\approx \mathbb{E}_{q(x,z)}\left[\ln q(z) - \ln q(z|x)\right] - \mathbb{E}_{q(x)}\left[\ln p(\tilde{x})\right] \\ &= -\mathbb{E}_{q(x,z)}\left[\ln q(z|x) - \ln q(x)\right] - \mathbb{E}_{q(x)}\left[\ln p(\tilde{x})\right]\end{aligned} \tag{4.6}$$

where $\tilde{x}$ is the preferred observation encoded in $p(\tilde{x})$, which the agent seeks to find policies expected to generate those observations. Eq. (4.6) can be rewritten in terms of KL divergence

and entropy as:

$$\mathscr{G} = D_{KL}\left[q(x)||p(\tilde{x})\right] + \mathbb{E}_{q(z)}\left[H\left[p(x|z)\right]\right] \tag{4.7}$$

Therefore, EFE is expressed as a measure between preferred outcomes and those expected under a policy (sometimes called risk or expected complexity), plus the information gain.

4.7 Summary and discussion

In this chapter, self-awareness definitions and characteristics were introduced from different viewpoints. The five fundamental characteristics (perception-action cycle, memory, attention, and intelligence) of a cognitive agent were explained according to the computational context of self-awareness.

In self-representation systems, the perception of the environment is continually influenced by current observations received by the agent, in addition to the knowledge gained from experience (Haykin, 2012). In these systems, the nature of recognition in retrieving learned knowledge from AM, and its update process due to new information received, must be performed in a self-organized and adaptive way. Thus, the cognitive preceptor can be recognized as a hierarchical probabilistic filter representing the updated knowledge at different abstraction levels.

The main focus of this thesis is building self-expressive systems learned from multi-sensorial information in a self-supervised manner. In this context, we build a self-awareness architecture composed of sensing, low-level representation, high-level representation and interaction, memory, cost, action and actuation. We then consider to learn online and incremental integrated models for a cognitive agent.

The developed self-awareness architecture is flexible to be learned from the combination of multi-sensory data, using different learning algorithms (se Sections, 4.5, 4.6, and 4.4). Multi-agent self-awareness architecture (that is discussed in chapter 7), is carried out through imitation learning and active inference based algorithms.

Chapter 5

Datasets

Experience is the integration of knowledge, activity, and reflection. Learning is a process of reflection, critical analysis, decision making, and accountability. A designed learning experience includes causes and consequences, mistakes and success. Experiential learning is the process whereby knowledge is created through the transformation of experiences. Knowledge results from the combination of grasping and transforming experience in four stages (Sims, 1983). Experiencing, reflecting, thinking, and acting are the four stages in human experience learning process. Building concrete experience starts by learning from experienced users through perception. Reflection is a process of inference on the learned perception. Analytical thinking can be associated with the process of proposing possible solutions used in the acting stage. The action stage or active experimentation, tests the learned knowledge by taking actions to get a feedback (reward) and then to create new experience.

In machine learning, datasets are experiences of expert demonstrations that are accumulated and used to train a new model. As in fig. 5.1, which basically used in education, the cycle starts by understanding experiences (expert-knowledge) through learning, an agent can build an understanding of the world. This knowledge can be incrementally developed into abstract concepts from which an action can be drawn to meet certain requirements of performance. It then can create new experiences by itself through active experimentation (action).

In this thesis we used public, private, and simulated datasets (expert experiences) to develop a framework through different learning, inference, and decision-making algorithms. We introduce these datasets in the following sections.

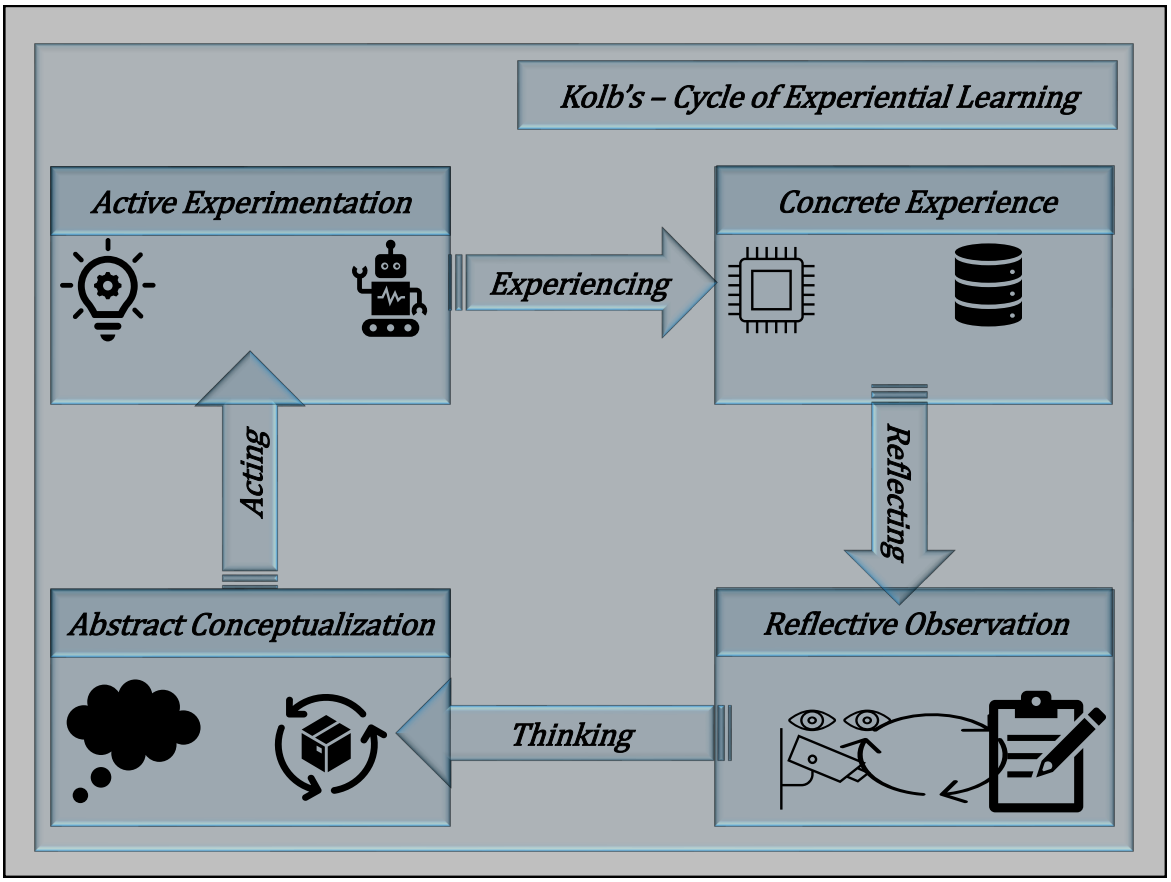
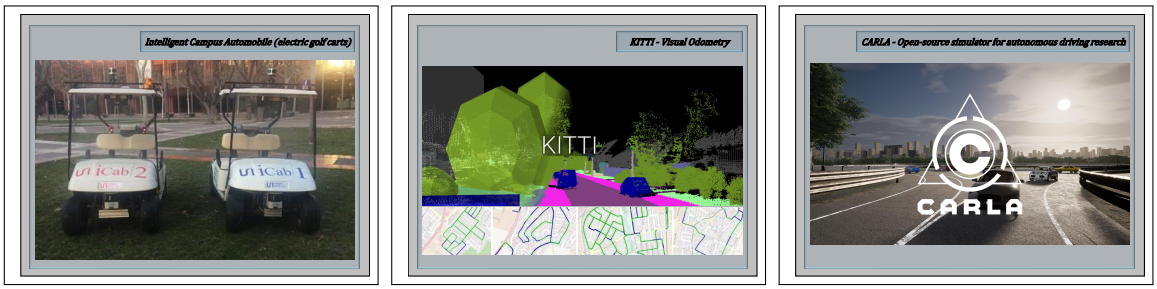


Figure 5.1: Kolb's Experiential Learning Cycle, learning is the process whereby knowledge is created through the transformation of experience.



(a) iCabs (b) KITTI (c) CARLA

Figure 5.2: (a) iCab (intelligent Campus automobile), electric golf carts navigate autonomously within the UC3M campus(a). (b) KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute), public datasets for use in mobile robotics and autonomous driving. (c) Simulation platform supports flexible specification of sensor suites, environmental conditions, and full control of all static and dynamic actors.

5.1 iCab

iCab is an acronym for Intelligent Campus Automobile, at which multiple unmanned ground vehicles (electric golf carts) (Marín-Plaza et al., 2016) navigate autonomously, within the campus vicinity, to transport visitors from one spot to another between buildings.

In the iCab dataset, overtaking interaction is demonstrated between the two golf carts fig. 5.2 (a), operated by human driver. Five overtaking interaction experiments are used (see the odometry plots in fig. 5.3). Each overtaking is done in close similar regions. Each experiment is synchronized between agents, and the data frames between odometry and video are synchronized as well. Experiments have 393, 415, 397, 349, and 434 synchronized frames for each agent.

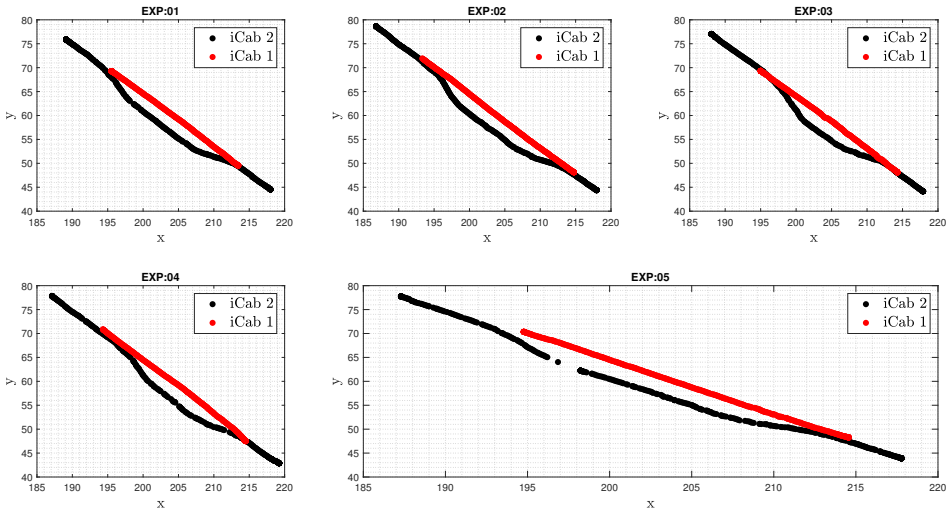


Figure 5.3: iCab: iCab 2 overtaking iCab1, odometry trajectories. Five experiments, where experiments 1 to 3 for training, experiment 4 for validation and experiment 5 for testing.

iCab overtaking experiments are used in Chapters 6 and 7. Interaction modeling and testing in real dataset is performed to validate the interaction from real dataset for the localization of interacting agents (Chapter 6). In Chapter 7, in addition to the simulator based dataset, we used iCab interactions, to demonstrate the performance of MASAA, for multi agents.

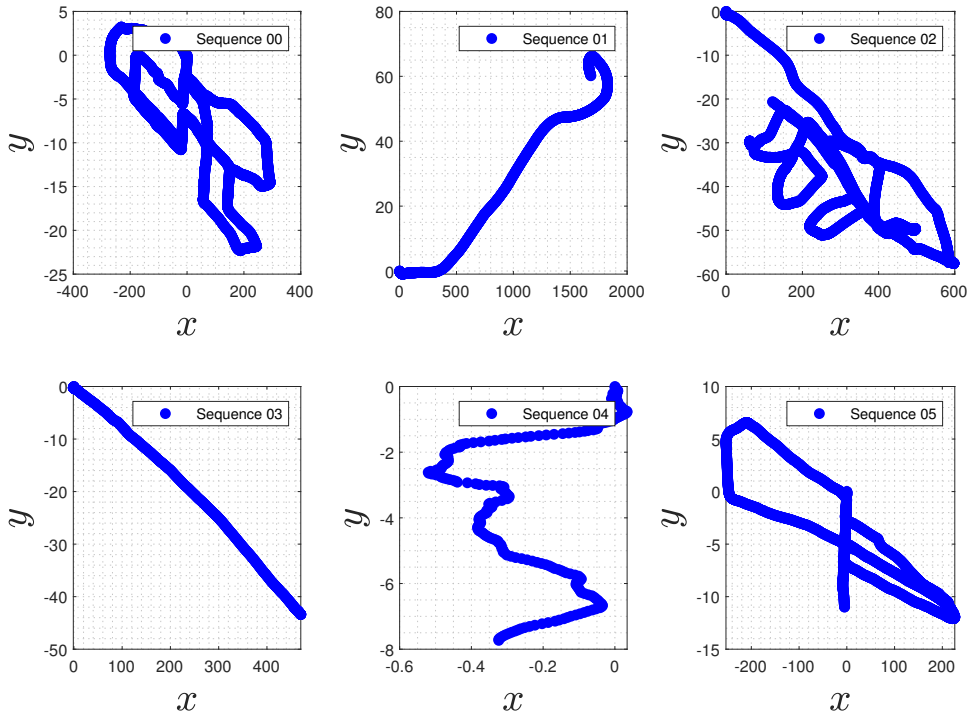


Figure 5.4: KITTI, odometry trajectories of six sequences.

5.2 KITTI

KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute) (Geiger et al., 2012), is a public dataset for mobile robotics and autonomous driving. To validate single agent localization in MASAA architecture, we used six (sequence: 00, 01, 02, 03, 04, and 05) sequences of the grey scale odometry dataset. Each sequence has 4541, 1101, 4661, 801, 271, and 2761 synchronized odometry and video frames (see the odometry plots in fig. 5.4).

5.3 CARLA

CARLA is a simulator which has been developed to support development, training, and validation of autonomous driving systems. The simulation platform supports flexible specifications of sensor suites, environmental conditions, full control of all static and dynamic actors. This capacity is very important to test systems online. In addition, since CARLA supports multi-client architecture, it is invaluable to control multiple actors. Carla has also

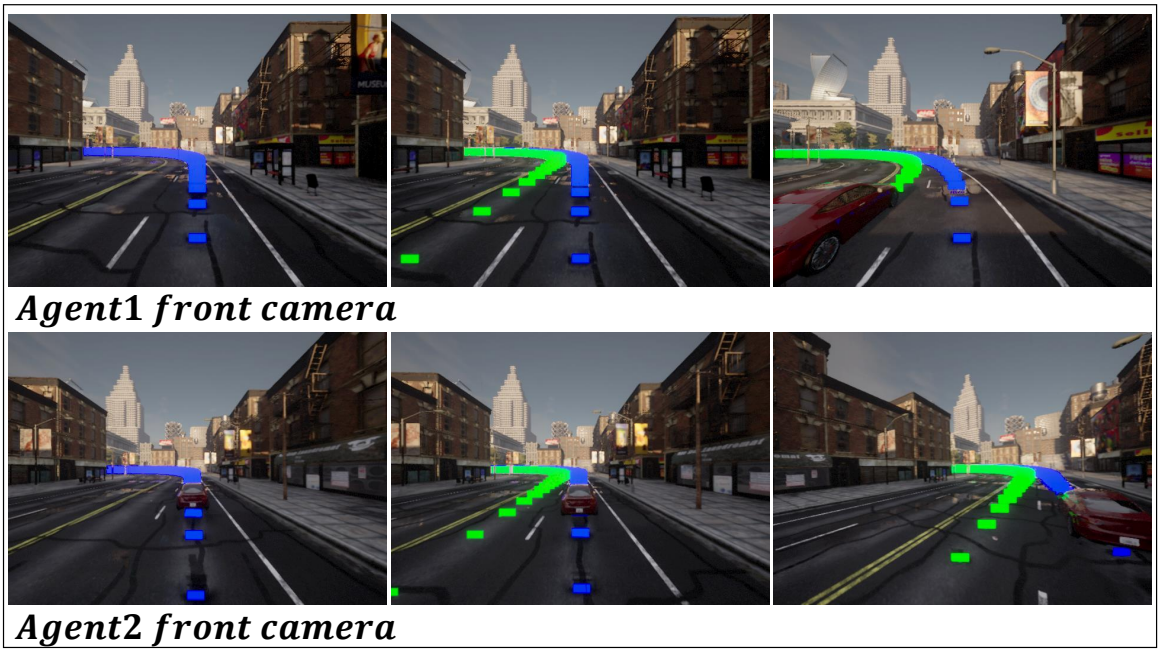


Figure 5.5: CARLA Simulation for interacting agents.

APIs that allows us to control traffic generation, weathers conditions, and integration of multiple sensors.

In this thesis we synchronize Agent 2 and Agent 1 clients performing overtaking maneuvering (see fig. 5.5). Eight experiments, where experiments 01 to 04 are for training, experiments 05 and 07 are for validation, and experiments 06 and 08 are for testing (see the odometry plots in fig. 5.6) Collision sensors and GPS sensors are also used in (Alemaw et al., 2024), which will be discussed in chapter 8.

5.4 Heterogeneous Agents

For heterogeneous interaction, we performed an experiment between iCab1 and a DJI drone (fig. 5.7). The drone's task was to autonomously land on iCab1. The objective of the experiment is to use these heterogeneous agents for tasks like disaster recovery onsite assessments. We choose the landing action to be synchronized with the iCab for data sharing purposes, or charging of the drone's battery.

This experiment is conducted to test and apply the incremental learning ability of a ground vehicle when interacting with drones to perform a particular task. We performed a landing experiment considering both static and dynamic iCab agents. This is relevant for

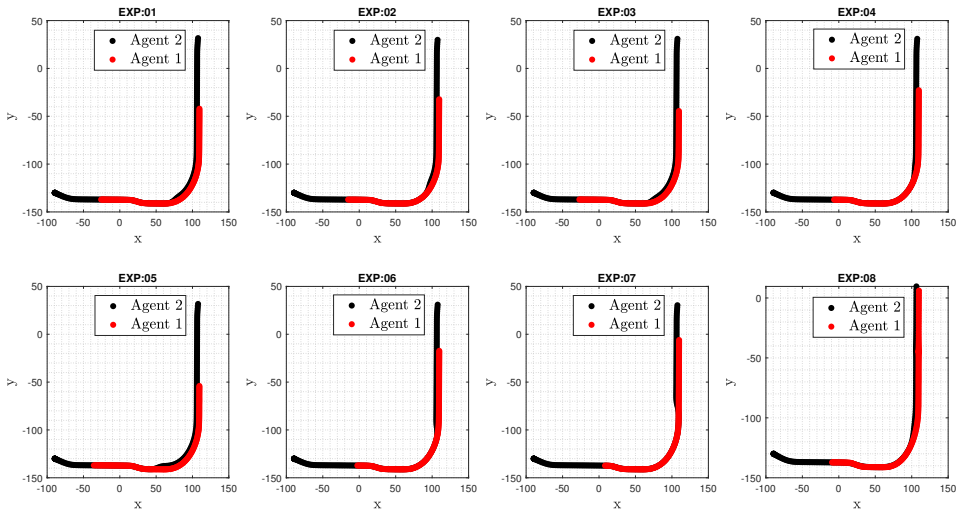


Figure 5.6: CARLA: Agent 2 overtaking Agent 1, odometry trajectories. Eight experiments, where experiments 01 to 04 are for training, experiments 05 and 07 are for validation, and experiments 06 and 08 are for testing.

(a) DJI Drone type (b) iCab1 and DJI drone interaction

Figure 5.7: (a) Drone type used in the interaction experiment. (b) iCab and drone interaction single image taken by iCab camera sensor.

validating the generalizability and scalability of the proposed MASAA architecture and could be extended in our future research to account for multi-heterogeneous agent interactions.

5.5 Summary

These datasets are heterogeneous and collected in different ways. This heterogeneity helps in learning different experiences and as well they are used to validate developed models and algorithms. The data sequences include short, medium, and long trajectory sizes, to have a combined understanding for the learning models. Additionally, testing a developed system using different set of sequences can indicate how robust and generative a model is.

The CARLA dataset is used in Chapters 6, 7, and 8. The CARLA simulator in addition to creating the experiences, it also tests online the learned models. For online learning and decision making we used only the CARLA dataset for model learning, and the simulator for online learning and decision making scenarios. The KITTI dataset, which is a public dataset, is only used in Chapter 7, to validate the MASAA architecture. As we noted in Section 5.4, we did not use the DJI drone in this thesis.

Chapter 6

A Data-Driven Approach for the Localization of Interacting Agents via a Multi-Modal Dynamic Bayesian Network Framework

Interaction can be seen as a representation of an environment in a mental space and making decisions in that environment to coexist with other actors. Representing many scenes, recognizing participating agents, and transmitting control signals to the physical body seems easy in the biological brain. The abstraction of input information for an artificial agents is rather a complex task. An interaction model must abstract the environment and represent the interacting objects in a precise but low-dimensional representation to ease the decision-making process. In this chapter, we introduce a methodology to localize two interacting artificial agents only from higher-dimensional video data. We use the Bayesian approach introduced in Chapter 3. The interaction is modeled hierarchically from lower-dimensional vocabulary to higher-dimensional vocabulary inferred through Multi-Agent Coupled Markov Jump Particle Filter (MAC-MJPF).

6.1 Introduction

The development and growth of Artificial Intelligence (AI) is making our environment more responsive and smarter (Pentland and Choudhury, 2000). AI algorithms, in particular Deep Learning (DL) (LeCun et al., 2015) frameworks, are being developed to mimic Human Intelligence (HI). Some of the most complex and studied aspects of HI are the ability

to abstract higher-level concepts from regular experiences, to adapt to the surrounding environments, and ultimately, to learn from the interactions with other similar agents while maintaining a self state. The ability to abstract higher-level concepts, from observations of experiences, can be adapted to structured knowledge. Environmental abstraction is a fundamental process to make an agent environmentally aware (situation awareness). Artificial agents need to learn from their environment through smooth interactions with neighboring agents. The environment is a shared medium where agents have to implicitly interact to cohabit smoothly. Modeling interactions while localizing oneself is still a challenging problem, and it is still an active study area (Ferdowsi et al., 2019; Forlizzi and DiSalvo, 2006; McFarlane et al., 2016).

Modeling interaction is the way forward to realize self-driving vehicles by improving their environmental understanding as well as their state. Realistic representation systems require low latency in communication and reliable data analytics solutions (Ferdowsi et al., 2019). These solutions can characterize and combine heterogeneous data models that originate from individual systems sharing the same environment. Interaction models can be dealt with the fusion of sensory information that has different modalities. A sensory input can be high or low dimensional in terms of the number of features that are captured. In machine learning, low dimensional data are easy to process and also to represent in a model compared to high dimensional data. Interaction modeling can be carefully performed using one of the data modalities. But a more robust way is to fuse both of them in an innovative way, to gain good coverage of the environmental scene and for effective self understanding.

Lower dimensional information based interaction of dynamic agents, can be modeled in different ways. For example, in (Baydoun et al., 2019), agents are modeled as a series of interplays between attractive and repulsive forces. This interaction is based on low dimensional data represented in terms of Hierarchical Dynamic Bayesian Networks (H-DBNs). This study can be a base to create research questions like, *How can we automatically learn from high dimensional sensors, by deducing information that can be used to estimate an interaction model?* As higher dimensional data give more coverage, thus the representation of high-level detail such us characterizing sudden incidents, these modalities must be included. The video and positional information fusion process introduced in (Fraccaro et al., 2017), is an example in robust multi-sensor data fusion. The authors provide an approach to disentangle content and motion from a bouncing ball video data. For this disentanglement they used two different latent states (a, and z) encoding content and motion respectively. This approach provides a way forward in other areas like image processing for self-driving vehicles (Slavic et al., 2021).

Predicting and modeling interacting agents need to consider the movement trajectory and the type of interaction between them. Studies have been proposed and developed to model the interaction between pedestrians (Ridel et al., 2018) and vehicles (Lefèvre et al., 2014; Zhan et al., 2018). Further in (Kanapram et al., 2021), a situational collective awareness model to learn collective interaction for joint state estimation is performed. However, the authors only use low dimensional odometry data. These studies focus on the detection of anomalies while predicting the trajectory of a single ego vehicle. Even if they can detect and interpret the anomalies to a certain extent, those models still have limitations in modeling the interactions between agents by using visual information.

By taking inspiration from (Fraccaro et al., 2017), we propose to localize interacting agents from video data, using a *relative distance* metric. The proposed method utilizes heterogeneous sensorial information to estimate the relative distance at each time step and use it to predict the relative location of an interacting agent. A data-driven unsupervised approach is proposed for the localization of interacting agents. The representations of multi-agent interaction models are learned from multi-sensorial modalities in the form of multi-agent H-DBNs (MAH-DBN). H-DBNs, as we described them in Section 3.3, are robust in representing sequential date that can be used for anomaly detection (Slavic et al., 2022), continual learning (Ravanbakhsh et al., 2021), lower-dimensional interaction modeling (Baydoun et al., 2019), and for collective awareness (CA) in IoT devices (Kanapram et al., 2021). H-DBNs also have advantages in predicting intentions and trajectories simultaneously (Slavic et al., 2021).

From a statistical point of view, H-DBNs have the capabilities to model an environment in an explainable way. In addition, H-DBNs are appropriate options for representing multi-sensory information in a hierarchical form. In these representation models, to make inferences and anomaly signal detection, a Markov Jump Particle Filter (MJPF) (Iqbal et al., 2019b) can be used. As can be refereed in Section 3.4.3, MJPF is a combination of Kalman Filters (KFs) and Particle Filters (PFs). KFs and PFs are used to make inferences at the continuous and discrete levels of H-DBNs, respectively.

In this chapter, we describe a modified MJPF that integrates multi-agent video and odometry vocabularies for the localization of interacting agents using video data. The main contributions are: *i)* a data-driven approach, with a novel integration with a Bayesian inference model, is proposed for the localization of the interacting agents based on the video modality; *ii)* the learned multi-sensorial modalities, i.e., odometry and video, are represented as a MAH-DBN; *iii)* the proposed method is validated by developing a modified MJPF that is tested with CARLA simulated data and a real interacting iCab vehicles data.

6.2 Multi-Modality Learning Framework

The proposed method has two main modules. The first module is the Offline training module, a multi-sensory-multi-agent H-DBN module learned from odometry and video data sets of interacting agents. The second online testing module is the multi-modal combined MJPF which fuses learned vocabularies of video and odometry modules.

We consider the sensory observations x from two modalities of moving vehicles at each time instant t, i.e., video acquired from an onboard camera (First Person Viewpoint - FPV) x^{v}_{t,u_i} and the corresponding odometry trajectories of vehicles x^{o}_{t,u_i}, where $u_i = \{u_1, u_2\}$ are two unmanned ego vehicles; v and o indicates the video and odometry modality, respectively. As mentioned above the proposed methodology is divided into two parts: *i) Training phase*: the training datasets are used to learn models to predict future odometry and video states. We leverage the odometry data and learned rules of the odometry modality called odometry vocabularies to focus or lead the learning process of the video sequences accordingly. *ii) Testing phase*: the learned models (both video and odometry vocabularies) are tested with datasets obtained from different scenarios. Here, predictions that diverge from the actual observed values indicates the presence of an anomalous activity. More importantly during the testing phase, odometry data is no longer used; we employ only the properties of learned probability distributions (called vocabularies) of the odometry modality to localize our interacting agents solely from the video modality. fig. 6.1 displays the block diagram of the training and testing phase of the proposed methodology.

6.2.1 Learning Lower Dimensional Vocabulary

The sensory observations of the odometry module is comprised of the positional information of the agent such as $z^{o}_{t,u_i} = [\mathrm{x}_{t,u_i}, \mathrm{y}_{t,u_i}]$. Null Force Filter (Iqbal et al., 2021) is employed to obtain the first order Generalized State (GS) $\tilde{z}^{o}_{t,u_i}$, which comprises of position and velocity corresponding to each agent, i.e., $\tilde{z}^{o}_{t,u_i} = [\mathrm{x}_{t,u_i}, \mathrm{y}_{t,u_i}, \mathrm{v}_{\mathrm{x}_{t,u_i}}, \mathrm{v}_{\mathrm{y}_{t,u_i}}]$. We further process the GSs to have a semantic category (cluster) depicting the movement modality of both agents using the Growing Neural Gas (GNG) algorithm (Fritzke, 1994).

Null Force Filter or Unmotivated Kalman Filter is a family of Kalman filtering where we assume there is no force applied on the object, hence the velocity is considered to be zero. It

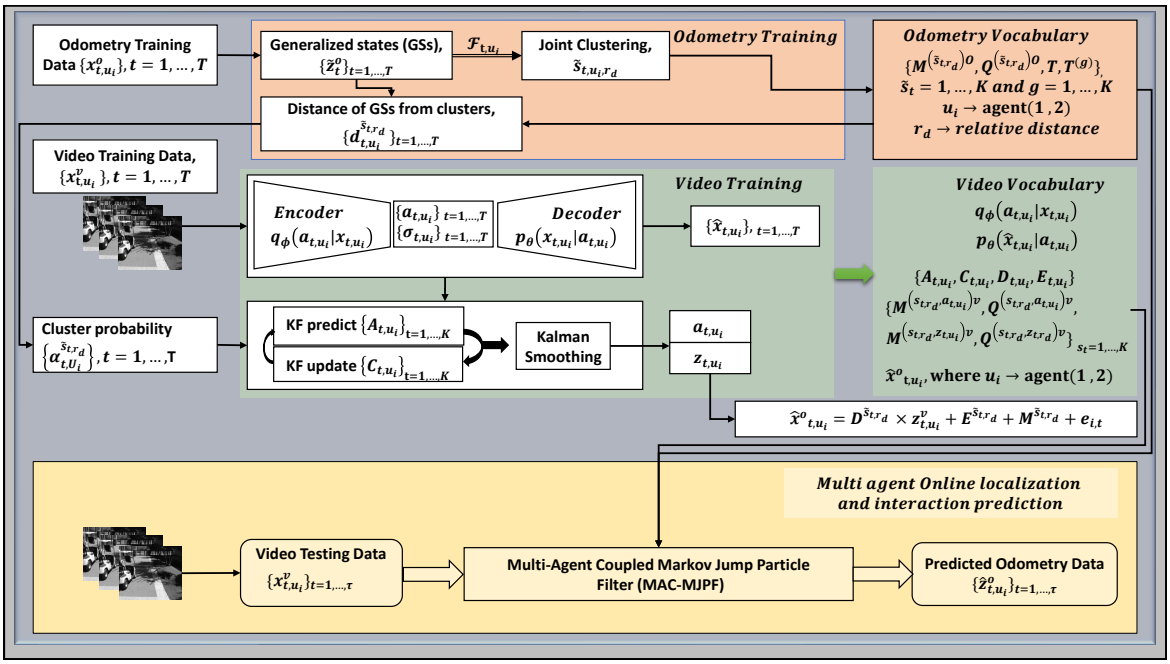


Figure 6.1: Block diagram of the proposed method. This architecture can support more than two autonomous agent but here we used it for unmanned vehicles u_1 and u_2. © 2022 IEEE.

is in line with section 3.4.1, where the dynamic model (eq. (3.7)) is modified as:

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \dot{x_t}_{+1} \\ \dot{y_t}_{+1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \triangle & 0 \\ 0 & 1 & 0 & \triangle \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_t \\ y_t \\ \dot{x}_t \\ \dot{y}_t \end{pmatrix} + v_t \tag{6.1}$$

Eq. (6.1) is stating that except some noise, the object (agent) is at rest (steady state). But here we are learning from sequences of positional information which are actually are changing in time. NFF considers the changes in position as generalized errors (GEs), which actually are velocity information.

Here we are modeling interacting agents, hence the feature vector is eight-dimensional, $\mathscr{F}^o_{t,u_1}$ wrt u_1 is provided as an input for the clustering algorithm, which can be written as;

$$\mathscr{F}^o_{t,u_1} = [\mathrm{x}_{t,u_1}, \mathrm{y}_{t,u_1}, \mathrm{v}_{\mathrm{x}_{t,u_1}}, \mathrm{v}_{\mathrm{y}_{t,u_1}}, \triangle \mathrm{x}_{t,u_{12}}, \triangle \mathrm{y}_{t,u_{12}}, \triangle \mathrm{v}_{\mathrm{x}_{t,u_{12}}}, \triangle \mathrm{v}_{\mathrm{y}_{t,u_{12}}}], \tag{6.2}$$

where $\{\triangle \mathrm{x}_t, u_{12}, \triangle \mathrm{y}_t, u_{12}\}$ and $\{\triangle \mathrm{v}_{\mathrm{x}_{t,u_{12}}}, \triangle \mathrm{v}_{\mathrm{y}_{t,u_{12}}}\}$ are the relative positions and velocities of the interacting vehicle u_1 to the neighbouring vehicle u_2, which is computed as

follows;

$$\begin{aligned}
\triangle \mathrm{x}_{t,u_{12}} &= \mathrm{x}_{t,u_1} - \mathrm{x}_{t,u_2}, \\
\triangle \mathrm{y}_{t,u_{12}} &= \mathrm{y}_{t,u_1} - \mathrm{y}_{t,u_2}, \\
\triangle \mathrm{v}_{\mathrm{x}_{t,u_{12}}} &= \mathrm{v}_{\mathrm{x}_{t,u_1}} - \mathrm{v}_{\mathrm{x}_{t,u_2}}, \\
\triangle \mathrm{v}_{\mathrm{y}_{t,u_{12}}} &= \mathrm{v}_{\mathrm{y}_{t,u_1}} - \mathrm{v}_{\mathrm{y}_{t,u_2}}.
\end{aligned} \tag{6.3}$$

As depicted in fig. 6.2 for each cluster $\tilde{s}_t$, where $t = 1 \ldots K$ are the number of clusters. Cluster or super state properties, that are commonly called vocabularies includes: *i)* the cluster mean $M^{(\tilde{s}_k)O}$; *ii)* the cluster covariance $Q^{(\tilde{s}_k)O}$; *iii)* a transition matrix T defining the probability of transition from one cluster to the other; *iv)* a transition time $T^{(g)}$ defining the time spent in each cluster before switching (moving) to the next cluster can be learned in an unsupervised machine learning manner. H-DBNs (see fig. 6.2) are used as a deep or a hierarchical generative models, to represent the learned probabilistic conditional distribution of each agent between two consecutive time instants, $t-1$ and t (previous state and current state), so that we can predict the next state $t+1$ as a semi- Markovian models.

The trajectory prediction of overtaking vehicles aims to estimate the probability of the future states by using the previous states and the relative positions wrt interacting vehicles. Therefore, this prediction problem can be learned as conditional inference;

$$p(z_{t,u_1}|z_{t-1,u_1}) = p(z_{t,u_1}|z_{t-1,u_1}, \tilde{s}_{t-1,u_1}), \tag{6.4}$$

$$p(z_{t,u_2}|z_{t,u_1}, r_{d_t,u_{12}}) = p(z_{t,u_1}|z_{t,u_2}, r_{d_t,u_{12}}, \tilde{s}_{t-1,u_1}), \tag{6.5}$$

where $r_{d_t,u_{12}}$ is a relative distance vector, which can be defined as;

$$r_{d_t,u_{12}} = [\triangle x_{t,u_{12}}, \triangle y_{t,u_{12}}, \triangle v_{x_{t,u_{12}}}, \triangle v_{y_{t,u_{12}}}]. \tag{6.6}$$

Eq. (6.4) and (6.5) are representing the state dynamic model of the interacting agents in a probabilistic way. Eq. (6.4) particularly depicts the state evolution of the overtaking ego vehicle u_1 from the discrete and continuous hidden states of itself as graphically modeled in fig. 6.2. Eq. (6.5) models the time evolution of interacting ego vehicle from the view-point of the neighbouring ego vehicle. Note, if we change the interaction model from agent two to agent one, the subscripts representing each agent will be changed or swapped accordingly.

6.2.2 Higher Dimensional Vocabulary Learning

After learning from the odometry modality, we used the learned vocabulary to train the video sequences. For this purposes, a Kalman Variational Autoencoder (KVAE) (Fraccaro et al.,

2017) is employed to obtain the latent states $a_t^{(u_i)}$ corresponding to video of the training data x^v_{t,u_i}. To elaborate more, we have two hidden states, i.e., a_{t,u_i} and z_{t,u_i}, which represent the content of an image and its corresponding dynamics, respectively. While training, since our main objective is to localize both agents only from the video sequence at each time instant, we learned other pseudo observation matrices, that are, D and E, from the latent distribution and the odometry vocabulary. These matrices encode the information of the location parameters of interacting agents. Therefore, in general, we learned four matrices (A, C, D, and E) that represent our complex model. Trained odometry vocabulary $(\tilde{s}^o_{t,u_i})$ is employed to guide the model to assign a semantic category to the latent states. Mathematically these two latent states are modeled as follows;

$$a_{t,u_i} = \sum_{k=1}^{K} \alpha_{t,k,u_i} \times C_{k,u_i} \times z_{t,u_i} + v_{t,u_i}, \tag{6.7}$$

where a_{t,u_i} is the content distribution that is obtained from the dynamical latent state z_{t,u_i} as an observation model, whereas z_{t,u_i} at consequent time instants are connected through the dynamical model; such as,

$$z_{t+1,u_i} = \sum_{k=1}^{K} \alpha_{t,k,u_i} \times A_{k,u_i} \times z_{t,u_i} + \omega_{t,u_i}, \tag{6.8}$$

where v_t and ω_t are the Gaussian noises; matrices $\{C_{k,u_i}\}_{k=1\ldots K}$ and $\{A_{k,u_i}\}_{k=1\ldots K}$ represent a set of pseudo-observation models and transition models, respectively, as defined in the traditional Kalman Filter (KF) (Ravanbakhsh et al., 2021). These models are combined through a probabilistic vector $\alpha^{\tilde{s}_k}_{t,u_i}$ as a *guiding vector*, which guide the model to assign a semantic category (cluster) to the dynamic video state z_{t,u_i}, mathematically it can be defined as follows;

$$1/\alpha^{\tilde{s}_k}_{t,u_i} = \left(d^{\tilde{s}_k}_{t,u_i}\right)^n \times \sum_{k=1}^{K} \frac{1}{\left(d^{\tilde{s}_k}_{t,u_i}\right)^n}, \tag{6.9}$$

where $d^{\tilde{s}_k}_{t,u_i} = \mathscr{D}_M((z^v_{t,u_i}), (M^{(\tilde{s}_k)o}, Q^{(\tilde{s}_k)o}))$ is a Mahalanobis distance ($\mathscr{D}_M$) (Xiang et al., 2008) which is calculated from each latent state distribution z^v_{t,u_i} to the probability distribution of each cluster of the odometry module $(M^{(\tilde{s}_k)o}), Q^{(\tilde{s}_k)o})$, and n is a temperature value to make the probabilities of the selected cluster more probable for each video sequence. The value of n is selected according to table I, in Chapter 7. In first iteration, z^v_{t,u_i} will be assigned to the superstates corresponding to the shortest distance $d^{\tilde{s}_k}_{t,u_i}$ values. After the first iteration, $d^{\tilde{s}_k}_{t,u_i}$ will be multiplied by the previous $\alpha^{\tilde{s}_k}_{t,u_i}$ value to cluster the z^v_{t,u_i}, having similar probability distributions.

As a result, we will have two sets of vocabularies from both odometry and video modalities.

Odometry Vocabularies: Cluster mean $(M^{(\tilde{s}_k)o})$, cluster covariance $(Q^{(\tilde{s}_k)o})$, transition probability (T) and time in each cluster $(T^{(g)})$.

Video Vocabularies: Cluster mean ($M^{(s_k,a_t)v}$, $M^{(s_k,z_t)v}$) and cluster covariance ($Q^{(s_k,a_t)v}$, $Q^{(s_k,z_t)v}$)

While training, we also learned the *pseudo observation vectors* $\hat{z}^o_{t,u_1}, \hat{z}^o_{t,u_2}$ as follows:

$$\hat{z}^o_{t,u_1} = D^{\tilde{s}_k} \times z^o_{t,u_1} + E^{\tilde{s}_k} + M^{\tilde{s}^O_k} + e_{1_t}, \tag{6.10}$$

$$\hat{z}^o_{t,u_2} = z^o_{t,u_1} + r_{d_t,u_{12}} + e_{2_t}, \tag{6.11}$$

where the prediction of $\hat{z}^o_{t,u_1}$ and $\hat{z}^o_{t,u_2}$ are performed by minimizing the errors (e_{1_t}, e_{2_t}) w.r.t. the ground truth training odometry trajectory and the predicted parameters. Optimization is performed through minimization of the following losses:

$$L_{\delta,u_1} = \begin{cases} \frac{1}{2}(\tilde{z}_{t,u_1} - \hat{z}_{t,u_1})^2 & if\,|(\tilde{z}_{t,u_1} - \hat{z}_{t,u_1})| < \delta \\ \delta((\tilde{z}_{t,u_1} - \hat{z}_{t,u_1}) - \frac{1}{2}\delta) & otherwise \end{cases}, \tag{6.12}$$

$$L_{\delta,u_2} = \begin{cases} \frac{1}{2}(\tilde{z}_{t,u_2} - \hat{z}_{t,u_2})^2 & if\,|(\tilde{z}_{t,u_2} - \hat{z}_{t,u_2})| < \delta \\ \delta((\tilde{z}_{t,u_2} - \hat{z}_{t,u_2}) - \frac{1}{2}\delta) & otherwise \end{cases}, \tag{6.13}$$

where L_{δ,u_1} and L_{δ,u_2} are the *Huber Loss functions* (Hastie et al., 2009) with $\delta = 1$. During the (online) testing phase, the learned vocabularies s^v_{k,u_1}, $\tilde{s}^o_{k,u_1}$ and $\hat{z}^o_{t,u_1}$, $\hat{z}^o_{t,u_2}$ are employed to predict the trajectories of interacting agents.

6.3 Multi-modality Interaction Modeling

In this section, we are going to tackle both localization and interaction predictions of an agent simultaneously. Since we want to achieve these capabilities only from the video information of each agent; the learned *pseudo-observation vectors* (see eq. (6.10) and eq. (6.11)) are employed. The *Multi-Agent Coupled Markov Jump Particle Filter* (MAC-MJPF) is proposed for the prediction of the trajectories ($\tilde{z}^o_{t,u_1}$,$\tilde{z}^o_{t,u_2}$) at the online phase. MAC-MJPF employs KF and PF to make inferences of the future states over the proposed H-DBN model. The latent states features of u_i are obtained by employing KVAE. Here, the KF is used to predict the

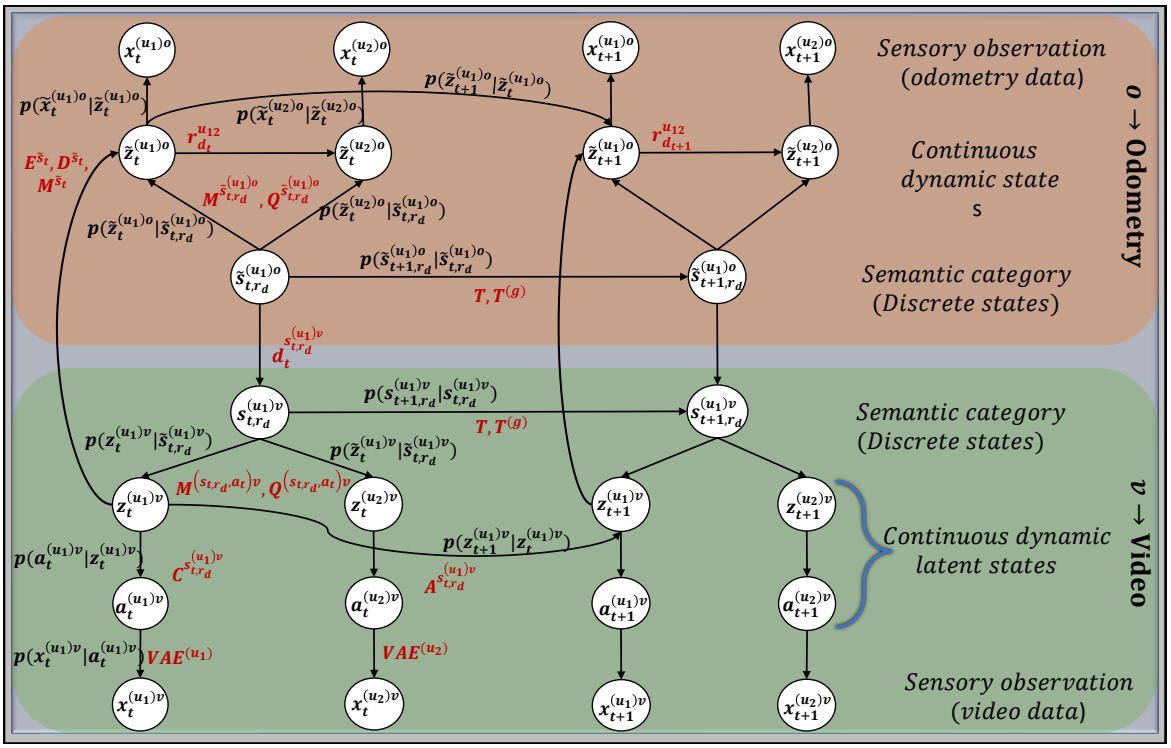


Figure 6.2: Multi-sensory-multi-agent learned Hierarchical Dynamic Bayesian Network (H-DBN) situation model. © 2022 IEEE.

future latent state $z^{v}_{t+1|t,u_1}, \Sigma^{v}_{t+1|t,u_1}$ corresponding to each particle n of PF. The total number of particles is N, and it can be determined based on table V of Chapter 7. This latent state is then used to predict the location of u_1 (itself), i.e., $\tilde{z}^{o}_{t+1|t,u_1}$, as well as $r_{d_t,u_{12}}$ by using KF. Once the location of u_1 is estimated, the next task is to observe the environment and estimate the GSs of the interacting agent. Based on the predicted $r_{d_t,u_{12}}$, u_1 estimates the location of the interacting agent u_2 by mapping $r_{d_t,u_{12}}$. At each time instant, the update stage of KF updates the predicted latent states $z^{v}_{t|t,n,u_1}$. Similarly, the predicted GSs $\tilde{z}^{o}_{t+1|t,u_1}, \tilde{z}^{o}_{t+1|t,u_2}$ are also updated based on the learned *pseudo-observation vectors* $(\hat{z}^{o}_{t,u_1}, \hat{z}^{o}_{t,u_2})$ see eq. (6.10) and eq. (6.11).

For a better clarification and understanding of the mathematical model, as well as the algorithmic detail, we describe each step in Algorithm 1. One thing to note is that, for algorithm readability, the subscripts that describe the agents are changed to superscripts. As an example, $z^{v}_{t|t,n,u_1}$ (the video state of agent u_1 at time t for a particular filter n) is written as $z^{(u_1)v}_{t|t,n}$. The total number of particles (N), can be determined based on the complexity analysis shown in table V of Chapter 7.

Algorithm 1: Localization and Interaction algorithm.

Require: $M^{(\tilde{s}_k)o}, Q^{(\tilde{s}_k)o}, T, T^{(g)}, M^{(s_k,a_t)v}, M^{(s_k,z_t)v}, Q^{(s_k,a_t)v}, Q^{(s_k,z_t)v}, x_t^{(u_1)v}$

for t = 1 to τ (Time evolution) do
  Obtain image latent state using VAE: $a_t^{(u_1)v}$, $\sigma_t^{(u_1)v}$ and
  Obtain latent state covariance $\leftarrow \Sigma_t^{(u_1)v} \sim I_L \times \sigma_t^{(u_1)v}$, where L is the number of latent state features
  Compute distances of video encodings from video clusters: $d_t^{s_k^{(u_1)v}} = \mathscr{D}_B((a_t^{(u_1)v}, \Sigma_t^{(u_1)v}), (M^{(s_k,a_t)v}, Q^{(s_k,a_t)v}))$
  for $s = 1, ..., K$
  Calculate probabilistic *guiding vector* $\alpha_t^{(\tilde{s}_k^{u_1})}$ using eq. (6.9)
  Begin Filtering
  if time step == 1
    for n = 1 to N $\leftarrow$ number of Particles
      $W_n = \frac{1}{N} \leftarrow$ weight of the particles
      Initialize the video and odometry vocabularies:
      $s_t^{(u_1)v} \sim p(z_t^{(u_1)v} | s_t^{(u_1)v})$
      $\tilde{s}_t^{(u_1)o} \sim p(\tilde{z}_t^{(u_1)o} | \tilde{s}_t^{(u_1)o})$
  Else
    UPDATE:
      for n = 1 to N Particles
        Update video states:
        $z_{t|t,n}^{(u_1)v}, \Sigma_{t|t,n}^{(u_1)v} = KF_{update}^{Video}(z_{t|t-1,n}^{(u_1)v}, \Sigma_{t|t-1,n}^{(u_1)v}, C_{t,n}^{s_t^{u_1}}, a_{t,n}^{(u_1)v}, \Sigma_{t,n}^{(u_1)v})$
        Update $\tilde{z}_{t|t,n}^{(u_1)o}$ (itself) according to eq.(6.10)
        Update $\tilde{z}_{t|t,n}^{(u_2)o}$ according to eq.(6.11)
      Re-sample if $\sum_n \frac{1}{W_n^2} <$ threshold
  PREDICTION:
    for n = 1 to N Particles
      Predict vocabulary of video:
      $s_{t+1,n}^{(u_1)v} \sim T(\tilde{s}_{t,n}^{o})$,
      Predict vocabulary of odometry:
      $\tilde{s}_{t+1,n}^{(u_1)o} \sim T(\tilde{s}_{t,n}^{o})$,
      Predict Video latent state:
      $z_{t+1|t,n}^{(u_1)v}, \Sigma_{t+1|t,n}^{(u_1)v} = KF_{pred}^{Video}(z_{t|t,n}^{(u_1)v}, \Sigma_{t|t,n}^{(u_1)v})$
      Predict location of u_1 (itself) and $r_{d_t}^{u_{12}}$ w.r.t. interacting agent u_2 :
      $\tilde{z}_{t+1|t,n}^{(u_1)o}, \Sigma_{t+1|t,n}^{(u_1)o}, r_{d_t}^{u_{12}} = KF_{pred}^{Odometry}(z_{t|t,n}^{(u_1)v}, \Sigma_{t|t,n}^{(u_1)v})$
      Predict location of neighboring agent u_2 using $r_{d_t}^{u_{12}}$:
      $\tilde{z}_{t+1|t,n}^{(u_2)o}, \Sigma_{t+1|t,n}^{(u_2)o} = \tilde{z}_{t+1|t,n}^{(u_1)o}, \Sigma_{t+1|t,n}^{(u_1)o}) + r_{d_t}^{u_{12}}$
Smoothing: $\tilde{z}_{t|t-1}^{(u_1)o} = \mathbb{E}[\tilde{z}_{t|t-1}^{(u_1)o}]_n$ and $\hat{z}_{t|t-1}^{(u_2)o} = \mathbb{E}[\tilde{z}_{t|t-1}^{(u_2)o}]_n$
Output: Predicted trajectories $\tilde{z}_{t|t-1}^{(u_1)o}$ (oneself) and the interacting agent $\tilde{z}_{t|t-1}^{(u_2)o}$.

6.4 Used Datasets

Two semi-autonomous vehicles with the names *iCab*1 and *iCab*2 (Marín-Plaza et al., 2016) are employed to perform the overtaking experiments. Since our objective is the localization of the interacting agents, we considered an overtaking maneuver in which $iCab2 \sim u_2$ overtakes $iCab1 \sim u_1$ while interacting.

In addition to the above datasets we also tested our model on data from the well-known CARLA simulator (Dosovitskiy et al., 2017) which is built from the ground to support development, training, and validation of autonomous driving systems. To make it coherent with our proposed method, we performed an overtaking scenario by programming agents to

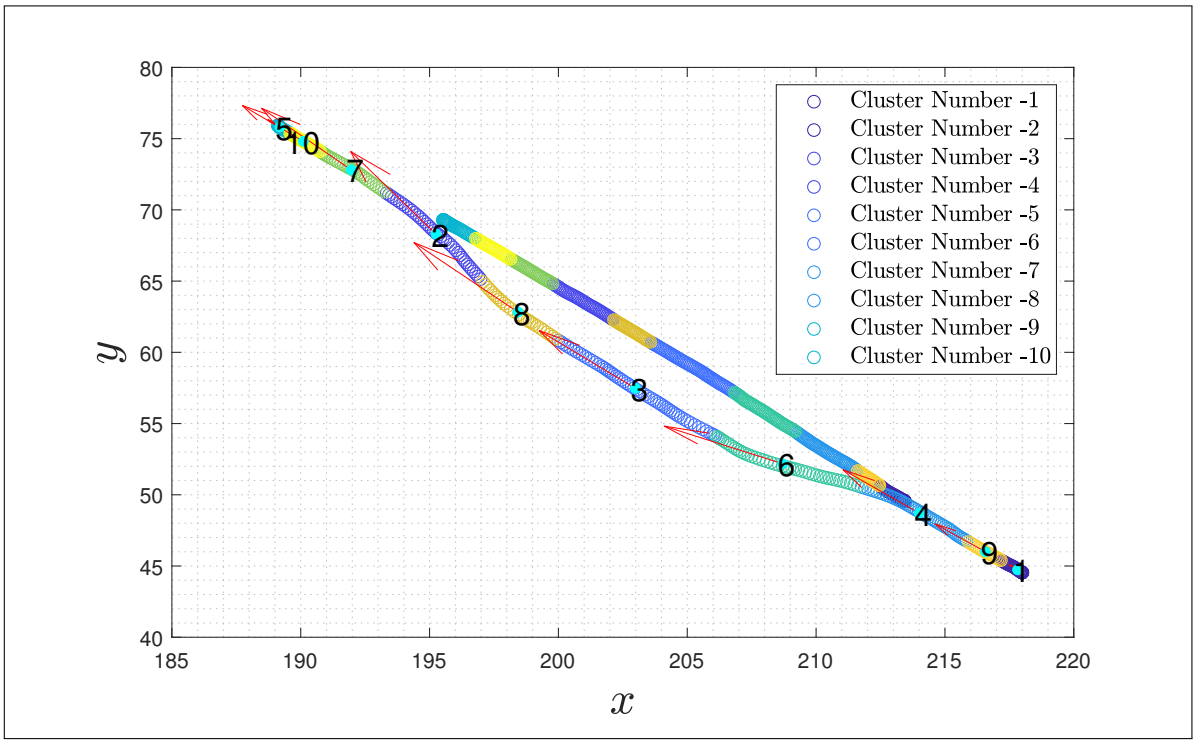


Figure 6.3: iCab clustered positional overtaking trajectories of the interacting agents. Arrows indicate the direction of motion of vehicles encoded in $\tilde{s}_{k,u_i}$ and different colors represent the different clusters. The filled cyan colors represents the mean position of each cluster. © 2022 IEEE.

collect the multi-modal training and testing simulation-based odometry and video datasets. Maneuvering of CARLA agents are in line with the above iCab based datasets, which are pre-processed to synchronize video and odometry datasets and the data between the two agents.

6.5 Results

The input feature vector from the odometry modality, i.e., $\mathscr{F}_t^{(u_1)O}$ (see eq. (6.2)) is clustered to learn the odometry vocabulary, $\tilde{s}_k^{u_1}$. figs. 6.3 and 6.4 show the clustered trajectory where different colors indicate the different clusters. The odometry vocabulary is employed to guide the clustering of the latent states by computing $\alpha_t^{(\tilde{s}_k^{u_1})}$. In online testing, the learned vocabulary is employed to predict the positional trajectories of the interacting agents.

Figs. 6.6 a and 6.6 b depict the predicting trajectory of u_2 (itself) and its interacting agent u_1. Similarly, u_1 is also interacting with u_2, therefore the trajectory of u_2 can be estimated from u_1. Figs. 6.7 a, and 6.7 b indicate the predicted trajectory of u_1 (itself) and

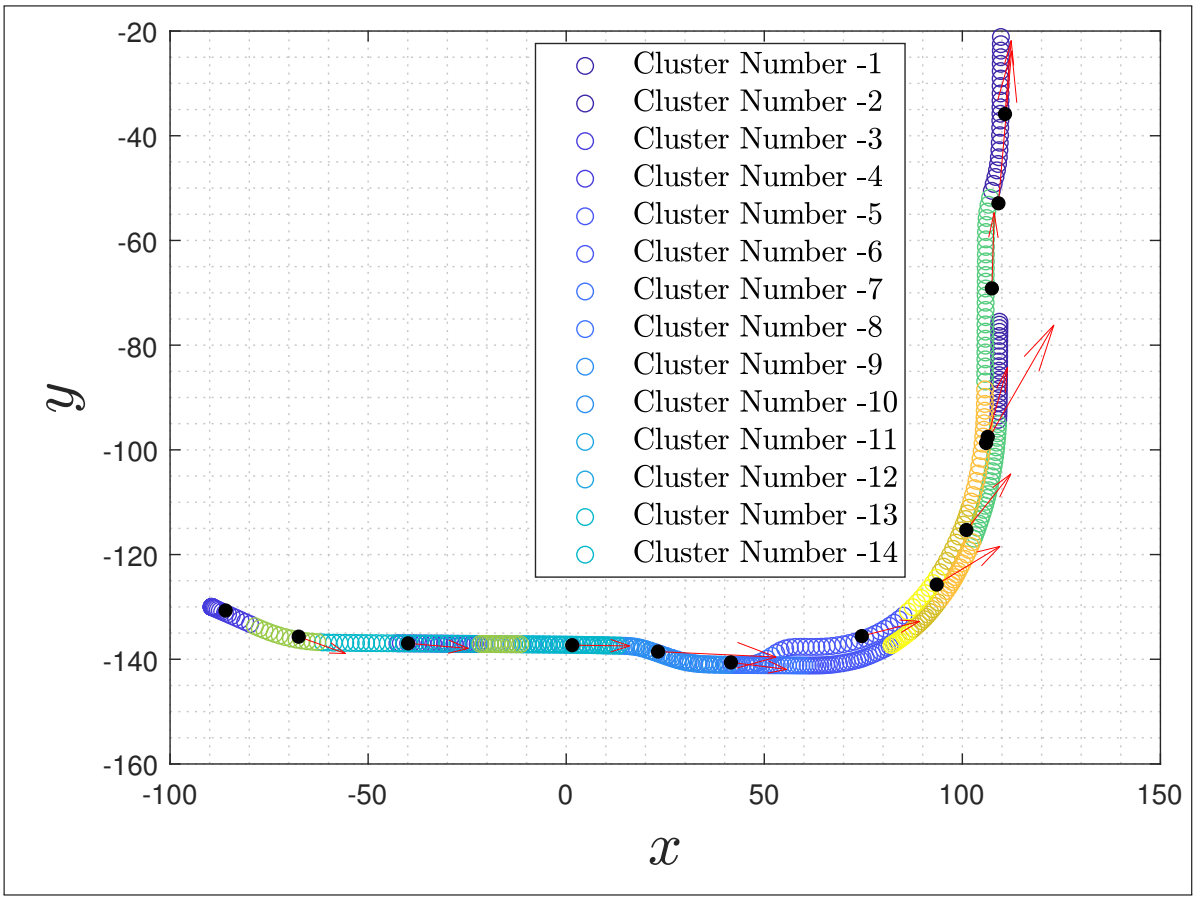
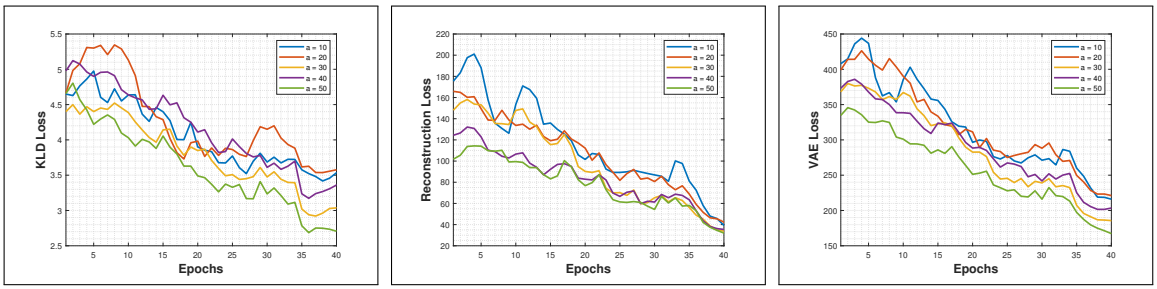


Figure 6.4: CARLA clustered positional overtaking trajectories of the interacting agents. Arrows indicate the direction of motion of vehicles encoded in $\tilde{s}_{k,u_i}$ and different colors represent the different clusters. The filled black colors represents the mean position of each cluster. © 2022 IEEE.



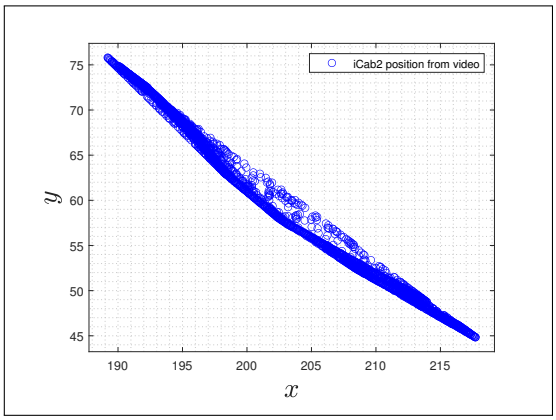
(a) KLD Losses (b) Reconstruction Losses (c)VAE Losses

Figure 6.5: (a) KLD, (b) Reconstruction, and (c) Total VAE losses from the iCab validation dataset of iCab2, for different content related latent state (a) dimensions: $a \in \left(\mathbb{R}^{10}, \mathbb{R}^{20}, \mathbb{R}^{30}, \mathbb{R}^{40}, \mathbb{R}^{50}\right)$.

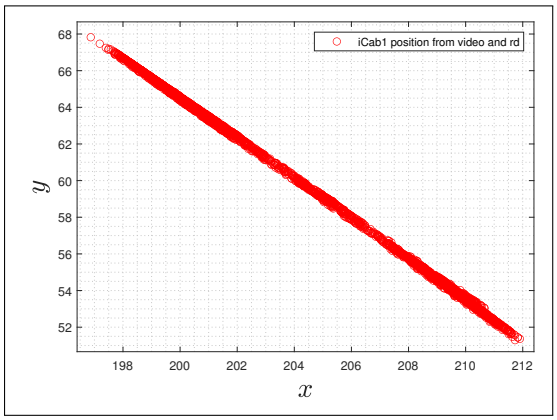
the interacting agent u_2 from the surroundings. The trajectories of the interacting agents u_2 and u_1 are estimated by employing the relative distance $r_{d_t}^{u_{12}}$ and $r_{d_t}^{u_{21}}$, respectively.

Table I: Analysis of latent state size influence on the relation of odometry and video vocabularies

Latent state dimension	a $\in \mathbb{R}^{10}$	a $\in \mathbb{R}^{20}$	a $\in \mathbb{R}^{30}$	a $\in \mathbb{R}^{40}$	a $\in \mathbb{R}^{50}$
iCab experiments	0.7054	0.7500	0.7679	0.7643	0.7679
CARLA experiments	0.8422	0.8203	0.8063	0.8297	0.8547

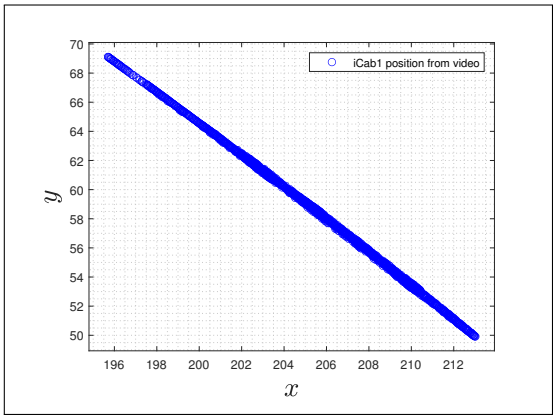


(a) Predicted u_2 (itself) trajectory from video sequences.

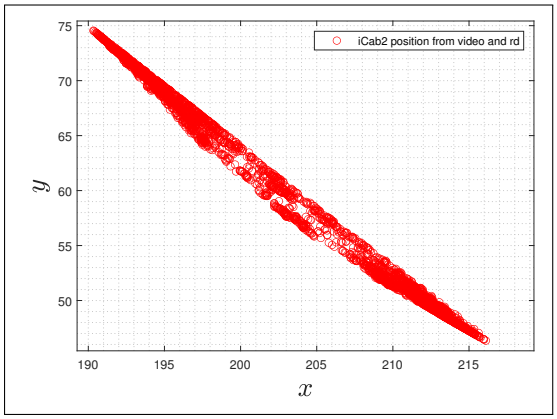


(b) Predicted u_1 trajectory from video sequences and relative distance vector.

Figure 6.6: (a) Predicted trajectories of iCab agent u_2, only from video modality. (b) Localization of iCab agent u_1, from u_2 based on $r_{d_t}^{u_{12}}$. © 2022 IEEE.



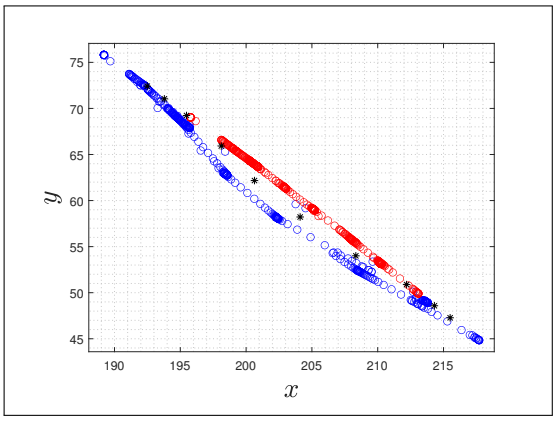
(a) Predicted u_1 (itself) trajectory from video sequences.



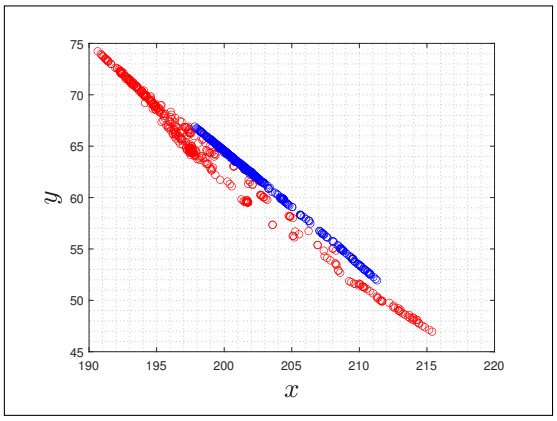
(b) Predicted u_2 trajectory from video sequences and relative distance vector.

Figure 6.7: (a) Predicted trajectories of iCab agent u_1, only from video modality. (b) Localization of iCab agent u_2, from u_1 based on $r_{d_t}^{u_{21}}$. © 2022 IEEE.

Figs. 6.8 a and 6.8 b plot the smoothed trajectories of both agents by superimposing them in the same coordinates. The blue color is the prediction of oneself trajectory and the red color is the localization of the interacting agents from the surroundings.

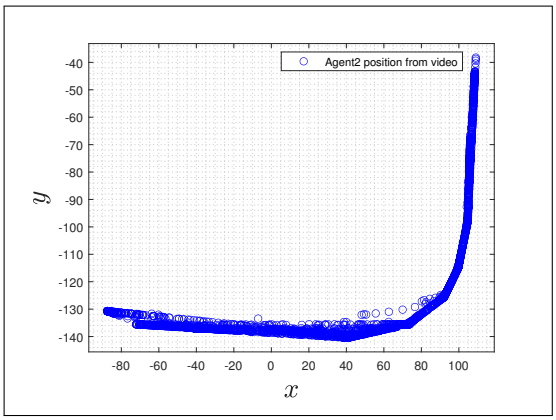


(a) Smoothed trajectories $\tilde{z}$, when the overtaking agent predicts interaction trajectory.

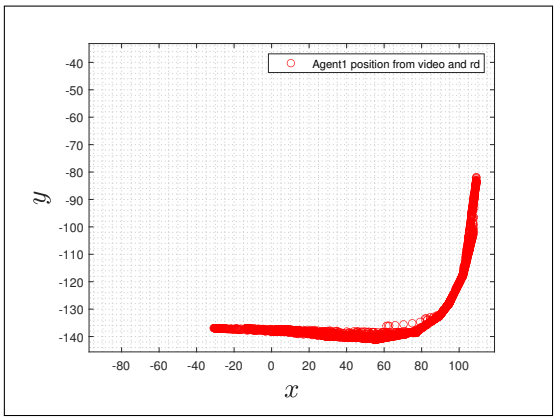


(b) Smoothed trajectories $\tilde{z}$, when the interacting agent predicts interaction trajectory.

Figure 6.8: (a, b) Show the smooth predicted trajectories of itself (blue) and of the interacting agents (red) using the iCab dataset. © 2022 IEEE.



(a) Predicted u_2 (itself) trajectory from video sequences.

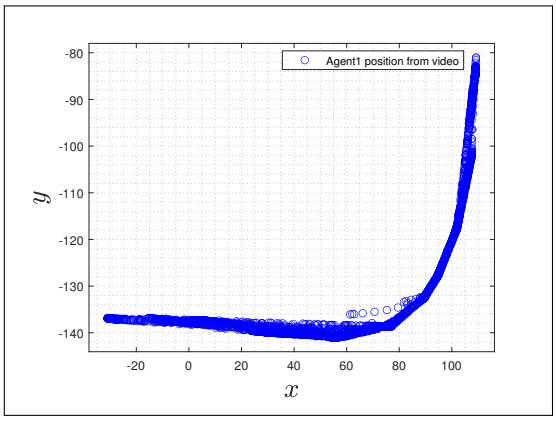


(b) Predicted u_1 trajectory from video sequences and relative distance vector.

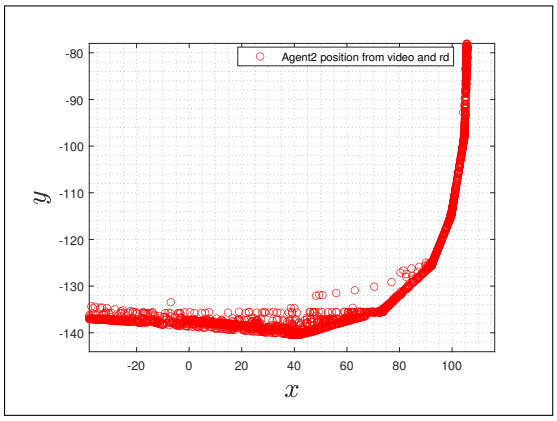
Figure 6.9: (a) Predicted trajectories of CARLA agent u_2, only from video modality. (b) Localization of CARLA agent u_1, from u_2 based on $r_{d_t}^{u12}$. © 2022 IEEE.

From the CARLA simulator dataset we similarly tested the interaction performance. In figs. 6.9 a and b, the predicted trajectory of u_2 (itself) and the interacting agent u_1 are presented respectively. Similarly, u_1 is also interacting with u_2 as their trajectories are shown in figs. 6.10 a and b. Smoothed trajectories superimposed in the same axis are shown in figs. 6.10.

Fig. 6.12 shows the predicted velocity at each time instant, indicating that the proposed model has an ability to predict the velocity of the agents along-with the prediction of the

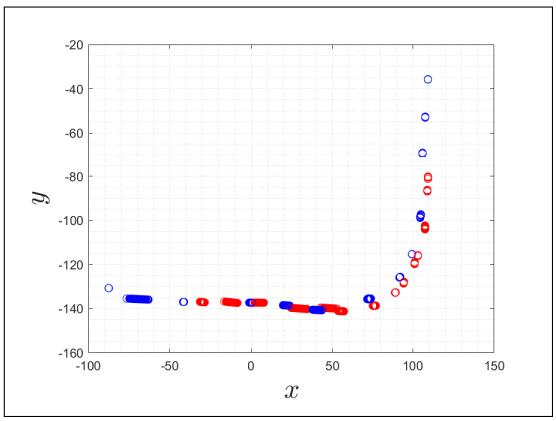


(a) Predicted u_1 (itself) trajectory from video sequences.

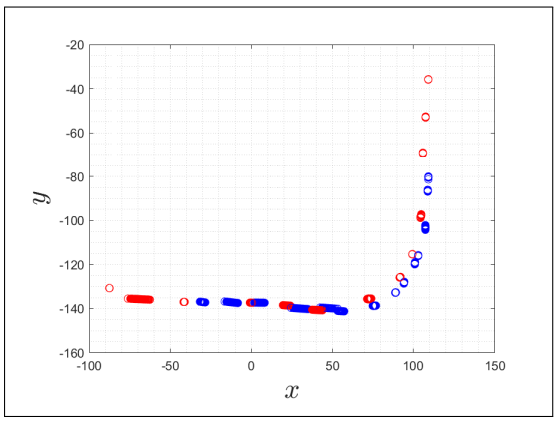


(b) Predicted u_2 trajectory from video sequences and relative distance vector.

Figure 6.10: (a) Predicted trajectories of CARLA agent u_1, only from video modality. (b) Localization of CARLA agent u_2, from u_1 based on $r_{d_t}^{u_{21}}$. © 2022 IEEE.



(a) Smoothed trajectories $\tilde{z}$, when the overtaking agent predicts interaction trajectory.



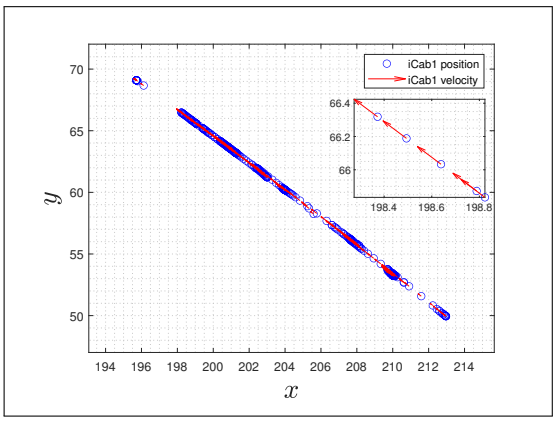
(b) Smoothed trajectories $\tilde{z}$, when the interacting agent predicts interaction trajectory.

Figure 6.11: (a, b) Show the smooth predicted trajectories of itself (blue) and of the interacting agents (red) utilizing the CARLA dataset. © 2022 IEEE.

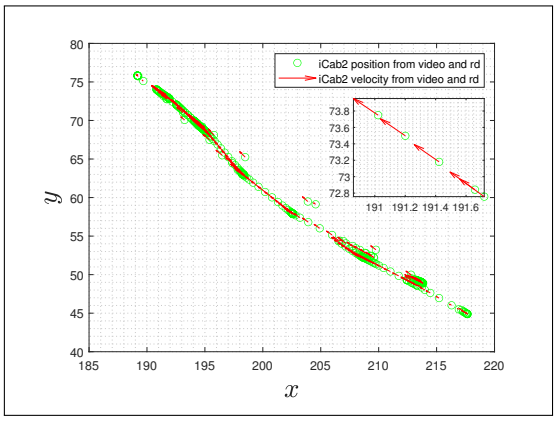
positional trajectories of itself as well as the interacting agent. Table II shows the RMSE of the r_d corresponding to the position and velocity of both agents, which is quite minimal.

In table II, we used the generalized training data as ground truth to compute the errors. The ground truth for Agent 1 and Agent 2, with the CARLA simulator dataset, is shown in fig. 6.4. Also for iCab 1 and iCab 2 (considering real iCab dataset), the ground truth is shown in fig. 6.3.

The performance of the interaction algorithm depends on the attention focusing while training, and experience modeling at the online inference stage. Hence the conformity

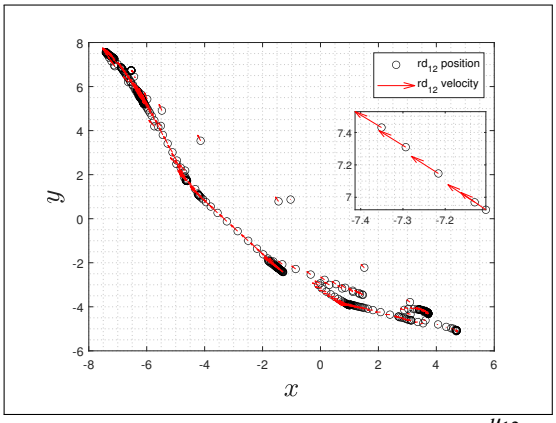


(a) Predicted position and velocity trajectory of u_1.

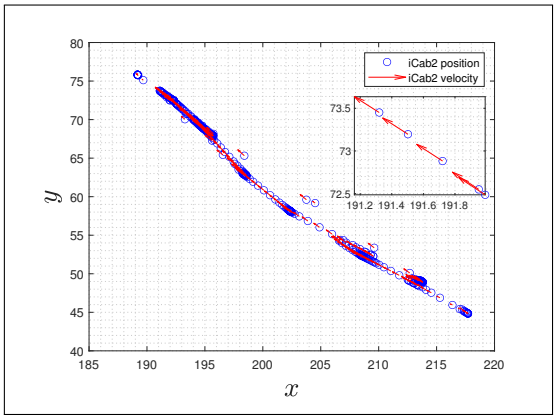


(b) Predicted position and velocity trajectory of u_2 from $r_{d_t}^{u_{12}}$

Figure 6.12: (a) Predicted GSs of u_1 (itself), (b) u_2 (interacting). The trajectories are smoothed to magnify the positional and velocity information. © 2022 IEEE.



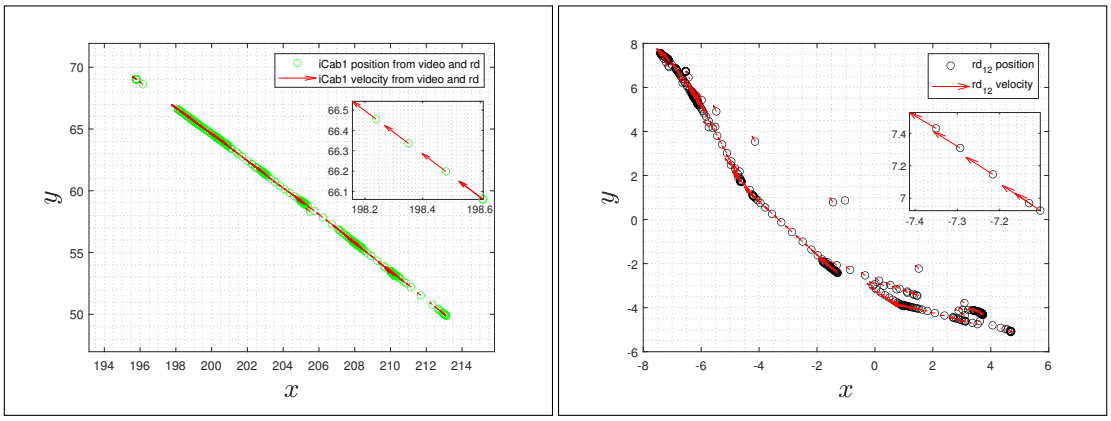
(a) Predicted relative distance vector $r_{d_t}^{u_{12}}$, trajectory.



(b) Predicted position and velocity trajectory of u_2

Figure 6.13: (a) smoothed trajectory of the relative distance vector $r_{d_t}^{u_{12}}$, (b) Predicted GSs of u_2 (itself) magnified to show trajectories of position and velocity. © 2022 IEEE.

between the higher-dimensional and lower-dimensional vocabularies are a crucial point. As shown in table I, the dimension of the content related latent state has a big impact in assigning the corresponding odometry cluster to each video frame sequences. When the latent state dimension is $a \in \mathbb{R}^{10}$, the network struggles to assign the correct cluster. As we increase the dimension, the correspondence between the odometry and video vocabularies gets better and better. Although increasing the dimension has a better coupling, it has a computational impact at the inference stage. Based on the VAE validation losses in figs. 6.15 and 6.5, dimensions $a \in \mathbb{R}^{20}$ and $a \in \mathbb{R}^{30}$ are recommended choices. Taking iCab 2 and the latent



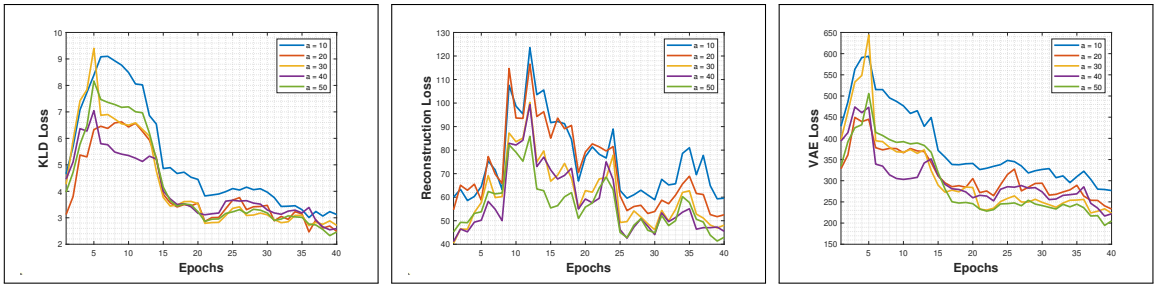
(a) Predicted position and velocity trajectories of u_1 from $r_{d_t}^{u_{21}}$.

(b) Predicted relative distance vectors $r_{d_t}^{u_{21}}$, from u_2 to u_1.

Figure 6.14: (a) Smoothed trajectory of u_1 (interacting) depicting position and velocity trajectories. (b) Relative distance vector trajectories $r_{d_t}^{u_{21}}$, illustrating the relative positions and relative velocities. © 2022 IEEE.

Table II: Table displays the quantitative analysis of the proposed method based on Root Mean Square Error (RMSE) measurement. We perform two different experiments to test our model, and obtained consistent results. © 2022 IEEE.

Experiments	**RMSE**			
	x	y	v_x	v_y
iCab1 experiment	6.1287	5.3117	0.0796	0.0686
iCab2 experiment	5.9369	7.1810	0.0455	0.0458
Agent1 experiment	6.1262	2.3529	0.0039	0.0044
Agent2 experiment	5.1235	1.8973	0.0022	0.0031



KLD Losses Reconstruction Losses VAE Losses

Figure 6.15: (a) KLD, (b) Reconstruction, and (c) Total VAE losses from the CARLA validation dataset of agent2, for different content related latent state (a) dimensions: $a \in \left(\mathbb{R}^{10}, \mathbb{R}^{20}, \mathbb{R}^{30}, \mathbb{R}^{40}, \mathbb{R}^{50}\right)$.

dimension $a \in \mathbb{R}^{30}$, the correspondence is $\approx 77\%$. The differences are mostly due to shadow presence in the image frames and sudden steering changes.

In this chapter and in Chapter 7, we used $a \in \mathbb{R}^{20}$. In Chapter 8, in the decision making stage, we faced a performance challenge, when we use $a \in \mathbb{R}^{20}$. Moreover, since we are using particle filters, to infer and act in this dimension, the performance speed was very low and delayed the action selection process. For this reason in Chapter 8, we used $a \in \mathbb{R}^{10}$.

6.6 Comparison with state-of-the-art trajectory estimation methods

DSV-SLAM (Mo et al., 2022) is a LiDAR descriptor-based SLAM approach to facilitate an efficient detection of loop closures for localization and mapping. They used the KITTI dataset and achieved RMSE of 3.693% on average. Other methods like UnDeepVO (Li et al., 2018) perform pose estimation by training deep neural networks in an unsupervised manner and attained an average of 4.07% RMSE drift. In (Li et al., 2019), they proposed a self-supervised learning framework for a representation of frame-to-frame correspondence for depth and pose estimation with RMSE drift of around 5%. However, to the best of our knowledge, we were unable to find a multi-agent localization and interaction framework to compare with our work. Therefore, we presented a comparison with the latest state-of-art methods of the localization of single-agent. The above discussion shows that our method allows to estimate the trajectories of interacting agents with RMSE drift which is comparable with the state-of-the-art methods.

6.7 Summary

In this chapter we proposed a data-driven approach for the localization of interacting agents based on multi-sensorial data. During offline training, the vocabulary of the low-dimensional modality is used to learn the latent states (and their dynamics) for the video modality, and obtain the corresponding video vocabulary. This training module allows the agents to build up autobiographical memories (AMs) incrementally from the fusion of video and odometry modalities. These AMs are learned in a hierarchical learning and representation networks (MAH-DBNs).

During online testing, the trajectories of the interacting agents are predicted from the video modality by employing the learned vocabulary. We show that the proposed MAH-DBNs are able to represent the interaction model robustly. The results show that the proposed MAC-MJPF inference algorithm applied in the learned MAH-DBN is able to localize and predict the GSs, i.e., position and velocity, of neighboring agents from the videos in an

unsupervised way. Next chapters will use this interaction scenario to build and test a sef-awareness architecture, for autonomous agents, validated through inference algorithm in Chapter 7 and concluded in Chapter 8 through online incremental learning and decision making algorithm.

Chapter 7

Modeling Interactions between Autonomous Agents in a Multi-Agent Self-Awareness Architecture

Self-awareness (SA) is a blending of situation understanding and conscious self expression when performing day to day activities. Self-awareness refers to a system's capability to recognize its state, possible actions, and the result of these actions on the system itself and its environment (Schlatow et al., 2017). Building self-awareness for artificial agents requires to understand multiple functions with different criticality level. These issues need to be designed well so as a self-aware system can manage different unexpected events that it encounter. In autonomous driving, these functionalities must be integrated with hardware and software platforms to make an agent reliable for users.

SA models have to be built to gain the ability to predict the future evolution of a situation and detect potentially unmanageable incidents. Predicting potential changes from previous experiences and the capability of detecting new situations are essential features as they allow autonomous systems to anticipate their states in such environmental changes (Ravanbakhsh et al., 2021). These capabilities improve the effectiveness of the decision-making modules by providing suggestions and predictions, and allows the detection of possible deviations from the learned situations.

Self-aware systems must be designed in relation to the functionalities of each subsystem hierarchically. All layers in the hierarchy must be considered in an integrated way, in order to build a coherent self-aware vehicle that does not cause conflicting decisions and undesired effects. The framework should insure the self-aware system to have knowledge of itself and its experiences, allowing reasoning and intelligent decision making to support effective, and adaptive autonomous behavior (Mitchell, 2005). Self-awareness can also be seen as a collec-

tive behavior experienced by emergent awareness between interacting agents governed by simple behavioral and local interaction models. From a system complexity view, a self-aware system should be able to reconfigure (adapt) its subsystems through incremental processes. These systems should be able to tackle the challenges of managing increasingly complex interaction computations (Kephart and Chess, 2003). To ease computational complexity, autonomous agents have to configure themselves automatically according to some policies, to achieve different objectives.

This chapter presents the proposed self-awareness framework for autonomous vehicles. As we introduced in chapter 4, it is fundamental to build a framework that enables an agent to reach self-awareness capabilities. The framework is based on general AI (LeCun, 2022) by Lecun, Cognitive Dynamic Systems (Haykin, 2012) by Simon Haykin, and Seth's general controlled hallucination theory (Seth, 2021).

7.1 Introduction

Autonomous systems rely on sensors that provide data about the environment and internal situations to their perception systems for learning and inference mechanisms. These systems can also learn Self-Aware and Situation Aware generative modules from these data to localize themselves and interact with the environment. To be self-aware, an agent should be capable of jointly handling knowledge related to sensing, perceiving, interpreting, planning, and acting in an efficient framework.

Interaction modeling is an essential component of autonomous driving. Uncertainty in the interaction models, like observations and hidden state variables and their relations, makes it necessary to choose an appropriate representation. As discussed in section 2.4, uncertainty in generative models probabilistic models are coherent natural basis to deal with such dynamic situations. Probabilistic interaction models (Schulz et al., 2018; Nozari et al., 2022; Alemaw et al., 2022) can be used to model cross inter-dependencies of the interacting sub-systems and predict deviations from expected dynamic behaviors. Such models handle variables that can represent the environment, and incorporate prior beliefs about its state as well as the model of the agent's state. Dynamic Bayesian Networks (DBNs) (Cappelle et al., 2008; Han et al., 2009) are Probabilistic Graphical Models (PGMs) that have been proved to embed the features described above. They are efficient generative tools that represent uncertain causal relationships between variables, allowing the use of inference mechanisms for the estimation and self-organizing incremental learning of new models within homeostatic processes (Ravanbakhsh et al., 2021). In general, these capabilities are the core characteristics of self-awareness models.

DBNs are well suited for jointly modeling, in a coherent way, discrete and continuous variables coming from multi-sensorial data and representing model components at different abstraction levels hierarchically. Hierarchical DBNs (H-DBNs) (Regazzoni et al., 2020) can robustly represent fusion models, considering heterogeneous time series multi-sensorial signals within artificial agents. The fusion process can be represented as a generative model, by integrating perception and action tasks using hidden states related to the agent and the external world (environment) (Novianto and Williams, 2009).

Perceiving, representing, and acting in a dynamic environment results from interconnected and incremental processes, which can be learned using a probabilistic inference methodology. These learned models aim to estimate the environment's general state jointly with the agent's self-state, as a self-aware agent. Therefore, self-awareness is based on representing the ego agent and possible interacting agents that share the same environment. Self-awareness maps sensory observations in a mental space (here defined as the World Model), where dynamic behaviors and intentions of objects are represented at different resolution levels (Chen et al., 2014). Being able to autonomously navigate an environment is an example of a process that can benefit from self-awareness. This type of navigation has been carried out by following heuristic and optimal approaches (Liu et al., 2021).

In the heuristic approaches, autonomy is reached by applying empirical rules governing the model behavior. Such rules form a model that can achieve an immediate goal by driving the agent to focus on parts of the sensed environment relevant to the task. Heuristic methods are task-specific and mostly face substantial limitations, especially in complex environments. Maze-solving vehicles (Kathe et al., 2015) and robotic vacuum cleaner agents (Kim, 2004) are examples using this approach. Now, let us consider a cognitive dynamic context where the agent uses self-awareness to model the world state and perception modalities (e.g., attention focusing, perception, and filtering) concurrently. In this case, a global optimization problem has to be defined.

Optimal coupling between sensed data and World Model variables has to be driven by the optimization of an objective function that depends on the future states of the vehicle, including those that can be determined by planning and action taking. In this approach, the agent updates a model of the environment at each observation step (starting from an initial one), and it figures out an optimal path to reach the destination using heuristic rules. In this way, despite coupling between attention focusing on sensory data, the World Model states can be optimized. Often, the selected World Model contains heuristic rules, in order to determine agent intentions, that do not allow the system to couple dynamics in attention focusing with agent action policies in an explainable way. Such methods used in autonomous navigation, such as in ORB-SLAM (Mur-Artal et al., 2015; Fu et al., 2022), give good results using the

Bag of Words method (Sivic and Zisserman, 2003) for loop closure detection and relative pose estimation. Also, global bundle adjustment is used to improve the accuracy further. Moreover, feature-based (Kaneko et al., 2018; Engel et al., 2018) and stereo-matching-based (Mo et al., 2022) methods are state-of-the-art methods performing well, but they do not consider ego action and environment modeling coherently (together), as map creation is separate from action. These different SLAM methods simultaneously build localization and global mapping for an agent. Here, instead, we are developing a latent state local trajectory, stored in the MMP model, as an ordered hallucinated (inside-out (Seth, 2021)) collection of high-dimensional stimuli. These are coming from generalized state Hierarchical Dynamic Bayesian Networks (H-DBNs) (Slavic et al., 2023) that an agent will learn internally, from low-dimensional World Models, as a local latent representation and inference of a map trajectory.

In general, autonomous navigation is a challenging problem domain, as environments can change and not be perfectly known (Janai et al., 2021). In order to create a plan, the agent needs to build a continuously updated model of the regularly changing environment. The more uncertainty in the environment and the environment model, the harder it is to create stable maps, i.e., map models that allow one to localize oneself at each fraction of a second. Hence, an autonomous vehicle must reach its destination efficiently, while following local traffic laws and avoiding other cars driven by different types of drivers in rainy, snowy, and other challenging conditions.

This chapter proposes to build an interaction model that endures self-state and interact with other agents in the surrounding environment. Perception and inference frameworks are learned from data, by introducing a new cognitive self-awareness architecture. The perception model is learned from higher-dimensional data, through a self-supervised approach that uses a World Model obtained from low-dimensional sensors. In this way, higher dimensional data are explored in a controlled hallucination modality, where control is experienced by unsupervised low-dimensional models. We follow the optimization approach, through the optimization of an objective function designed to have no heuristic rules for the World Model stage. In other words, the agent learns World Models by observing a series of latent space data produced by lower-dimensional sensors, and perception models from higher-dimensional sensors guided by the World Models themselves. Such models, in turn, can be improved incrementally when the agent interacts with new environmental setups based on anomaly signals.

As introduced in section 4.3, a general cognitive architecture called Multi-Agent Self-Awareness Architecture (MASAA), for autonomous systems (fig. 4.1), is proposed. The architecture has the following characteristics: adaptability to self-awareness learning frame-

works, scalability to represent multi-ego agents, reliability in representing and doing inference of heterogeneous agents, conceptual integrity in representing an overall vision of reasoning, and maintainability of its components. The Active First Person Model(AFPM) and the instructions associated with the action model will be discussed in chapter 8. In this chapter, we focused on developing the Multi-Modal Perception (MMP) and the World Model (WM) models, communicating through the Short-Term Memory Module. At the inference stage, the Cost Module measures the deviation of learned models from the current environmental setups (observations), indicating whether to adapt or use the model as it is. WM and MMP can jointly represent the dynamic evolution of knowledge sets where an agent can project its states and interact with the neighboring agents. These models are designed and implemented to achieve autonomous self localization and overtaking interaction tasks. In particular, we consider self-aware overtaking interaction tasks of two interacting agents by building a hierarchy of knowledge from multi-sensorial signals in a Hierarchical Dynamic Bayesian Networks (H-DBNs) framework.

The major contributions in this chapter are: *i)* an overall novel cognitive architecture with all modules learning hierarchy of representations for hierarchical prediction and resolutions under uncertainty; *ii)* Bayesian generative and predictive models, evaluating the effectiveness of MMP and WM, for a generalizable interaction modeling demonstrated through overtaking maneuver. Overtaking is selected as a particular type of maneuvering to show the performance of MASAA framework; *iii)* best dynamic coupling between experience modeling from higher dimensional sensor information, and attention focusing from lower dimensional sensor information, to minimize the prediction error while doing joint tasks (self-localization and interaction).

7.2 Component Analysis of the system model

Our work is motivated by general AI (LeCun, 2022) by Lecun and Seth's general controlled hallucination theory (Seth, 2021). Controlled hallucination is a perception theory explaining how things in the world appear in perceptual experience. According to this theory, these experiences are the construction of the brain, rather than sensory signals streaming into the brain. The brain continuously makes predictions about sensory signals, and verifies these predictions against the sensory signals that goes through the sense organs. Perceptual predictions flow predominantly in a top-down direction, meaning from a brain to the actual world (inside-to-outside) direction. Prediction errors flow in a bottom-up direction, meaning from the real world to the brain (outside-to-inside) direction. These prediction error signals

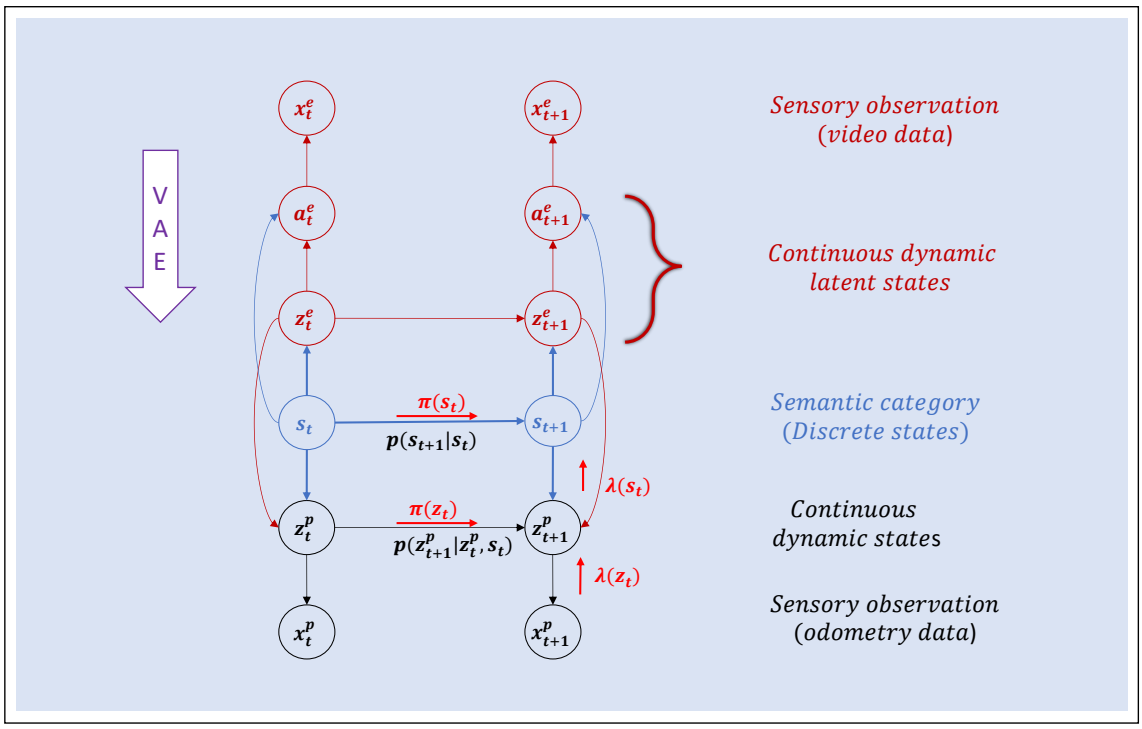


Figure 7.1: Single-agent H-DBN. A self-localization generative model that connects higher and lower dimensional sensory input variables, to produce a combined self-knowledge. © 2025 IEEE.

are used by the brain to update its predictions, which are then ready for the next round of sensory inputs. These process inside the brain makes oneself a self-aware entity.

Self-awareness is the representation of oneself and the other interacting objects within a shared space, i.e., being capable of mapping sensory observations in a mental space (that from now on we define as the World model). The dynamic behaviors and intentions of interacting objects can be represented at different resolution levels in the agent's hierarchical model (Chen et al., 2014). we propose a cognitive architecture based on perception and World Models to enable an agent to be aware(self aware). The architecture in fig. 4.1 shows the components of the novel self-awareness architecture. It shows the components os self-awareness profiling, filtering, anomaly detection, and incremental learning characteristics. Such models are often learned through world models connected with a heterogeneous multisensorial fusion process (Regazzoni et al., 2020; Liu et al., 2021). Each module in MASAA is the profile of an agent. The WM processes lower dimensional data to create lower-dimensional vocabulary through filtering and clustering of the filtered information. The same process can be applied on higher-dimensional data in the MMP model that can be

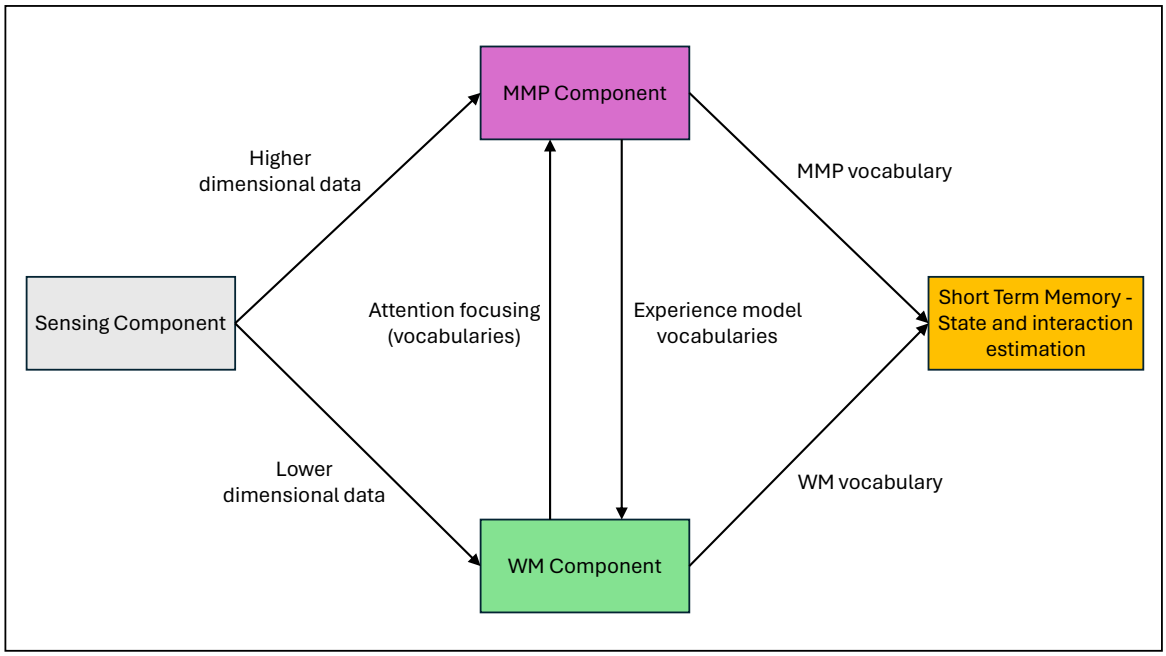


Figure 7.2: Simplified version of the self-awareness (MASAA) architecture that is currently implemented. The cost module, which is not included in this simplified MASAA version, is implicitly included in the short-term memory. The sensing component transmits odometry and video data to the WM and MMP components, respectively. First, the WM learns vocabulary representing the odometry data. Second, this vocabulary is used to guide (focus) the learning of video data according to the odometry module. Last, WM and MMP communicate through the short-term memory to apply the learned knowledge online (testing phase). © 2025 IEEE.

guided by the WM model. This integrated knowledge is used for localization and interaction in the Short-Term Memory module. This module is a hierarchical DBN that uses a more complex filter called MJPF, that integrates non-linear filtering and anomaly detection in both WM and MMP module. Model incrementality is already attended through this interaction between WM, MMP and Short-Term Memory modules. The continual learning profile is addressed in the Online and Incremental learning module of MASAA, detailed in Chapter 8. A simplified version of the architecture, which only includes the the sensing, WM, MMP, and Short Term Memory components, are displayed in fig. 7.2 for more readability.

In this chapter, our system model uses and implements the three components of the architecture. The remaining modules will be addressed in chapter 8. The first one is the Sensing component which gathers exteroceptive and proprioceptive sensory data. These collected data will be processed and transferred to the second and third component models (WM and MMP). The WM processes the data and produces vocabularies related to the lower

dimensional data. These WM-produced vocabularies will be used in the MMP component to guide the training of the higher-dimensional data using a cluster-guided KVAE.

After training is completed, the agent uses the combined vocabularies of the WM and MMP models as experience models. By leveraging this knowledge, it interacts with the environment through the MAC-MJPF algorithm. The system can be decomposed into Input / Output format:

- Sensing (physical sensors)
  - Input :
    - External environment
    - Internal environment
  - Output:
    - Extroceptive data
    - Proprioceptive data
- Multi-Modal Perception Model (MMP)
  - Input :
    - Extroceptive data
    - Proprioceptive data
    - Lower dimensional data vocabularies
  - Output :
    - Extroceptive Information
    - Proprioceptive Information
    - Higher dimensional data vocabularies
- World Model (WM)
  - Input :
    - Extroceptive data
    - Proprioceptive data
    - Extroceptive Information
    - Proprioceptive Information
    - Higher dimensional data vocabularies
  - Output :

* Lower dimensional data vocabularies
* Extroceptive Predicted state of interacting agent
* Proprioceptive Predicted state of self

In the localization and interaction process, the agent needs to know its internal state, specifically the state of the lower dimensional cognitive knowledge about its generalized state (position and its first-order derivative). These generalized states of the agent, learned as a pose estimation, are clustered together to partition the state space into different regions. This results a general lower dimensional map. This will guide the MMP model in learning the higher dimensional data according to the learned WM vocabulary.

7.3 Learning the World Model

Learning starts from the WM. In this model, we have lower-dimensional data that should be trained to focus the attention of the MMP model towards learning the best feature representations of the higher-dimensional data.

As a first stage, the WM-MA (MA = Multi-Agent) and WM-SA (SA = Single Agent) proprioceptive learning is applied. WM-MA takes sensory observations of the odometry module that contain the positional information of the agents along axis x and y i.e., $z^o_{t,u_i} = [\mathrm{x}_{t,u_i}, \mathrm{y}_{t,u_i}]$. The subscript t refers to the time step, whereas u_i discriminates the *i-th* (u_1 : agent 1 and u_2 : agent 2) agent involved in the interaction. A Null Force Filter (NFF) (Iqbal et al., 2021) is employed to obtain the first order Generalized States (GSs) $\tilde{z}^o_{t,u_i}$, which comprise position and velocity of each agent, i.e., $\tilde{z}^o_{t,u_i} = [\mathrm{x}_{t,u_i}, \mathrm{y}_{t,u_i}, \mathrm{v}_{\mathrm{x}_{t,u_i}}, \mathrm{v}_{\mathrm{y}_{t,u_i}}]$. We further process the GSs to have a semantic category (cluster) depicting the movement modality of both agents using the modified Growing Neural Gas (M-GNG) algorithm (Iqbal et al., 2021). An eight-dimensional feature vector $\mathscr{F}^o_{t,u_1}$ w.r.t. u_1, is provided as input for the clustering algorithm, and can be written as:

$$\mathscr{F}^o_{t,u_1} = [\mathrm{x}_{t,u_1}, \mathrm{y}_{t,u_1}, \mathrm{v}_{\mathrm{x}_{t,u_1}}, \mathrm{v}_{\mathrm{y}_{t,u_1}}, \triangle\mathrm{x}_{t,u_{12}}, \triangle\mathrm{y}_{t,u_{12}}, \triangle\mathrm{v}_{\mathrm{x}_{t,u_{12}}}, \triangle\mathrm{v}_{\mathrm{y}_{t,u_{12}}}], \tag{7.1}$$

where $\{\triangle\mathrm{x}_{t,u_{12}}, \triangle\mathrm{y}_{t,u_{12}}\}$ and $\{\triangle\mathrm{v}_{\mathrm{x}_{t,u_{12}}}, \triangle\mathrm{v}_{\mathrm{y}_{t,u_{12}}}\}$ are the relative positions and velocities of the interacting vehicle u_1 w.r.t. the neighboring vehicle u_2, which are computed as follows;

$$\begin{aligned}
\triangle\mathrm{x}_{t,u_{12}} &= \mathrm{x}_{t,u_1} - \mathrm{x}_{t,u_2}, \\
\triangle\mathrm{y}_{t,u_{12}} &= \mathrm{y}_{t,u_1} - \mathrm{y}_{t,u_2}, \\
\triangle\mathrm{v}_{\mathrm{x}_{t,u_{12}}} &= \mathrm{v}_{\mathrm{x}_{t,u_1}} - \mathrm{v}_{\mathrm{x}_{t,u_2}}, \\
\triangle\mathrm{v}_{\mathrm{y}_{t,u_{12}}} &= \mathrm{v}_{\mathrm{y}_{t,u_1}} - \mathrm{v}_{\mathrm{y}_{t,u_2}}.
\end{aligned} \tag{7.2}$$

On the other hand, in the WM-SA, the sensory observations of the odometry module are composed of the positional information of the agent along axis x and y, as $z_t^o = [\mathrm{x}_t, \mathrm{y}_t]$. Here, since we wanted to model the proprioceptive information, we only need to learn the internal dynamics of the overtaking agent. Hence, as in the WM-MA, we also apply NFF to obtain the second order Generalized States (GSs) $\tilde{z}_t^o$, which comprises position, velocity, and acceleration. We also include other motion parameters, such as the magnitude of the velocity, and the angular velocity, to have better semantics of the WM. The GNG algorithm is then applied on these eight dimensional vectors:

$$\tilde{z}_t^o = [\mathrm{x}_t, \mathrm{y}_t, \mathrm{v}_{\mathrm{x}_t}, \mathrm{v}_{\mathrm{y}_t}, \mathrm{v}_{\mathrm{norm}_t}, \mathrm{a}_{\mathrm{x}_t}, \mathrm{a}_{\mathrm{y}_t}, \omega_t], \tag{7.3}$$

where $\mathrm{v}_{\mathrm{x}_t}$ and $\mathrm{v}_{\mathrm{y}_t}$ are first order time derivatives w.r.t. position x and position y, respectively; $\mathrm{v}_{\mathrm{norm}_t}$ is the magnitude of the velocity; $\mathrm{a}_{\mathrm{x}_t}$ and $\mathrm{a}_{\mathrm{y}_t}$ are second order time derivatives w.r.t. position x and position y, respectively; ω_t is the angular velocity.

Once we learn the MMP model, the modeled agent has a prior belief of the environment and its state. In the World Model, the overtaking agent should be able to localize itself, and predict the state of the neighboring agent from the observed video frame sequences. Both exteroceptive and proprioceptive information are transferred from the MMP model as experience modeling. The transport (inter-model communication) between the SM and FPM is a learned process that handles knowledge transfer between the situation and first-person models. To clarify further, an agent can learn from the interaction of two external agents without involving itself as a third person, called Situation Model knowledge. It can also learn from its actions as a First Person model when it involves itself in the interaction. The transfer of this knowledge can be learned as a transportation process. Through the imitation learning process, the knowledge gained from situation models, can be utilized to solve related tasks when used in the first-person models. The more related the tasks, the easier it is to transfer or cross-utilize the acquired knowledge. In this case, we want to transfer the learned knowledge of different but interrelated aspects of the environment, modeled as first-person and situation models to share the knowledge between them.

7.4 Learning the Multi-Modal Perception Model

In this section, we discuss how the Multi-Modal Perception (MMP) module, represents the environment we model inside the artificial agent, through the integrated sensors to better understand the system. This module combines self-awareness and situational sensory fusion modality representations. Self-awareness allows an agent to monitor and predict its

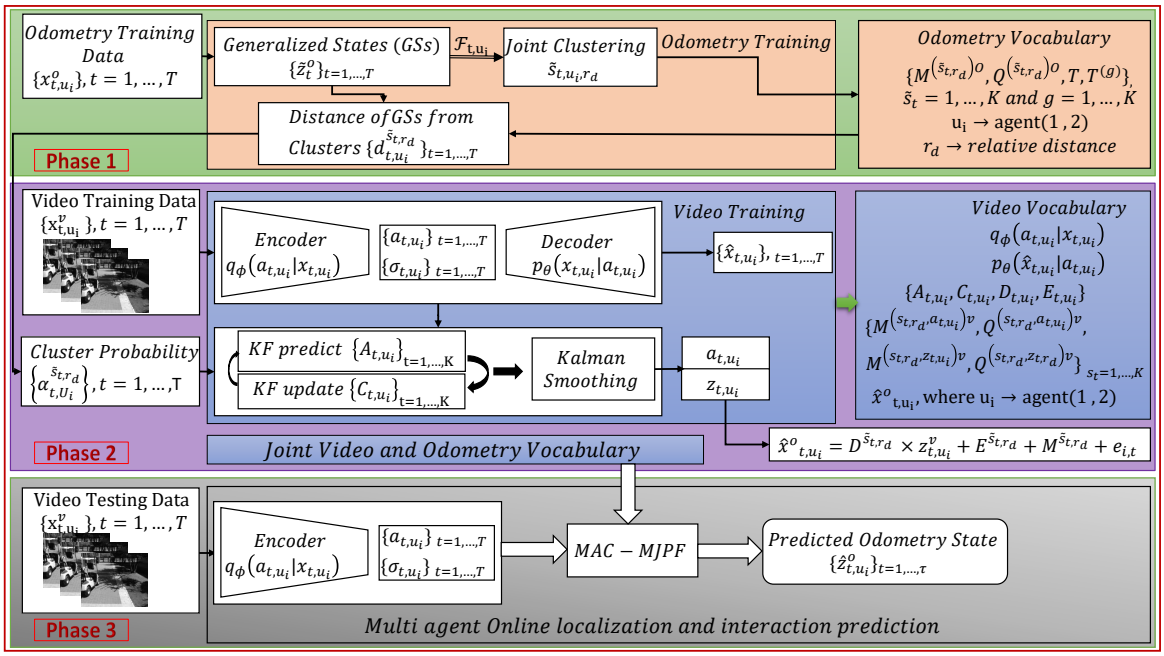


Figure 7.3: MASAA: Self-representation learning and inference frameworks. A learned vocabulary of proprioceptive information (controlled hallucination) drives exteroceptive information learning. © 2025 IEEE.

state at each step; simultaneously, the situational modality observes the environment for smooth interaction with other agents in the shared environment. In fig. (7.1) single-agent and (7.4) multi-agent perception modules are represented in multi-sensory generalized H-DBN frameworks. The SAH-DBN determines the self-state evolution corresponding to the environment (self-awareness) and, at the same time, the MAH-DBN regulates the contextual understanding of the environment using the self-state evolution where the agent operates.

We have two cluster-based dynamic Variational Autoencoders named MMP-MA and MMP-SA dynamic-VAEs. Both networks are utilized to reduce the video data's dimensionality, and to learn a representation of the latent state for later reconstruction and inference. MMP-MA learns the interaction model between the two agents, giving us exteroceptive information in the World Model. MMP-SA generates the proprioceptive information of the ego agent in the World Model. This type of architecture makes our model a hybrid between end-to-end and modular learning approaches.

After learning from the odometry modality, we use the learned vocabulary to train the video models. For this purpose, a Kalman Variational Autoencoder (KVAE) (Fraccaro et al., 2017) is employed to obtain the latent states $a_t^{(u_i)v}$ corresponding to the video of the training data $x_t^{(u_i)v}$. To elaborate more, we have two hidden states, i.e., $a_t^{(u_i)v}$ and $z_t^{(u_i)v}$, which

represent the content of an image and its corresponding dynamics, respectively. In the VAE encoding phase, we encode $\{A_k\}_{k=1,\dots,K}$ and $\{C_k\}_{k=1,\dots,K}$ matrices that are a set of K globally learned basic transition and pseudo-observation models, respectively. While training, since our main objective is to localize both agents only from the video sequence at each time instant, we learned two additional pseudo observation models, i.e., D and E, from the latent distribution and the odometry vocabulary. These matrices encode the information of the location parameters of interacting agents; therefore, we used four matrices (A, C, D, and E) representing our complex model.

The VAE encoder, for both MMP-SA and MMP-MA settings, is a three-layer convolutional neural network with 32, 64, 128 unites in each layer, kernel size of 3x3, stride of 2, and LeakyReLU activation function. The decoder is an equally sized network for deconvolution. As an optimizer, we use ADAM with an initial learning rate of 0.001, and an exponential decay scheme with a rate of 0.97 every 20 epochs.

For improved reconstruction and easier inference of objectives, for the single agent we set the minimum motion related latent state $z_t^v \in \mathbb{R}^4$ and content related latent state $a_t^v \in \mathbb{R}^{10}$. In the multi-agent setting, $z_t^{(u_i)v} \in \mathbb{R}^8$ and $a_t^{(u_i)v}$ is the same dimension as the single agent $a_t^{(u_i)v} \in \mathbb{R}^{10}$. Motion related latent states are matched with the parameters of the odometry generalized state vectors $z_t^o \in \mathbb{R}^4$ for the single agent, and $z_t^{(u_i)o} \in \mathbb{R}^8$ for multi-agent. The pseudo-observation matrix dimensions are $D_t \in \mathbb{R}^{N\times M\times L}$ and $E_t \in \mathbb{R}^{N\times L}$, where N is number of clusters, M is the dimension of the cluster mean vector, and L is the dimension of the motion related latent state vector of the video data.

The trained odometry vocabulary $\tilde{s}_k^{(u_i)o}$ is employed to guide the model to assign a semantic category to the latent states. The two latent states are modeled as follows:

$$a_{t,u_i} = \sum_{k=1}^{K} \alpha_{t,u_i}^{\tilde{s}_k} \times C_{k,u_i} \times z_{t,u_i} + v_{t,u_i}, \tag{7.4}$$

where a_{t,u_i} is the content distribution that is obtained from the dynamical latent state z_{t,u_i} through a pseudo-observation model. On the other hand, z_{t,u_i} at consequent time instants are connected through the following dynamical model:

$$z_{t+1,u_i} = \sum_{k=1}^{K} \alpha_{t,u_i}^{\tilde{s}_k} \times A_{k,u_i} \times z_{t,u_i} + \omega_{t,u_i} \tag{7.5}$$

In eq. (7.4) and in eq. (7.5), v_t and ω_t are Gaussian noises; matrices $\{C_{k,u_i}\}_{k=1\dots K}$ and $\{A_{k,u_i}\}_{k=1\dots K}$ represent a set of pseudo-observation models and transition models, respectively. These models are combined through a probabilistic vector $\alpha_{t,u_i}^{\tilde{s}_k}$ as a *guiding vector*, which guides the model to assign a semantic category (cluster) to the dynamic video state

z_{t,u_i}. Mathematically, $\alpha_{t,u_i}^{\tilde{s}_k}$ can be defined as follows:

$$1/\alpha_{t,u_i}^{\tilde{s}_k} = \left(d_{t,u_i}^{\tilde{s}_k}\right)^n \times \sum_{k=1}^{K} \frac{1}{\left(d_{t,u_i}^{\tilde{s}_k}\right)^n}, \tag{7.6}$$

where $d_{t,u_i}^{\tilde{s}_k} = \mathscr{D}_M((z_{t,u_i}^v),(M^{(\tilde{s}_k)o},Q^{(\tilde{s}_k)o}))$ is the Mahalanobis distance ($\mathscr{D}_M$) (Xiang et al., 2008) calculated between each latent state distribution z_{t,u_i}^v and the probability distribution of each cluster of the odometry module $(M^{(\tilde{s}_k)o},Q^{(\tilde{s}_k)o})$. $M^{(\tilde{s}_k)o}$ is the mean of cluster $\tilde{s}_k$, and $Q^{(\tilde{s}_k)o}$ is its covariance, i.e., the probability distribution is a Gaussian. In the first iteration, z_{t,u_i}^v will be assigned to the superstates corresponding to the shortest distance $d_{t,u_i}^{\tilde{s}_k}$ values. After the first iteration, $d_{t,u_i}^{\tilde{s}_k}$ will be multiplied by the previous $\alpha_{t,u_i}^{\tilde{s}_k}$ value to cluster the z_{t,u_i}^v, having similar probability distributions. n is a temperature value used to assign, to image sequences, the highest probability of the WM cluster.

In (Fraccaro et al., 2017), after performing Kalman smoothing, four main losses are used to optimize the model:

(a) the traditional reconstruction loss (between $x_t^{(u_i)v}$ and its VAE reconstruction $\hat{x}_t^{(u_i)v}$).

(b) the KullbackLeibler Divergence term of the VAE, which represent the VAE loss $\mathscr{L}_{VAE}$;

(c) a transition loss $\mathscr{L}_{trans}$ enforcing the learning of the transition models A and B.

(d) an emission loss $\mathscr{L}_{emiss}$ enforcing the learning of the emission models C.

In our implementation, as transition loss, we used the Bhattacharya distance DB between the smoothed latent state at time $t+1$, i.e., zs_{t+1}, and the state predicted from zs_t using the matrices A:

$$\mathscr{L}_{trans} = D_B(zs_{t+1}, Azs_t) \tag{7.7}$$

For the emission loss, we used DB between the latent state a_t and its dynamics predicted from zs_t using the matrices C:

$$\mathscr{L}_{emiss} = D_B(a_t, Czs_t) \tag{7.8}$$

While training, we also learned the *zero-th order observation matrices* $\hat{z}_{t,u_1}^o, \hat{z}_{t,u_2}^o$ as follows:

$$\hat{z}_{t,u_1}^o = D^{\tilde{s}_k} \times z_{t,u_1}^o + E^{\tilde{s}_k} + M^{\tilde{s}_k^O} + e_{1_t}, \tag{7.9}$$

$$\hat{z}_{t,u_2}^o = z_{t,u_1}^o + r_{d_t,u_{12}} + e_{2_t}, \tag{7.10}$$

where the prediction of $\hat{z}^o_{t,u_1}$ and $\hat{z}^o_{t,u_2}$ are performed by minimizing the errors (e_{1_t}, e_{2_t}) w.r.t. the ground truth odometry. $r_{d_t,u_{12}}$ is a relative distance vector, which can be mathematically defined as:

$$r_{d_t,u_{12}} = [\triangle x_{t,u_{12}}, \triangle y_{t,u_{12}}, \triangle v_{x_{t,u_{12}}}, \triangle v_{y_{t,u_{12}}}]. \tag{7.11}$$

Optimization is performed through cross modality losses using *Huber Loss functions* (Hastie et al., 2009).

During the (online) testing phase, the learned vocabularies s^v_{k,u_1}, $\tilde{s}^o_{k,u_1}$ and $\hat{z}^o_{t,u_1}$, $\hat{z}^o_{t,u_2}$ are employed to predict the trajectories of the interacting agents.

One thing to note in the MMP module is that the mathematical notations are presented by assuming multi-agent interactions. When a single agent is trained, the mathematical models will be updated. The pseudo-observation model becomes:

$$a_t = \sum_{k=1}^{K} \alpha_t^{\tilde{s}_k} \times C_k \times z_t + v_t$$

The dynamical model connects z_t values at consequent times:

$$z_{t+1} = \sum_{k=1}^{K} \alpha_t^{\tilde{s}_k} \times A_k \times z_t + \omega_t$$

As a result, we have two sets of vocabularies from both odometry and video modalities. Odometry vocabularies include, cluster mean $(M^{(\tilde{s}_k)o})$, cluster covariance $(Q^{(\tilde{s}_k)o})$, transition probability (T) and time spent in each cluster $(T^{(g)})$ as WM knowledge. Video vocabularies having the same characteristics as the odometry, have cluster mean $(M^{(s_k,a_t)v}, M^{(s_k,z_t)v})$ and cluster covariance $(Q^{(s_k,a_t)v}, Q^{(s_k,z_t)v})$ as MMP representation knowledge.

7.5 Online Modal Interaction

In the online interaction, the MMP and WM models should communicate robustly to switch the situation model to first-person model. The main task of these modules is to fuse models and sensors to determine the state of the ego-vehicle and the interacting agent. Since we do not have the exact state information (position and velocity) for both agents, except the learned models of the interacting agent, what is recommended in the literature is to use Interacting Multiple Model (IMM) filters (Genovese, 2001). So, we are fusing multiple models with many hypotheses and combining them based on their strength (low error) to get better overall estimations. Learning well-structured models should provide optimal results in estimating ego-state and interacting agent states. The agent may face a challenge in predicting other

neighboring agents, since their control input is unknown. When predicting motion, a system have to consider the following main issues:

1. The dynamics and kinematics of the system;
2. The commands and known inputs;
3. The unknown and random inputs.

Hence, a model must control to a certain level these situations, when estimating ego-motion and neighboring agent states. The integrated filter has to access the dynamics as a form of a mathematical model to be able to estimate both states. If we are considering ego-motion, the filter will have access to the control inputs directly. Regarding the unknowns affecting the model accuracy, the filter should account for the process noise. If this noise is very high, the certainty of the prediction will drop.

Regarding the interacting agent, since we are modeling uncooperative agents, we do not have the access to the control inputs directly, so the filter should consider it as unknown in prediction. Introducing more unknowns to our models makes the prediction less confident, leading us to increase the process noise. This implies that we have to trust more the correction measurement. Since sensors are very noisy, if the process noise is very high, the difference between the actual situation and the estimation will be very high. Therefore, integrated multiple (multi-hypothesis) filter models should be applied simultaneously, with different prediction and process noise models. By fusing the output of each filter based on the model likelihood or averaging them, we can have a better estimate.

In our model, the communication mechanism is mainly in the short-term memory, comprised of Multi-Agent Combined Markov Jump Particle Filters (MAC-MJPFs). We chose these filters because they are integrated and well-suited to adjust particle estimations, by fast re-initialization, when the agent changes its motion. As a single integrated multi-filter, it allows updating each filter after each measurement to get an updated state and state covariance based on the fusion of the most likely models based on their weights. Our algorithm MAC-MJPFs is composed of interrelated MJPFs which have to handle:

- Video latent state prediction;
- Generalized proprioceptive state prediction;
- Interacting agent generalized state prediction.

The *Multi-Agent Coupled Markov Jump Particle Filter* (MAC-MJPF) is proposed for the prediction of the trajectories $(\tilde{z}^{o}_{t,u_1}, \tilde{z}^{o}_{t,u_2})$ at the online phase. MAC-MJPF employs KF and PF

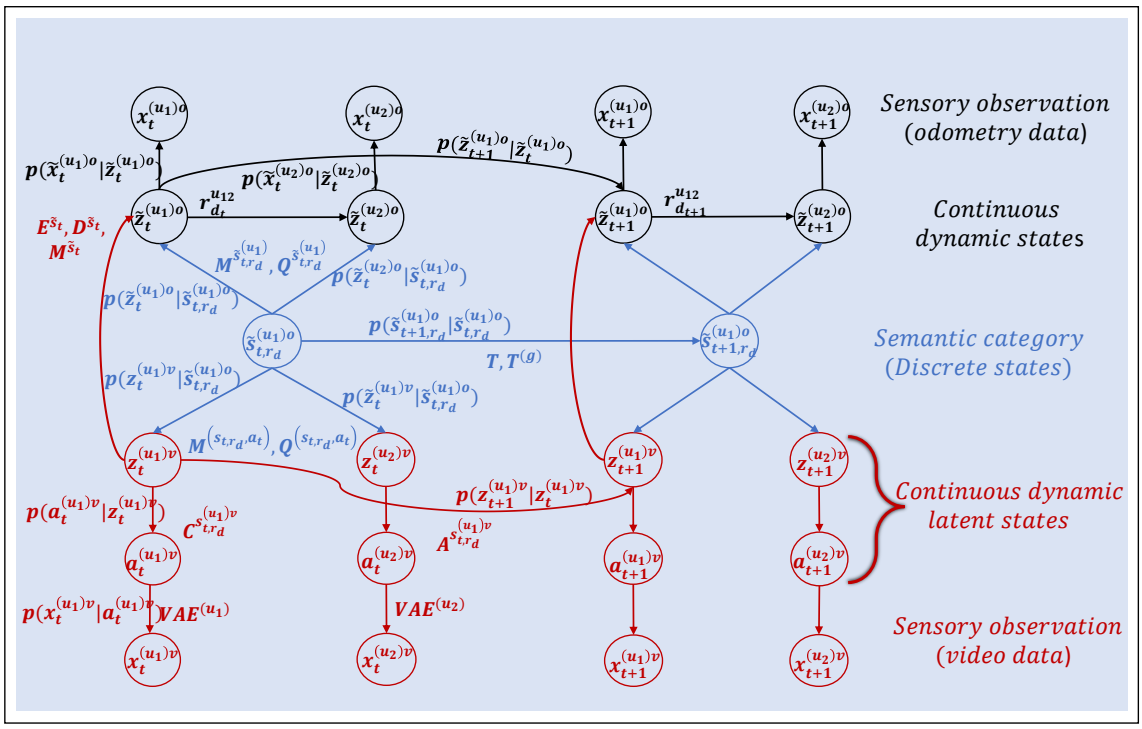


Figure 7.4: Multi-Agent H-DBN. It illustrates the interaction which models the inter-relation between the higher and lower dimensional vocabularies, and also represents multi-agent interaction knowledge through a relative distance vector for the overtaking interaction. © 2025 IEEE.

to make inferences about the future states over the proposed H-DBN model. The MAC-MJPF algorithm is detailed in Alg. 2, and represented with a flow chart in fig. 7.5. It integrates WM-SA and MMP-SA (see fig. 7.1) when the configuration is related to a single agent. When the configuration is related to a multi-agent, it integrates WM-MA and MMP-MA (see fig. 7.4). In these figures the black color indicates proprioceptive information, and the red color is used to indicate the exteroceptive information. The interaction between them is represented in a light-blue color.

7.6 Employed Datasets

Two semi-autonomous real vehicles, called *iCab*1 and *iCab*2 (Marín-Plaza et al., 2016), are employed to perform the overtaking experiments. Since our objective is the localization of the interacting agents, we considered an overtaking maneuver in which *iCab*2 overtakes *iCab*1. *iCab*1 and *iCab*2 are u_1 and u_2 in the previous notations. In this dataset training includes

Algorithm 2: Localization and Interaction algorithm.

Input: $M^{(\tilde{s}_k)^o}, Q^{(\tilde{s}_k)^o}, T, T^{(g)}, M^{(s_k,a_t)^v}, M^{(s_k,z_t)^v}, N_{th_1}, N_{th_2}, Q^{(s_k,a_t)^v}, Q^{(s_k,z_t)^v}, x_t^{(u_1)^v}, Int_{agent}, startRandom$
for $t = 1,\ldots,\tau \leftarrow$ *Time evolution* **do**
  Obtain image latent state using VAE: $a_t^{(u_1)^v}$, $\sigma_t^{(u_1)^v}$ and obtain latent state covariance $\leftarrow \Sigma_t^{(u_1)^v} \sim I_L \times \sigma_t^{(u_1)^v}$
  where L is the number of latent state features
  Compute distances of video encodings from video clusters:
  $d_t^{s_k^{(u_1)^v}} = \mathscr{D}_B((a_t^{(u_1)^v}, \Sigma_t^{(u_1)^v}), (M^{(s_k,a_t)^v}, Q^{(s_k,a_t)^v}))$, where $s = 1,...,K$
  Calculate probabilistic *guiding vector* $\alpha_t^{(s_k^{u_1})}$ using eq. (7.6)
  Begin Filtering
  for $n = 1,\ldots,N \leftarrow$ *number of Particles* **do**
    if $t == 1$ **then**
      $W_n = \frac{1}{N} \leftarrow$ weight of the particles
      Initialize the video and odometry vocabularies:
      $s_t^{(u_1)^v} \sim p(z_t^{(u_1)^v} | s_t^{(u_1)^v})$
      $\tilde{s}_t^{(u_1)^o} \sim p(\tilde{z}_t^{(u_1)^o} | \tilde{s}_t^{(u_1)^o})$
    else
      UPDATE:
      Update video states:
      $z_{t|t,n}^{(u_1)^v}, \Sigma_{t|t,n}^{(u_1)^v} = KF_{update}^{Video}(z_{t|t-1,n}^{(u_1)^v}, \Sigma_{t|t-1,n}^{(u_1)^v}, C_{t,n}^{s_t^{u_1}}, a_{t,n}^{(u_1)^v}, \Sigma_{t,n}^{(u_1)^v})$
      Update $\tilde{z}_{t|t,n}^{(u_1)^o}$ (itself) according to eq. (7.9)
      if $Int_{agent} == True$ **then**
        Update $\tilde{z}_{t|t,n}^{(u_2)^o}$ according to eq. (7.10)
      end if
      if $t == 1$ *and* $startRandom == True$ **then**
        Re-sample if $\sum_n \frac{1}{W_n^2} < N_{th_1}$
      else
        Re-sample if $\sum_n \frac{1}{W_n^2} < N_{th_2}$
      end if
    end if
    PREDICTION:
    Predict vocabulary of video:
    $s_{t+1,n}^{(u_1)^v} \sim T(\tilde{s}_{t,n}^o)$,
    Predict vocabulary of odometry:
    $\tilde{s}_{t+1,n}^{(u_1)^o} \sim T(\tilde{s}_{t,n}^o)$,
    Predict Video latent state:
    $z_{t+1|t,n}^{(u_1)^v}, \Sigma_{t+1|t,n}^{(u_1)^v} = KF_{pred}^{Video}(z_{t|t,n}^{(u_1)^v}, \Sigma_{t|t,n}^{(u_1)^v})$
    Predict location of u_1 (itself) and $r_{d_t}^{u_{12}}$ wrt interacting agent u_2 :
    $\tilde{z}_{t+1|t,n}^{(u_1)^o}, \Sigma_{t+1|t,n}^{(u_1)^o}, r_{d_t}^{u_{12}} = KF_{pred}^{Odometry}(z_{t|t,n}^{(u_1)^v}, \Sigma_{t|t,n}^{(u_1)^v})$
    Smoothing: $\tilde{z}_{t|t-1}^{(u_1)^o} = \mathbb{E}[\tilde{z}_{t|t-1}^{(u_1)^o}]_n$
    if $Int_{agent} == True$ **then**
      Predict location of neighboring agent u_2 using $r_{d_t}^{u_{12}}$:
      $\tilde{z}_{t+1|t,n}^{(u_2)^o}, \Sigma_{t+1|t,n}^{(u_2)^o} = \tilde{z}_{t+1|t,n}^{(u_1)^o}, \Sigma_{t+1|t,n}^{(u_1)^o}) + r_{d_t}^{u_{12}}$
      Smoothing: $\tilde{z}_{t|t-1}^{(u_2)^o} = \mathbb{E}[\tilde{z}_{t|t-1}^{(u_2)^o}]_n$
    end if
  end for
  Output: Predicted trajectories $\tilde{z}_{t|t-1}^{(u_1)^o}$ (oneself) and
  if $Int_{agent} == True$ **then**
    Update Output:
    Predicted trajectories of interacting agent: $\tilde{z}_{t|t-1}^{(u_2)^o}$
  end if
end for

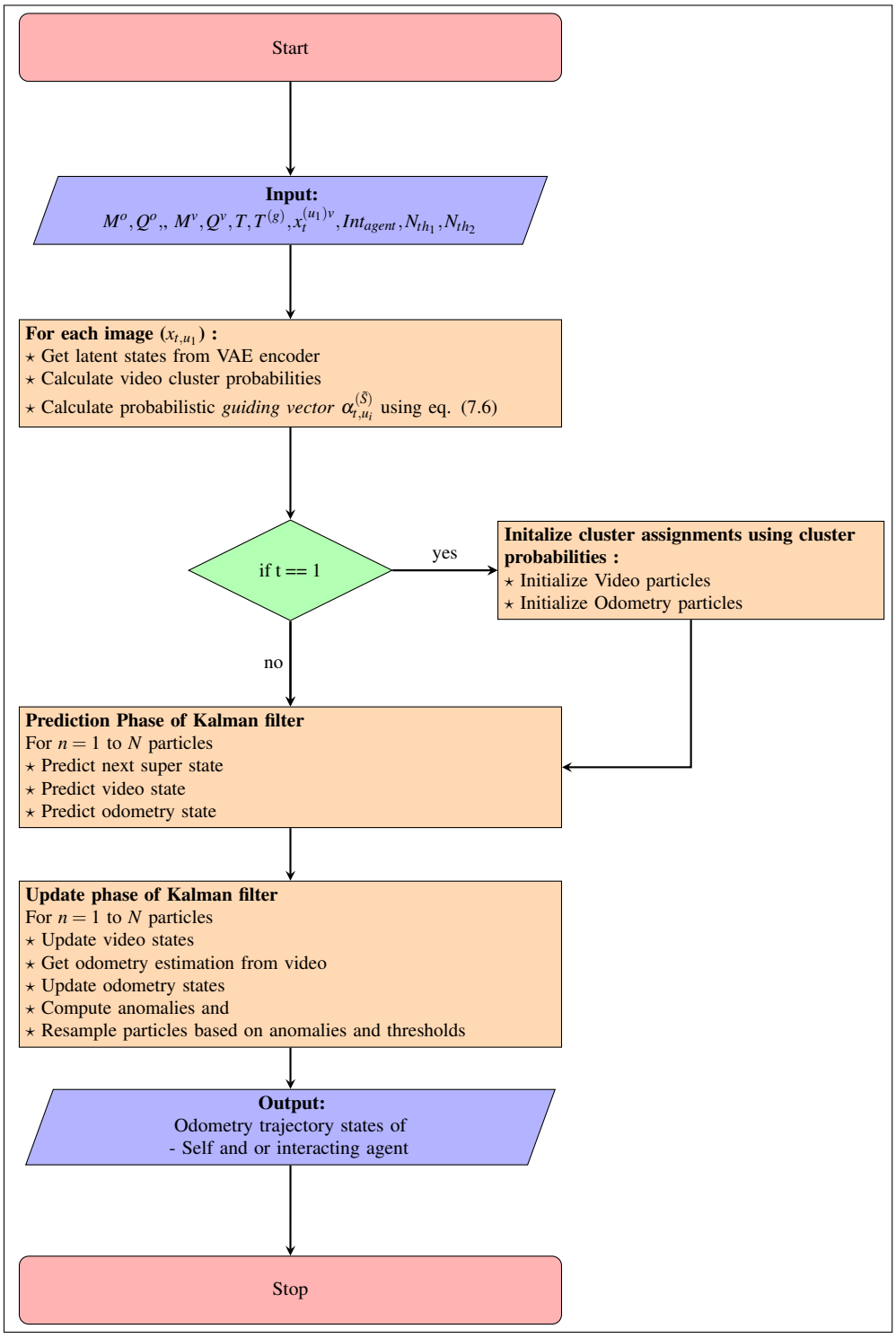


Figure 7.5: Flow chart model of multi-modal MJPF: a detailed description, including stages of the filtering process, are described in algorithm 2. © 2025 IEEE.

three overtaking experiments (Exp:01, Exp:02, and Exp:03); the validation experiment is Exp:04, while Exp:05 is the testing experiment (can be seen in fig. 5.3).

In addition to the above dataset, we tested our model on data coming from the well-known CARLA simulator (Dosovitskiy et al., 2017), built to support the development, training, and validation of autonomous driving systems. We considered an overtaking scenario to make it coherent with our proposed method. Both datasets are pre-processed to synchronize video and odometry data, and also synchronize the agents. We performed eight experiments using the CARLA simulator. Training includes four overtaking experiments (Exp:01, Exp:02, Exp:03, and Exp:04) performed in normal daylight conditions. Validation comprises two experiments (Exp:05 and Exp:07). Exp:05 is performed under normal daytime environmental conditions, as in the training cases. Exp:07 is performed at sunset time. The remaining experiments (Exp:06 and Exp:08), shown in figs. 7.13 and 7.14, are used for testing. Exp:06 is performed in light rain conditions in the evening (sunset), with a stop maneuver due to a red traffic light. Exp:08 is performed in heavy rain conditions while performing the overtaking task, which is different from the learned overtaking one.

In addition to these experiments, we considered following (an agent follows instead of overtaking) experiments. These experiments are performed to analyze the behaviours of different models (following and overtaking) tested on the same scenario: first, testing the overtaking scenario on the wrong training model, which is the following model, and then with the appropriate model, the overtaking model. These will give us the extent to which our model can generalize a given scenario.

To test the generalizability of our model, we also used a public dataset (KITTI (Geiger et al., 2012)). We specifically tested the single agent localization using these six sequences (sequence: 00, 01, 02, 03, 04, and 05), and compared our results with the state-of-the-art methods.

All these experiments are analyzed to better represent the self-state of an agent and the base interaction with the environment. In MASAA, however, other interaction types (e.g., an interaction with an unmanned aerial vehicle that can be analyzed to jointly predict trajectories while performing a particular task) with similar configurations can also be performed.

In figs. 7.6, 7.7, 7.8, and 7.9, the blue oval shape show, the overtaking interaction zone. In fig. 7.6,overtaking interaction is performed after both agents change lanes. The experiment in fig. 7.7, resembles the curved regions of the World Model. The testing experiment in fig. 7.8, the overtaking interaction is performed in a different region from the training. Stop experiment for a traffic red light is shown in fig. 7.9.

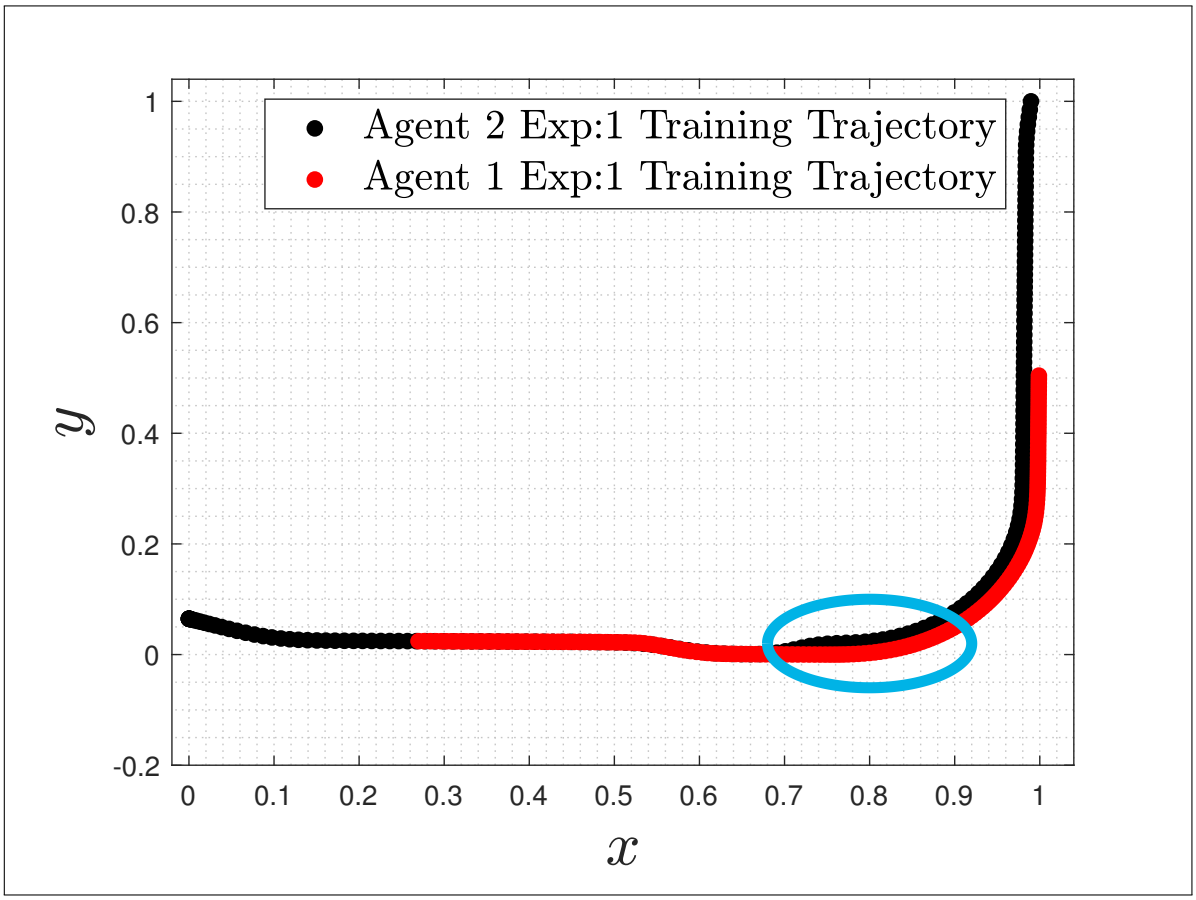


Figure 7.6: Training trajectory: overtaking interaction takes place in a starting point of a curved zone. The light-blue oval shape represents the overtaking area. © 2025 IEEE.

7.7 Learned H-DBN Models

The agent is able to learn H-DBNs, representing the learned vocabularies of self-localization and interaction, that will be used for Bayesian inference. Combined vocabularies, representing the WM and MMP, are learned for multi-agent and single-agent networks as shown in fig. 7.4 and 7.1, respectively.

The unsupervised learning based WM vocabularies are represented by clustered positional trajectories are shown in figs. 7.10 and 7.11 for iCab and CARLA agents respectively. Arrows indicate the direction of motion of vehicles encoded in $\tilde{s}_{k,u_i}$, and different colors represent the different clusters.

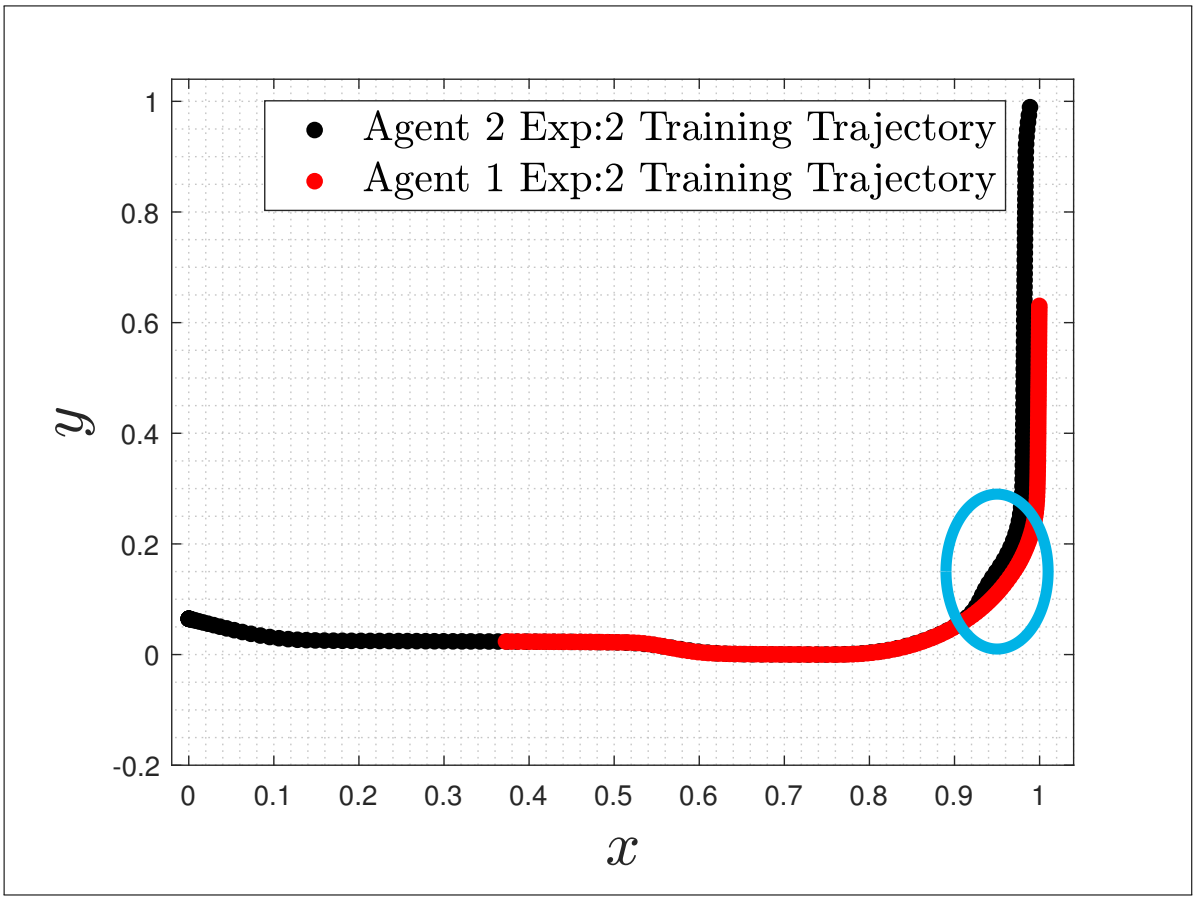


Figure 7.7: Training trajectory: overtaking interaction takes place in a curved area. The light-blue oval shape represents the overtaking area. © 2025 IEEE.

7.7.1 World Model H-DBN

WM network is a low-dimensional cluster network used to steer the MMP model towards the best features representing the WM vocabularies. It is a low-dimensional generative model representing how an agent, as a point in a 2D space, interacts with the environment when doing a particular task (e.g., overtaking). It is the base knowledge (fig. 7.3 phase 1) guiding the higher dimensional knowledge learning process (fig. 7.3 phase 2).

7.7.2 Multi-Agent CG-KVAE H-DBN

In the MMP, we have two distinct and related networks. The first network is the Multi-agent MMP which is responsible for learning interactions based on the relative state estimation. It is a generative model of higher dimensional data (e.g., video) guided by the WM model vocabularies. For example, in the case of overtaking, the MMP module learned the motion predictions coherent with changes in the video content. This multimodal coupling between

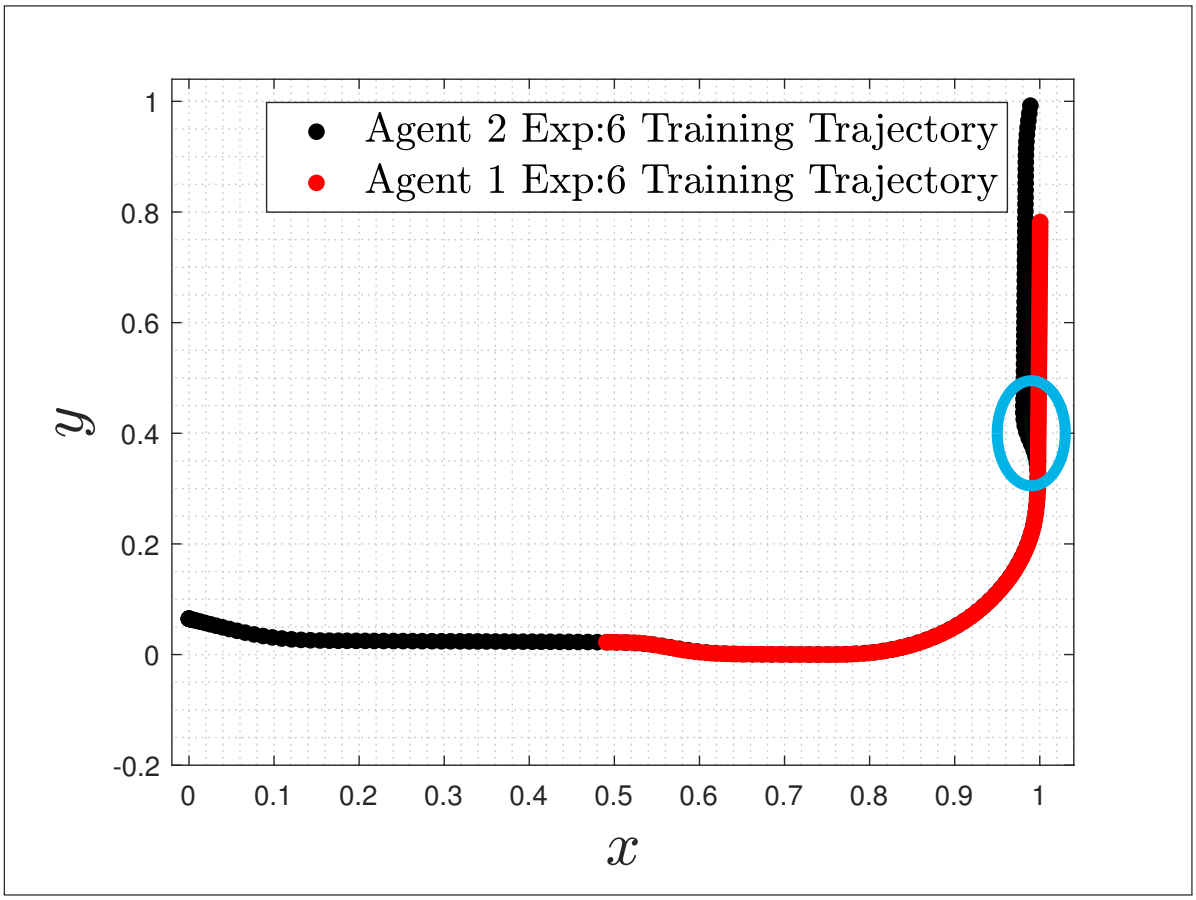


Figure 7.8: Testing trajectory: Overtaking is performed in straight area under the condition of heavy rain. The light-blue oval shape represents the overtaking area. © 2025 IEEE.

learned models represents a basic feature of a self-aware agent. It has two interrelated sub-networks, each one handling each interacting agent trained together. These sub-networks allow us to model the state of one agent in terms of the other agent's state.

7.7.3 Single-Agent CG-KVAE H-DBN

Instead of learning the relative state of an agent, here the model has learned to predict the ego-motion parameters, which are the generalized states from the higher dimensional sensory input. The cognitive knowledge of an agent's self-state will allow the agent to interact with the environment and execute any possible maneuvering tasks.

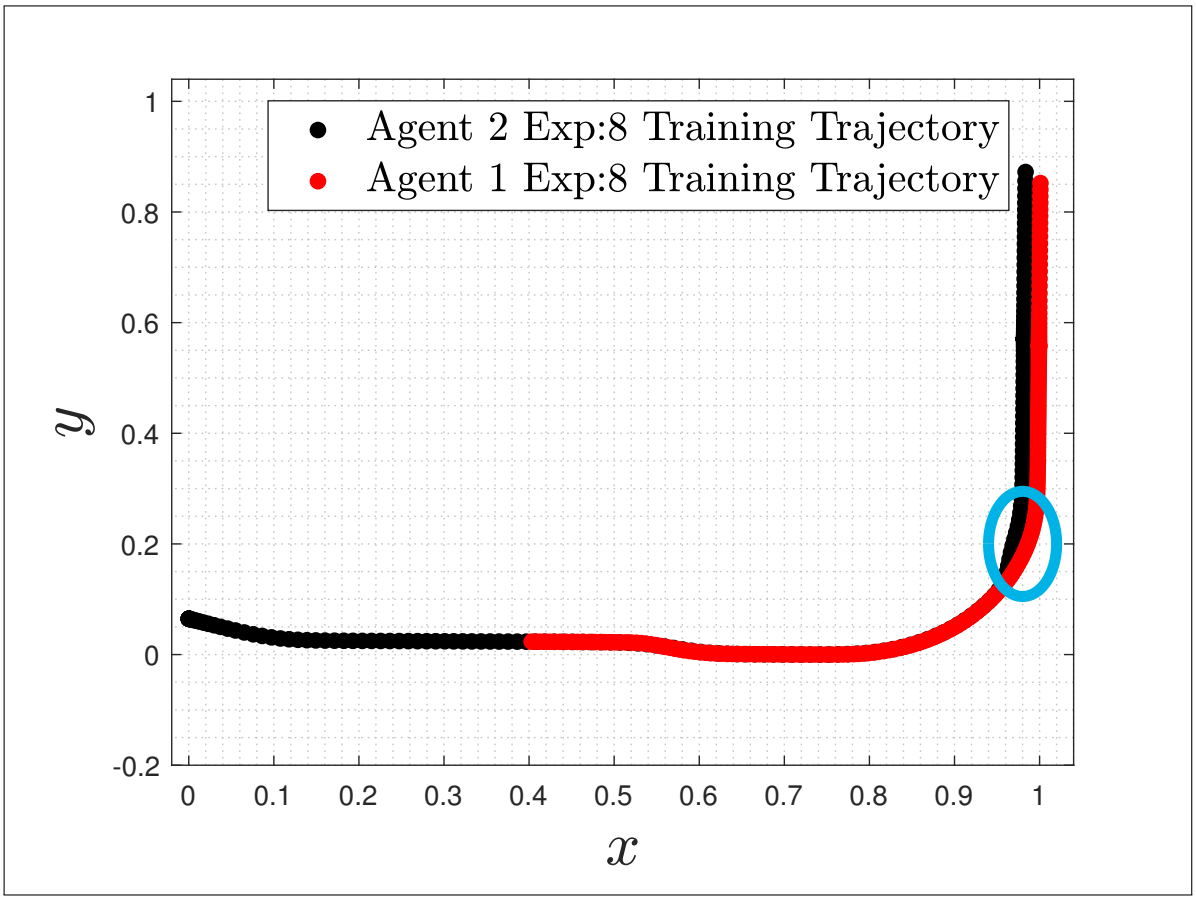


Figure 7.9: Testing trajectory. This scenario is performed to test traffic light anomaly and light rain in the evening time of the day. The light-blue oval shape represents the overtaking area. © 2025 IEEE.

7.8 Localization and Interaction Algorithm

The MAC-MJPF algorithm 2 has four sections. Section one is the input to the algorithm which takes the combined vocabulary of video and odometry and the input image frame. This is an online algorithm which performs the entire process of interaction while localization for each single image frame. As we discussed in 2, 3.4.3 MJPF is a systematic integration of Particle and Kalman filters to account for nonlinear state dynamics model.

The challenge in MJPF is in deciding the number of particles. The number of particles decides how many prediction and update messages have to be aggregated to have a balanced inference model between complexity and accuracy of the model. This challenge has a big impact specially in online learning and decision-making process. As we will discuss these challenges in 8, while tackling the online policy learning and action selection in the active inference framework (Friston et al., 2015), increasing the number of particles has to be

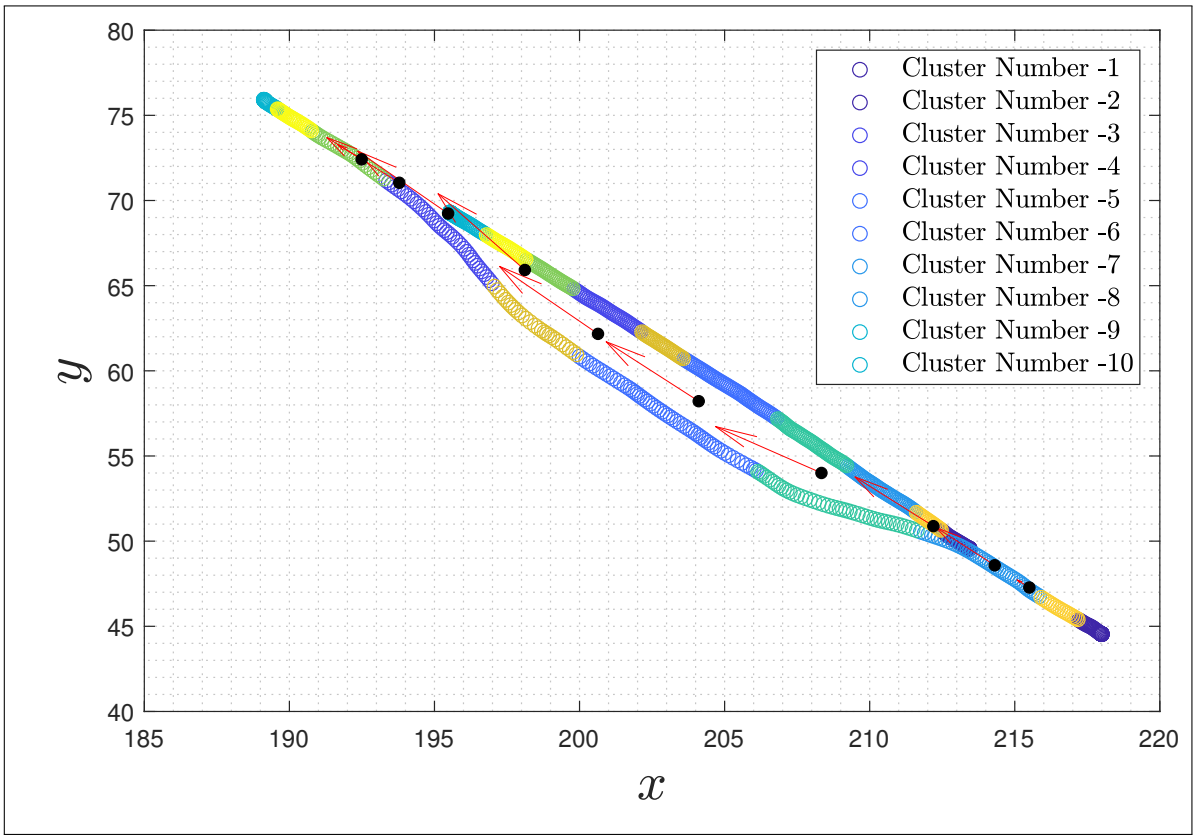


Figure 7.10: Clustered positional trajectories of iCab - WM. The red arrow represents mean velocity of each cluster. Mean position of each cluster is represented by the filled black colored points. The difference from fig. 6.3 of Chapter 6 is due to the fusion of second-order motion parameters in the clustering algorithm to improve the WM vocabulary. © 2025 IEEE.

bounded based on the computational resources. To minimize the computation time, for each image frame before going through the inference algorithm, there the second section which performs latent state generation process.

In section two the algorithm first feeds the image frame to the VAE encoder, which will generate the latent state (a_t) of the input x_t. a_t is the encoding of x_t, encoded to a low-dimension space. The VAE encoder also provides the mean and the variance associated with the image(recall that in Ch. 7, section 2.5.3, $q(z|x) = \mathcal{N}(\mu, \sigma)$). The latent state covariance, is then obtained through the identity function according to the size of the latent state (I_L). If for example L is 10, the identity will be 10 by 10 (10×10). The identity is scaled by σ_t, an element wise operation.

Using the latent state and the covariance, Alg. 2 computes the distance between each vocabulary of the video modality. The distance metric used here is the Bhattacharyya distance, but other distance measures that are distribution based can be used. Also if one wants to

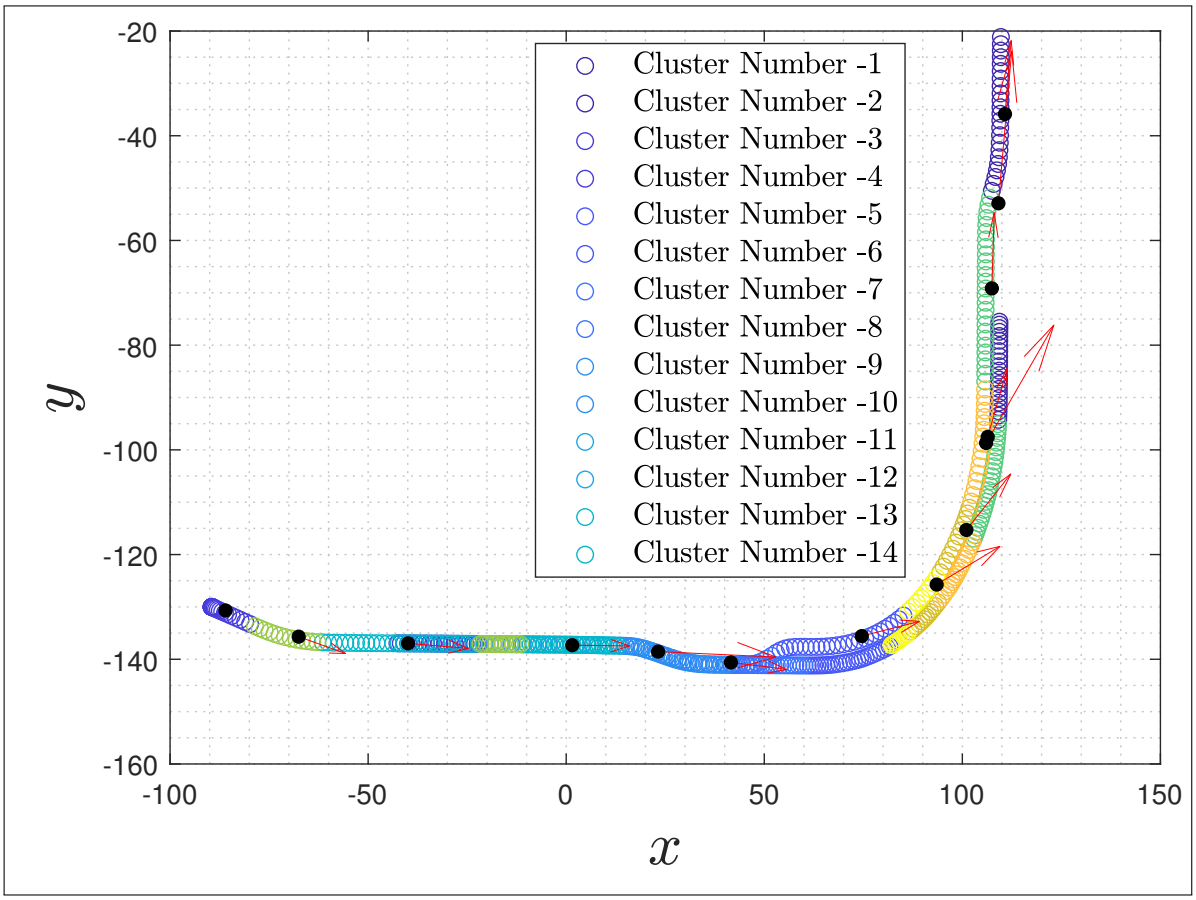


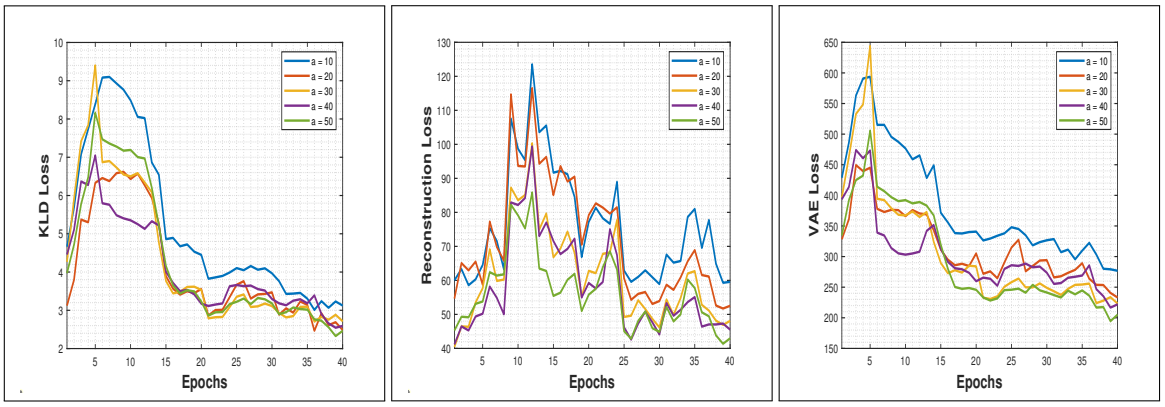
Figure 7.11: Clustered positional trajectories of CARLA - WM. Mean position and mean velocity are represented by the black doted points and red arrows respectively. As in fig. 7.10 second-order motion parameters are fused with the GSs in the clustering algorithm. © 2025 IEEE.

compute based on the mean of the latent state distances like Mahalanobis distance can be used. From this distance the probability vector is computed based on eq. (7.6), which will assign a probability based on the distance (greater distance means low probability to be in that cluster while the lowest distance indicates the agent is in that region (cluster)). This finalizes section two of the MAC-MJPF algorithm.

The third and forth part (section) of the algorithm is based on the number of particles. Particles are approximators of a dynamic system. Their propagation can be guided based on the learned vocabulary, since it represents the map of the environment. At the first time the measurement from each cluster (region) will have nearly same weight, which will be scattered allover the environment. But if the distance probability vector is used it will converge to the minimum distance region at least to a certain probability, approximating the dynamics very well. This measure will minimize the regions that are less probable, so the

Table I: Table displaying the temperature values according to eq. (7.6), and the corresponding RMSE error between the ground truth generalized states and the learned generalized states. The RMSE values shown here are the mean of the generalized state RMSE errors. From this, the best temperature value is between 10 and 25. Hence, $n = 15$ is used in our learning models.

Datasets	**RMSE for different n values**					
	$n = 5$	$n = 10$	$n = 15$	$n = 25$	$n = 50$	$n = 100$
iCab	0.0319	0.0257	0.0244	0.0274	0.0275	0.0294
Carla	0.0095	0.0093	0.0094	0.0098	0.0105	0.0110
Kitti	0.0019	0.0029	0.0036	0.0044	0.0046	0.0047



(a) KLD Losses for different latent state dimensions

(b) Reconstruction Losses for different latent state dimensions

(c) Total VAE Losses for different latent state dimensions

Figure 7.12: (a)KLD, (b)Reconstruction, and (c)Total VAE losses from the KITTI validation dataset of agent2, for different content related latent state (a) dimensions: $a \in \left(\mathbb{R}^{10}, \mathbb{R}^{20}, \mathbb{R}^{30}, \mathbb{R}^{40}, \mathbb{R}^{50}\right)$

agent measurements can be fused with the prediction model to take the best approximate state.

The prediction step is the third section of the algorithm. In this section, $t+1$ future states are estimated. The next step region for video and odometry($s^{(u_1)v}_{t+1,n}$ and $\tilde{s}^{(u_1)o}_{t+1,n}$) will be predicted respectively; Next motion related video latent states ($z^{(u_1)v}_{t+1|t,n}$ and $z^{(u_2)v}_{t+1|t,n}$) of agent 1 and agent 2 will be predicted respectively; Then the odometric localization estimation of interacting agents ($tildez^{(u_1)o}_{t+1|t,n}$ and $\tilde{z}^{(u_2)o}_{t+1|t,n}$) of u_1 (itself) and $r^{u12}_{d_t}$ based u_2 will be predicted. This is the common prediction step of the Kalman Filter, where for each particle n, there is a Kalman filtering process.

The forth step is the updation of each discrete and continuous states based on measurement (here the video signal). The video updation ($KF^{Video}_{update}(z^{(u_1)v}_{t|t-1,n}, \Sigma^{(u_1)v}_{t|t-1,n}, C^{s^{u_1}_t}_{t,n}, a^{(u_1)v}_{t,n}, \Sigma^{(u_1)v}_{t,n})$) for

Table II: Analysis of latent state size influence on the relation of odometry and video vocabularies.

Latent state dimension	a $\in \mathbb{R}^{10}$	a $\in \mathbb{R}^{20}$	a $\in \mathbb{R}^{30}$	a $\in \mathbb{R}^{40}$	a $\in \mathbb{R}^{50}$
iCab experiments	0.7054	0.7500	0.7679	0.7643	0.7679
KITTI experiments	0.6408	0.7483	0.7845	0.8094	0.8299
CARLA experiments	0.8422	0.8203	0.8063	0.8297	0.8547

each particle based on their updated cluster the C matrix can transform the latent $a_{t,n}^{(u_1)v}$ to $z_{t,n}^{(u_1)v}$, and using the video measurement covariance $\Sigma_{t,n}^{(u_1)v}$ it updates the motion related video latent state $z_{t|t,n}^{(u_1)v}, \Sigma_{t|t,n}^{(u_1)v}$.

The odometry update is not straightforward, because we are supposed to use only the camera sensor at the testing time. Hence, the agent does not have an odometric input, instead it uses the matrices (D and E) that are learned to change video observations to odometric observations. These matrices are approximations which in completely new environments, they might give non-optimal approximate results. The approximate odometry is $\tilde{z}_{t|t,n}^{(u_1)o}$ using eq.(7.9) and the interacting agent $\tilde{z}_{t|t,n}^{(u_2)o}$, will be approximated through (7.10).

The odometry updation process is not the exact update through measurements (sensory observations), rather it is an induced update through the video observation($D^{\tilde{s}_k}, E^{\tilde{s}_k}$) and the learned odometry vocabularies ($M^{(\tilde{s}_k)o}, Q^{(\tilde{s}_k)o}$).

As we already discussed it in Chapter 6, the performance of the interaction algorithm depends on the guiding vector. Selecting the salient features of video information and linking them to the WM vocabulary relies on the dimensions of the encoded video latent states. The results in table II exhibit, the dimension of the content related latent state has an influence when the KVAE network assigns the corresponding odometry cluster to each latent representations of the encoded video data. When the latent state dimension is $a \in \mathbb{R}^{10}$, the video vocabulary has low percentage of conformance to the odometry one. Hence to have best coupling one has to choose the dimensions cautiously. Based on the VAE validation losses in fig. 7.12, dimensions $a \in \mathbb{R}^{20}$ and $a \in \mathbb{R}^{30}$ are recommended choices. As we already presented the VAE losses for the iCab and CARLA datasets in Chapter 6, here, we only presented the KITTI dataset VAE losses for different dimensions of latent encodings.

7.9 Resampling and Anomaly Signals

The Cost module uses different anomaly signal measurement models, in different hierarchical levels, to explain the results of our model. Video content anomaly and latent state anomaly

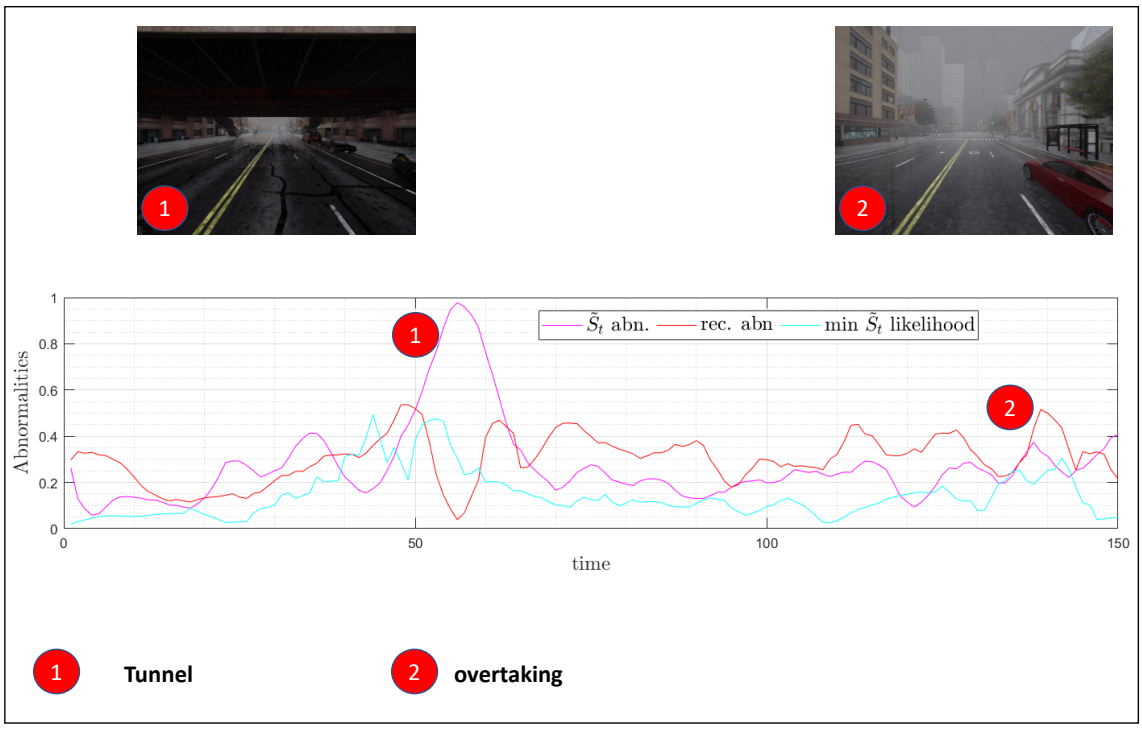


Figure 7.13: Anomaly signals: Exp:06. The highest anomaly signal is when the agent passes under a bridge. The light variability and limited number of such video frames in the training data results in higher video reconstruction anomaly. © 2025 IEEE.

are both indicators of the MMP module, while odometry anomaly is the indicator of the WM module. KLDA indicates the integrated WM and MMP anomaly signal.

7.9.1 Anomaly Signals

Anomaly signals used in this chapter are the following: **KLDA:** it is an energy-based model measuring anomalies (differences between probability distributions) from the predictive ($\pi(s_t)$) and diagnostic ($\lambda(s_{(t+1)})$) messages, when estimating superstates (clusters).

$$SuperStateLevelAnomaly = KLDA(\pi(\tilde{s}_t), \lambda(\tilde{s}_{t+1}))$$

Video content anomaly: it is estimated by taking the mean square error between the latent state prediction related to the content of each image from the models $(a_{t,t-1,n})$ and the encoded latent state of the testing image $(\hat{a}_{t|t-1,n})$.

$$a_{anomaly} = \frac{1}{N}\sum(a_{t|t-1,n} - \hat{a}_{t|t-1,n})$$

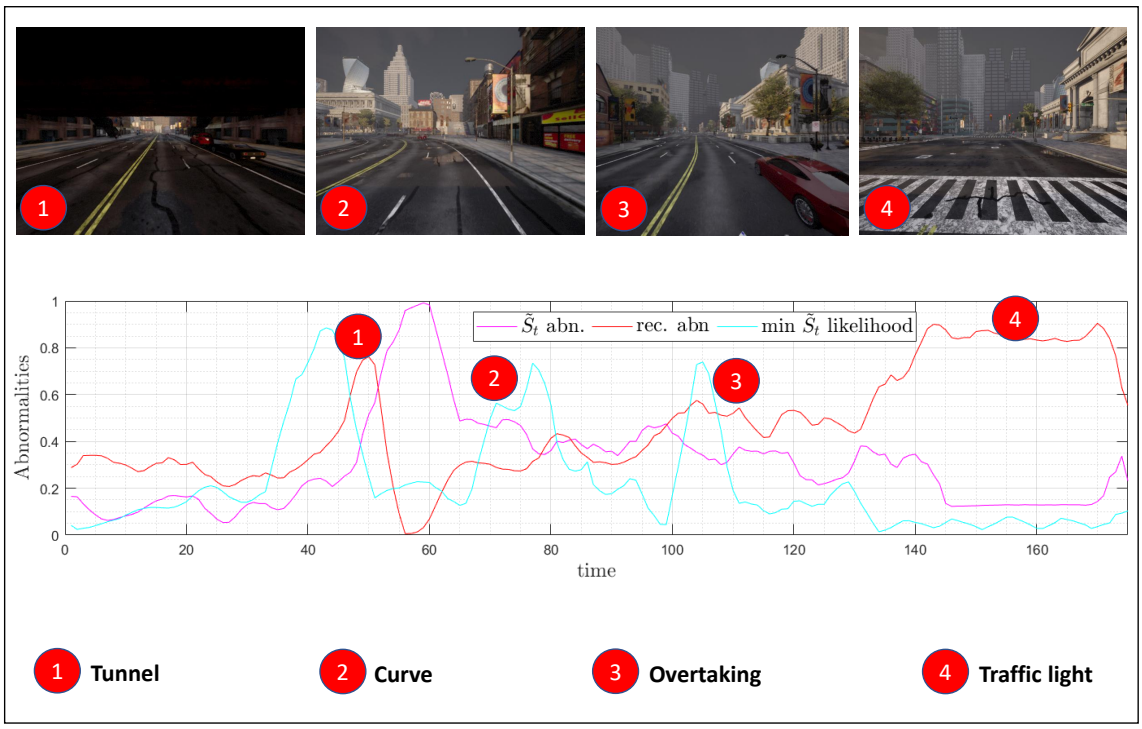


Figure 7.14: Anomaly signals: Exp:08. The light variability and the red traffic light introduction in the testing dataset results in a higher anomaly signal. © 2025 IEEE.

Latent state anomaly: mean square error between the predicted $(z_{t,t-1,n})$ and updated state of latent state $(\hat{z}_{t|t-1,n})$. It allows us to estimate the anomaly signal related to the motion parameter corresponding to each image in sequence.

$$z_{anomaly} = \frac{1}{N}\sum(z_{t|t-1,n} - \hat{z}_{t|t-1,n})$$

Image reconstruction anomaly: the mean square error between the real image (x_t) and the reconstructed image of VAE $(\hat{x}_t)$.

$$ImageReconstructionError = MSE(x_t, \hat{x}_t)$$

Odometry prediction anomaly: it is computed by taking the mean square error between the pseudo-observation odometry parameter $(z^o_{t,n})$ and its prediction parameter $(\tilde{z}^o_{t,n})$.

$$z^o_{anomaly} = \frac{1}{N}\sum(z^o_{t,n} - \tilde{z}^o_{t,n})^2$$

7.9.2 Particle resampling

The sampling process is a reweighting operation of particles with very low weights. We propose two-step thresholds for reweighting each particle. This process enhances the performance of particle filters and also solves the degeneracy problem of particle filtering. Reweighting is a process of resampling particles that have lower weight values. The resampling process will be enforced when the filter's effective sample size N_{eff} (see Section 3.4.2), is below a defined threshold (Arulampalam et al., 2002). This sample size can be determined using two thresholds. The first one, N_{th_1}, requires many samples. This process is performed at the starting step of the filtering process if the starting position of the vehicle is random. In this case, sampling will be in effect constantly until the tracking is stable. After stabilizing, it will be replaced by the second threshold, N_{th_2}, which requires a small number of effective samples compared to the first threshold. In this way, we can decrease the number of resampling operations.

When filtering begins, if the prediction is random, we resample most of the particles by assuming different starting points. If the agent always starts from the same starting position and that position is known, one threshold is sufficient since the first filtering step will not be done randomly.

7.10 Results

As a result of applying the trained models within the MASAA architecture, the derived features are used to predict the state of an agent and its neighboring agent. More specifically, these features are the outputs of the SA-MMP and MA-MMP models, which will be used to predict and then act online. The differences between these trained generative predictive models and the current situations are evaluated within the Cost module, by using anomaly signals obtained according to the free energy minimization principle.

Table I shows how the WM guides the MMP to learn the best features related to generalized states for different temperature values. RMSE values in this table are computed between the ground truth trajectory and the learned WM guided MMP learned trajectory. Table V displays the analysis of time, RMSE drift, and number of particle of MAC-MJPF; Tables VI and III show inference errors related to different datasets and models. Anomaly signals, indicating the performance of the learned models, are shown in figs. 7.13 and 7.14.

The sampling threshold parameter value, in table IV, is used to resample particles in algorithm 2. The sampling threshold is computed considering the number of particles, in order to limit the effective sample size. For example, if the number of particles is 100, the effective sample size will be 50 in single-agent state estimation and 28 in multi-agent state

Table III: Table displays the quantitative analysis of the proposed method based on Root Mean Square Error (RMSE) measurements.

KITTT experiments	**RMSE**			
	x	y	v_x	v_y
KITTI Sequence 00	1.6384	3.5576	0.0005	0.0011
KITTI Sequence 01	7.6647	5.8347	0.0001	0.0002
KITTI Sequence 02	2.4944	4.5604	0.0007	0.0006
KITTI Sequence 03	0.6865	0.8377	0.0002	0.0004
KITTI Sequence 04	0.0448	0.7097	0.0006	0.0003
KITTI Sequence 05	0.8615	0.7918	0.0007	0.0006

Table IV: Table displaying the testing parameters for MAC-MJPF. The temperature value in the testing scenario is according to eq. (7.6).

	Temprature Value	Sampling Threshold
Multi Agent	0.6	0.28
Single Agent	0.2	0.5

estimation. Resampling is activated based on the effective sample size. If the inverse sum of particle weights is less than the effective sample size, resampling will be executed.

The MMP sample learned trajectories are shown in figs. 7.15, 7.16, and 7.17 for iCab, CARLA and KITTI agents respectively. The trajectories in fig. 7.15, and fig. 7.16 are WM based learned trajectories from MMP of multi-agent scenarios. The single agent WM based learned trajectory from MMP is shown in fig. 7.17.

The results show that the proposed model has gained knowledge of self-localization and interaction, as shown in figs. 7.19(a) and 7.19(b) for 06 and 08 testing experiments respectively.

The proposed method implemented in the MASAA architecture, not only provides an explainable cognitive framework, but can achieve good estimation performance. To demonstrate this estimation capacity, under the MAC-MJPF algorithm, figs. 7.19(a) and 7.19(b) show particle approximations. The particle approximations shown in fig. 7.19(b) are more accurate than the ones reported in fig. 7.19(a), as both the training experiment and the testing interaction in fig. 7.19(b) were performed in the same area. More details on errors related to the state estimation of interacting agents as a Root Mean Square Error (RMSE) drift are presented in table VI.

The anomaly signals in experiment six (fig. 7.13) and eight (fig. 7.14) are color coded. The color codes for both experiments are: pink-colored is KLDA between the predictive

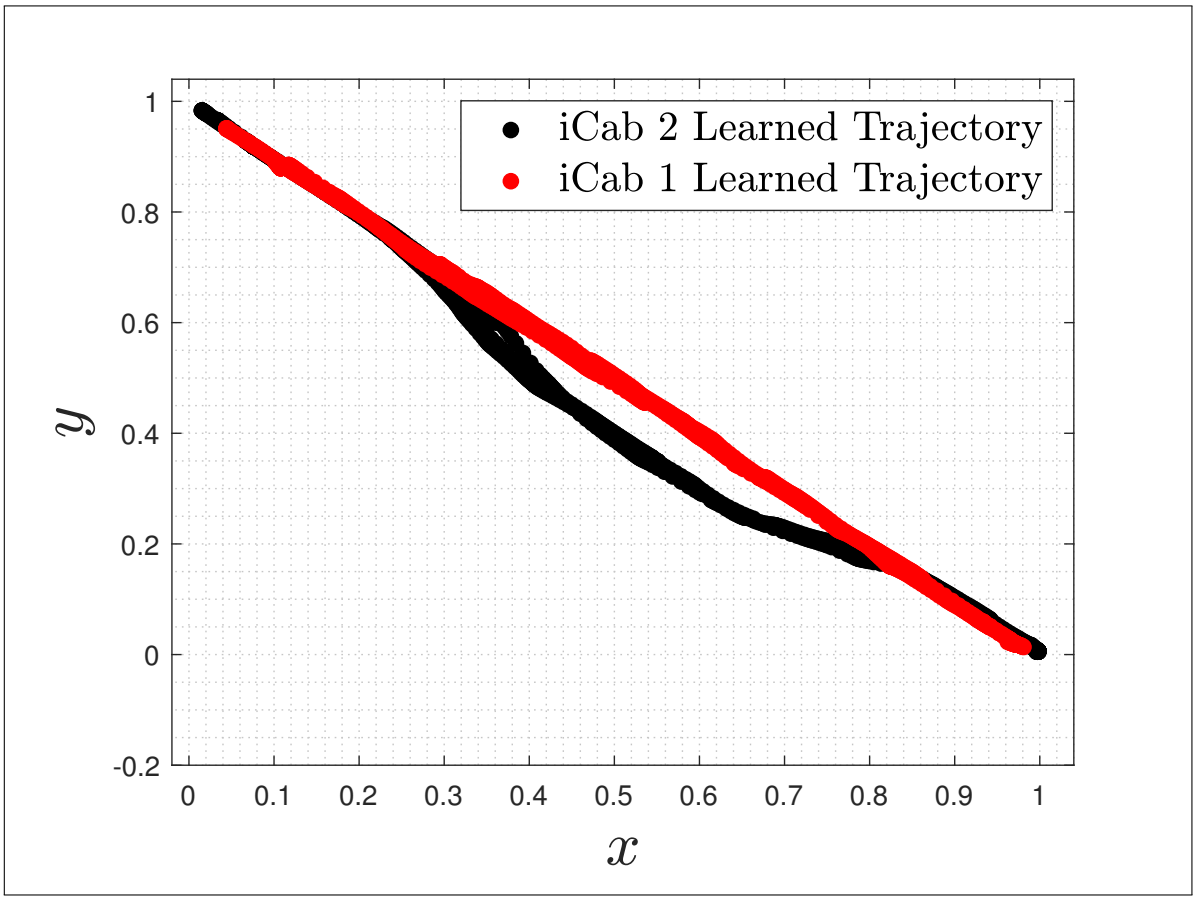


Figure 7.15: iCab trajectory. WM guided MMP learned trajectory from video sequences. © 2025 IEEE.

and diagnostic messages. The cyan-colored signal is the min likelihood of particles in each cluster. The red-colored signal is the frame reconstruction error.

The general free energy minimization concept applied at different representation levels allows the Cost module to compute and explain the behavior of anomaly signals. The anomaly signals depicted in figs. 7.13, and 7.14 exhibit an increase in amplitude when the model encounters an interaction area it has not seen before (the learned overtaking zone is different from the testing one). Particularly, in fig. 7.14, the peaks in the anomaly signal are highlighted; the labels in the red circles represent the cause of the peak. Zone 1 is a tunnel; zone 2 is a curved area; zone 3 is the overtaking area, and zone 4 is a traffic light area. In fig. 7.13, on the other hand, it shows the measured anomaly signals under heavy rain conditions. The video reconstruction anomaly signal corresponding to the end of the anomaly signal is lower in amplitude in experiment 06 (Exp:06) than experiment 08 (Exp:08). This happens because Exp:08 considers a red traffic light anomaly signal, which is not present in Exp:06. These multilevel anomaly signals can be used as a basic information source to

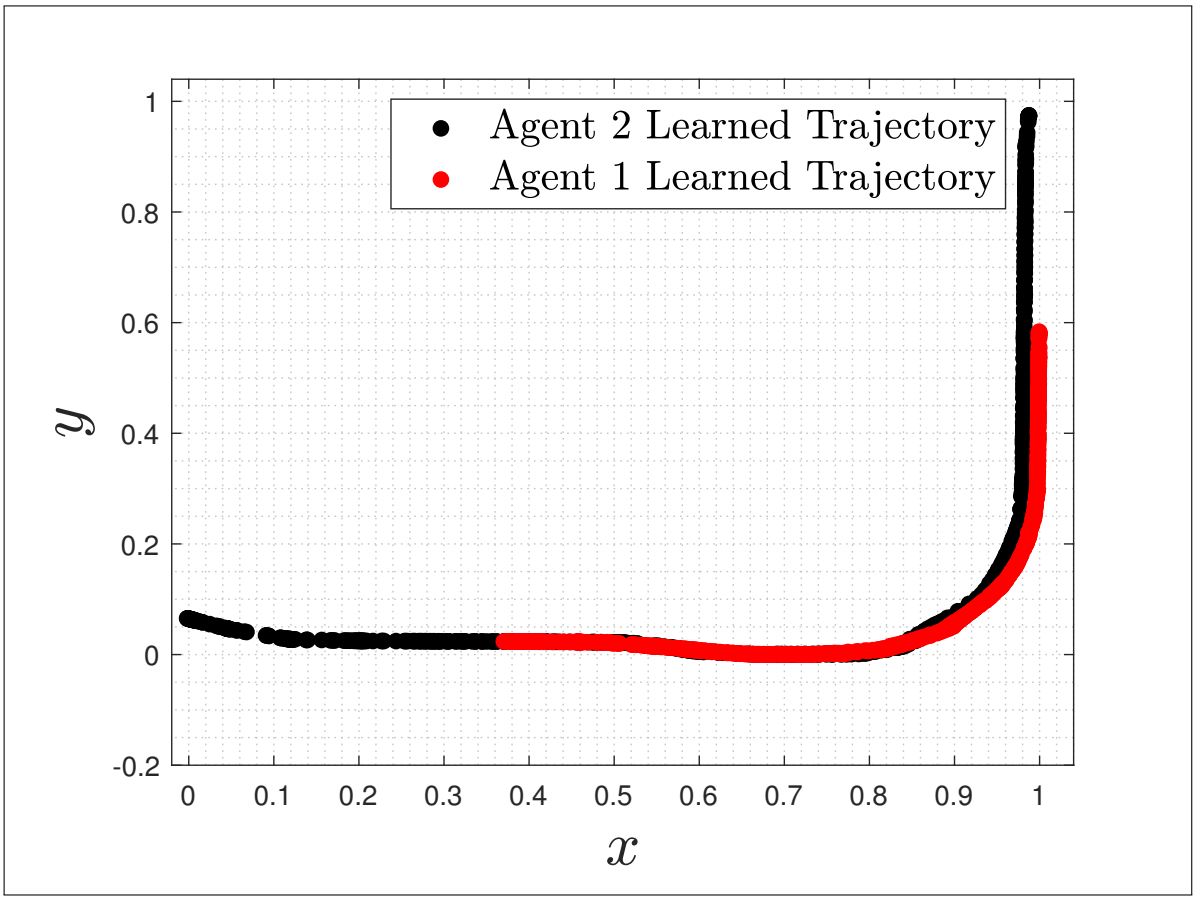


Figure 7.16: Carla trajectory. WM guided MMP learned trajectory from video sequences. © 2025 IEEE.

any reward-based (Li, 2021) or preferred observation-based (Nozari et al., 2022) learning frameworks that can be incorporated to learn from self-action in an incremental learning process.

The complexity analysis in table V is performed with short, medium, and long sequences of testing datasets to show the MAC-MJPF performance in different covered distances.

The estimation results demonstrate the capability of the MASAA architecture to learn and use a generative model representation for task-based systems on multimodal data (video and odometry in our case). SLAM methods are the reference class of approaches suitable for fair comparisons. Specifically, deep learning-based SLAM methods (Li et al., 2022; Lai et al., 2020) are being studied to generate high-level semantics and build an agent's knowledge base to enhance the agent's perception and intelligence. One of the goals of this chapter is to evaluate how well the integrated WM, MMP, and Cost modules can work in the MASAA architecture to learn from data in an explainable representation.

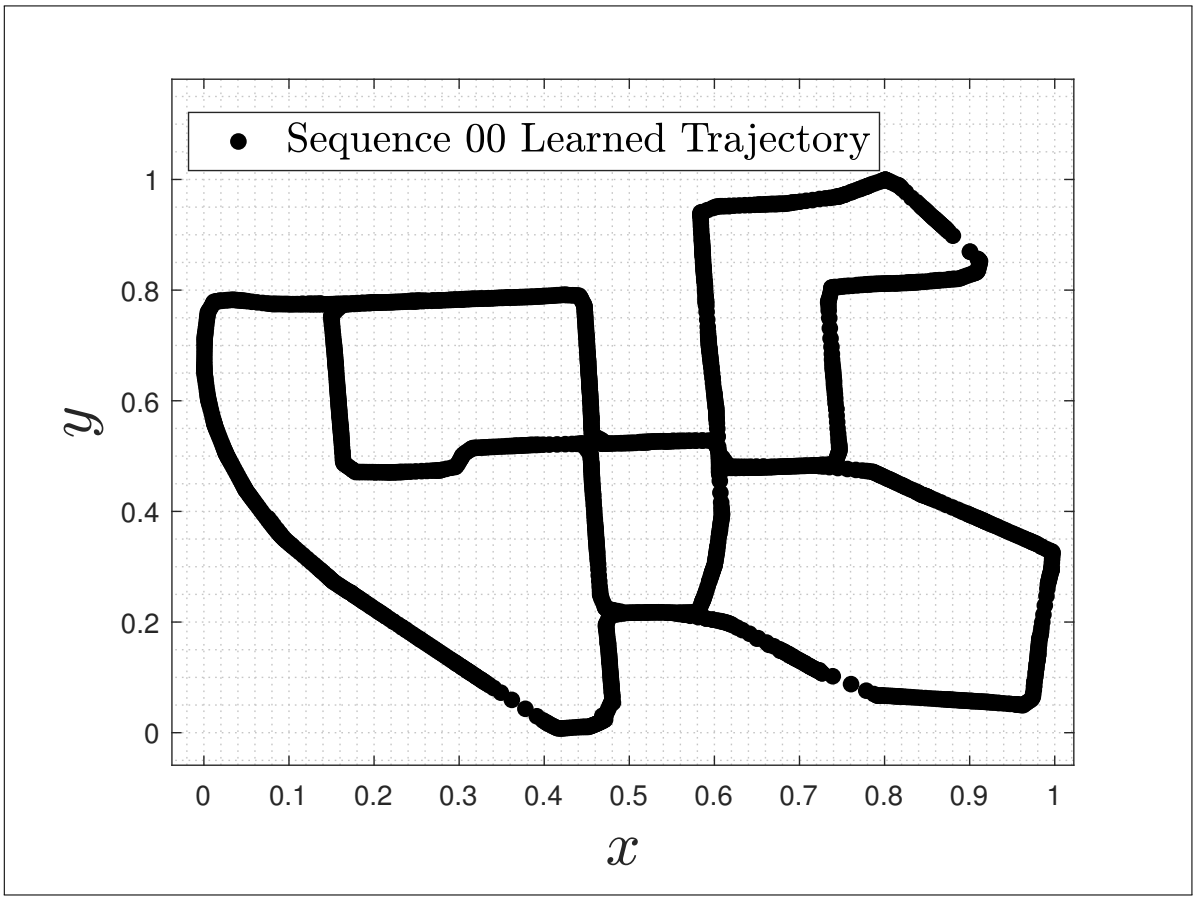
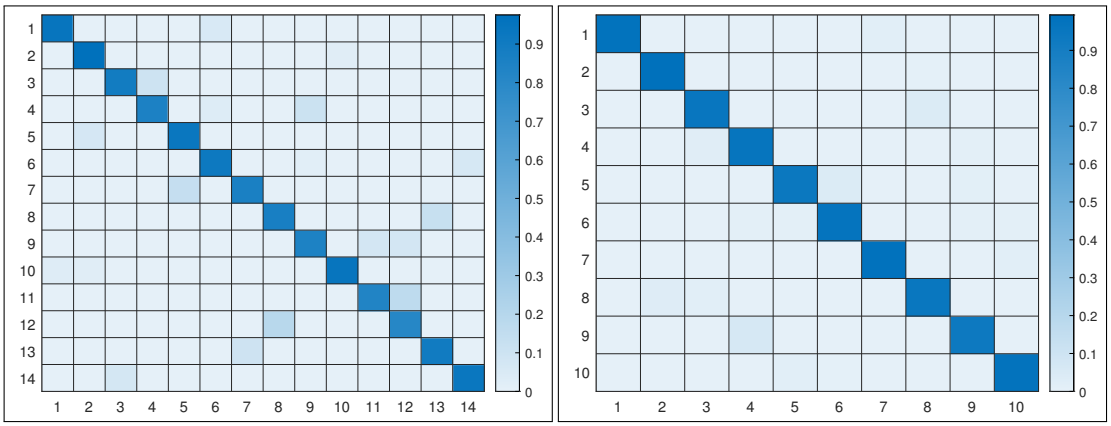


Figure 7.17: Kitti trajectory. WM guided MMP learned trajectory from video sequences. © 2025 IEEE.

The World Model (WM) learning process output is depicted in figs. 7.10 and 7.11, corresponding to the iCab and CARLA multi-agent vocabularies, respectively. As shown in the plots, the number of clusters are labeled with different colors, the mean of each cluster positions with black filled circles, and cluster velocities with red arrows (phase 1 of MASAA, see fig. 7.3). The movement between clusters is ruled by a transition matrix containing the transition probabilities as shown in figs. 7.18(a) and 7.18(b) for CARLA and iCab agents, respectively. Transition probability matrices are used to govern the learning of higher dimensional data, i.e., to assign the most probable WM states to each video sequence. These vocabularies are the output of the world model, that are then used to train the MMP model (phase 2 of MASAA).

The attention (guiding) vector in eq. (7.6), that governs the higher dimensional learning is a scaled version of the transition probability to assign the most probable WM states.

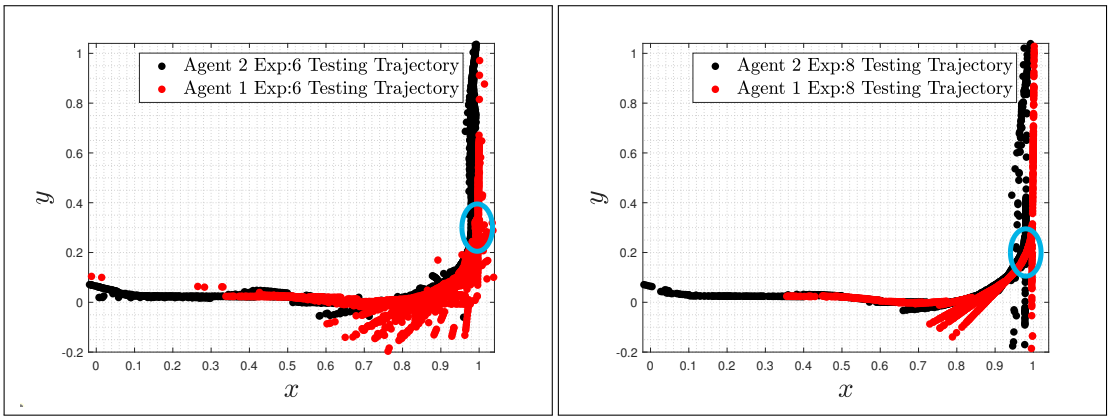
The results of the MMP module, shown in figs. 7.15,7.16, and 7.17, represent the best-learned dynamic coupling between attention focusing from the WM model, and experience



(a)Transition probability matrix of CARLA - WM.

(b)Transition probability matrix of iCab - WM.

Figure 7.18: (Transition probability Matrices. These probability matrices are used to govern the learning of higher dimensional data, i.e., to assign the most probable WM states to each video sequence. © 2025 IEEE.



(a)Testing result: Exp:06. Predictions of the trajectories corresponding to all particles.

(b)Testing result: Exp:08. Predictions of the trajectories corresponding to all particles.

Figure 7.19: (Predictions of the trajectories corresponding to all particles for testing experiments. The light blue oval shape represents the overtaking zone. © 2025 IEEE.

modeling from the MMP model. This allows the model to achieve the lowest RMSE error between the ground truth generalized states and the learned generalized states (see Table I).

These generated WM and MMP vocabularies, integrated through algorithm 2 (MAC-MJPF), corresponding to phase 3 of MASAA, are lower dimensional vocabularies (note that the output of the MMP model is also lower dimensional due to the VAE representation). This coupled knowledge is used to test the system in different environmental conditions (e.g., unseen heavy rain conditions) and in different overtaking regions (e.g., regions different from the learned overtaking regions). It also performs well when localizing the interacting agent

Table V: MAC-MJPF Complexity Analysis. MAC-MJPF performs best between 50 and 500 particles, with low mean RMSE of generalized states and within acceptable time. Time is in minutes.

Number of Particles			10	50	100	500	1000
Data Sets	iCab 420 Frames	Time	0.44	1.38	2.23	10.49	20.45
		RMSE	15.32	7.75	6.16	3.50	3.00
	Carla 180 Frames	Time	0.14	0.49	0.85	3.84	8.53
		RMSE	19.64	4.11	2.54	0.97	0.93
	Carla 160 Frames	Time	0.13	0.40	0.81	3.72	7.03
		RMSE	11.77	9.54	3.50	1.06	0.95
	Kitti 2761 Frames	Time	4.61	10.46	26.98	83.47	162.18
		RMSE	16.67	0.61	0.41	0.34	0.32
	Kitti 4541 Frames	Time	7.54	18.65	45.84	145.26	279.78
		RMSE	26.13	2.76	1.30	0.75	0.53

Table VI: Table displays the quantitative analysis of the overtaking model and following model, tested with overtaking data set, based on Root Mean Square Error (RMSE) measurement. We used the generalized training data as ground truth to compute the errors.

Experiment Types	**RMSE**				**Experiment No.**
	x	y	v_x	v_y	
Overtaking a1	1.7814	1.7560	0.8693	0.9646	06
Overtaking a1	1.5367	4.1848	0.8758	0.8644	08
Overtaking a2	1.7550	3.1502	0.0052	0.0049	06
Overtaking a2	1.7218	1.0127	0.0058	0.0184	08
Overtaking iCab1	0.9977	2.1985	0.0431	0.0363	05
Overtaking iCab2	1.8494	2.6795	0.0808	0.0981	05
Follow a1	24.9720	31.3412	2.0269	1.2849	09
Follow a2	61.6416	22.7203	3.6588	1.2832	09

states using Alg. 2 as shown in figs. 7.19(a) and 7.19(b). Moreover, table VI shows lower RMS Errors, which quantifies the performance of the algorithm.

Anomaly detection is a crucial step to guarantee a closed-loop incremental learning process. The results in figs. 7.13, and 7.14 show the capability of detecting anomalous signals that force the agent to correct its decisions. In the online testing phase, if there is an unexpected behavior and the predicted distributions are shifted compared to the learned model ones, the Active FP Model of the MASAA framework will be activated to correct

these bad actions. Furthermore, by applying thresholds on each anomaly signal, the agent can use a retraining module when it faces a completely new environment.

7.11 Comparison with state-of-the-art trajectory estimation methods

The estimation results presented above demonstrate the capability of the MASAA architecture to learn and use a generative model representation for task-based systems on multimodal data (video and odometry in our case). SLAM methods are the reference class of approaches suitable for fair comparisons. Specifically, deep learning-based SLAM methods (Li et al., 2022; Lai et al., 2020) are being studied to generate high-level semantics and build an agent's knowledge base to enhance the agent's perception and intelligence. One of the goals of this chapter is to evaluate how well the integrated WM, MMP, and Cost modules can work in the MASAA architecture to learn from data in an explainable representation. Hence, we compared the drift of state estimation errors in the agent state with some selected SLAM methods.

DSV-SLAM (Mo et al., 2022) is a LiDAR descriptor-based SLAM approach proposed to facilitate an efficient detection of loop closures for localization and mapping. In their work they used the KITTI dataset and achieved an RMSE of 3.693% on average. Other methods like UnDeepVO (Li et al., 2018) perform pose estimation by training deep neural networks in an unsupervised manner and attain an average of 4.07% RMSE drift. A self-supervised learning framework (Li et al., 2019) that represents frame-to-frame correspondence, for depth and pose estimation. In this work they also used the KITTI dataset, and their method has approximately 5% RMSE drift. These comparisons show that our method allows us to estimate the trajectories of interacting agents with lower RMSE drifts of 2.897%.

Compared to our previous work (Alemaw et al., 2022) presented in Chapter 6, the RMSE drift on the interacting agents on iCab dataset, is 6.1396% and on the CARLA dataset 3.875%. As can be seen in each GS variable in table VI, the drifts are much less than table II of Chapter 6. This is due to better parameter choice when training the WM vocabulary that guided the MMP model; the interaction assumption; and the two step threshold scenarios in the online interaction algorithm.

7.12 Summary

We proposed a cognitive architecture (MASAA), for autonomous agents. The architecture is a learning based system that is developed to be used for modeling self-aware systems. MASAA exhibits a distinctive characteristic in learning knowledge that represents dynamic and static interaction models, as demonstrated in the coupled systems. The incremental nature (behavior) of MASAA makes it possible to build fault-tolerant systems in autonomous driving scenarios when decisions like overtaking need to be made.

In the experiments, we focused to validate the MASAA architecture by learning data-driven localization of interacting agents, based from multi-sensorial data. Each component of the architecture is learned from data where all modules learn a hierarchy of inference and representation. During offline learning phase of MASAA, WM and MMP vocabularies are learned. The WM vocabulary guided the learning of the MMP vocabulary through attention mechanism.

Operational testing of MASAA through a particular overtaking interaction carried out online. The localization trajectories of the interacting agents are predicted only from the video sensory signals and the learned vocabulary. This type of testing which is only using the video modality further asserts the fail-safe characteristic of MASAA.

In the results most of the demonstrations are related to the learned knowledge, enabling an agent to localize itself, and to predict the GSs, i.e., position and velocity, of the neighboring agent from the videos in a self-supervised way. Through short-term memory model, MASAA allows the agent to store a multi-modal autobiographic memory. This can guide the agent to face new experiences by extracting the preferred observations from higher dimensional data that refine localization and anomaly detection for different interaction tasks. The remaining module, the Active FP Model, which is related to learning from action, is a component that we tackle in 8 with the online and incremental learning module of MASAA.

MASAA can be easily adapted for use in various applications and different research areas, such as wireless communications, where its characteristics could be exploited to detect abnormal activities within the spectrum induced by jammers (Krayani et al., 2020).

One can observe that the architecture is only tested based on a data-driven modular approach, as a disadvantage. As it is debatable to use the model free learning approaches for critical issues, like for self-driving problem domains, we agree that a modification in the communication or in the interlink between models is required.

Chapter 8

Incremental Learning and Decision Making in Continuous Action and State Space Models for Autonomous Vehicles

Generalizability and interpretability are major issues in designing and developing intelligent agents. Generalizability requires a clear understanding of one's own action (self-awareness) and a robust interaction with the environment (situation awareness). Many current studies are devoted in developing an algorithm that is more robust in generalizing unseen situations while explaining self-action. A novel methodology that fuses multi-sensorial data in a Hierarchical Dynamic Bayesian Network model in the Active Inference framework is proposed. An online model-based active learning is developed to represent and act in continuous and discrete state spaces.

This chapter concludes the MASAA architecture discussed in chapter 7, by developing the active first-person and the online incremental learning models. The development of these models build upon the other already developed and tested models for online incremental knowledge update.

We consider the learning and decision-making processes in the active stage as a Bayesian inference and decision-making problem using the modified MAC-MJPF algorithm. These problems can be solved by modeling perception and action selection procedures through VFE and EFE minimization techniques.

8.1 Introduction

Representing knowledge and inferring it for a particular course of action, like in a sequential state estimation of an agent, ranges from explicitly programming each step in the environment to learning from expert data acquired through different sensors. Autonomous systems are equipped with proprioceptive (steering, throttle, braking) and exteroceptive (camera, LiDAR) sensors to perceive their state and the surrounding environment. Integrating these sensors also with physiological ones (Aminosharieh Najafi et al., 2023b; Zontone et al., 2021; Aminosharieh Najafi et al., 2023a) further improves the driving behavior and the overall safety. This sensory information can be represented through random variables that take different values at each time step, linked through causal effect approaches in a hierarchical representation schema (Regazzoni et al., 2020). Modeling sequential data in generalized states has to couple continuous and discrete time series information hierarchically. This hierarchy allows a better representation of knowledge and eases performance measurement (anomaly detection) of each inference at each level, which is invaluable for continual learning. (Slavic et al., 2023; Ravanbakhsh et al., 2021). Hierarchical Dynamic Bayesian representation and inference networks (Cappelle et al., 2008; Han et al., 2009) are state-of-the-art methodologies for these types of problems. These networks are generative models based on the Bayes' principle as discussed in Chapters 6 and 7. Here, we include decision-making and hierarchical belief updating, which are the core benefits of an agent in combining both learning and decision-making into a unified process that minimizes the effect of outliers.

In this work we learn six Hierarchical Dynamic Bayesian Networks (H-DBNs) connected to each other for data representation and inference, as it can be seen in fig. 8.4: 1) Video State Network; 2) Odometry State Network; 3) Action State Network; 4) Coupled Video and Odometry Vocabulary Network; 5) Action Vocabulary Network; and 6) Configurator (Coupled Action, Video and Odometry) Vocabulary Network. Among these, the Action State Network, the Action Vocabulary Network and the Configurator Network are the innovative contributions on this work, compared to (Alemaw et al., 2025). Here, more emphasis is given to the active learning network, which has casual effect relations to all the other networks, excluding the configurator network, which has a dictionary relation. The active network, which includes both the action state and action vocabulary nodes, is developed based on the Active Inference framework (Friston et al., 2015). Unlike the other five networks, the configurator network is a dictionary composed of labels that connects the super states of action, odometry and video networks (vocabulary networks). Hence the action network does not have a direct causal effect relation with the configurator.

Deep Learning based autonomous driving models proposed in literature mainly use Reinforcement Learning (Hu et al., 2022), Imitation learning (Kim et al., 2020), Active

Inference (Friston et al., 2015), and a hybrid combination of these frameworks. The hybrid models are proposed to account for expert data insufficiency, minimizing the number of explorations (Li, 2021) and enhancing explainability (Nozari et al., 2022).

Imitation learning (IL) is a mechanism to learn a model offline and to use this model in online decision-making tasks. IL, in contrast to RL, uses expert knowledge to discriminate non-optimal actions without trial-and-error runs to get feedback from the environment. However, IL models suffer from cascading errors and distribution shifts since their models are trained only on a subset of the necessary samples (observation-action pairs). Considering self-driving vehicles, in this learning scenario, when these agents encounter slightly different lanes, they shift slightly to the left or right side of the road. This shift feedbacks new observations having more shifts to their models, disturbing their policy until no valid action can be taken anymore.

Reinforcement Learning (RL) is a process of training through an evolutionary method where an agent learns through trial and error actions to interact with the environment. This interaction provides a reward signal indicating the quality of the solution found. Deep RL methods interact with the environment and adjust driving strategies based on environmental feedback. However, Deep RL models (Gupta et al., 2021) require numerous interactions with the environment to ensure that the optimal policy is explored, which is less data efficient. Ideally, these methods can learn the optimal driving policy, but for an autonomous driving task in continuous action space, it requires an agent to spend a lot of time interacting with the environment to explore the optimal policy, and this is not efficient.

The incorporation of observations to action rules through energy-based models is a characteristic of the Active inference framework (Friston et al., 2015; Smith et al., 2022). In active inference, perception and action selection are inherently unified solutions. The state estimation and the resulting behavior can be harmoniously coupled in a unified framework as a minimization of VFE and EFE. The first step in active inference is to execute a pragmatic action based on the inferred state and observation that fulfills goals directly (i.e., exploitation), followed by performing an epistemic action (i.e., exploration), which discloses information that enables one to choose nonrandom action in the long run (future).

The active inference framework is based on the premise that perception and learning can be understood by minimizing a quantity known as variational free energy (VFE), and that action selection, planning, and decision-making can be understood by minimizing the expected free energy (EFE), which quantifies the VFE of various actions based on expected future outcomes (Smith et al., 2022). As discussed in (Friston et al., 2015), VFE is a resemblance of the capacity of a human brain in doing gradient descent at the time of perception, which can be associated to the mismatch between prediction and update

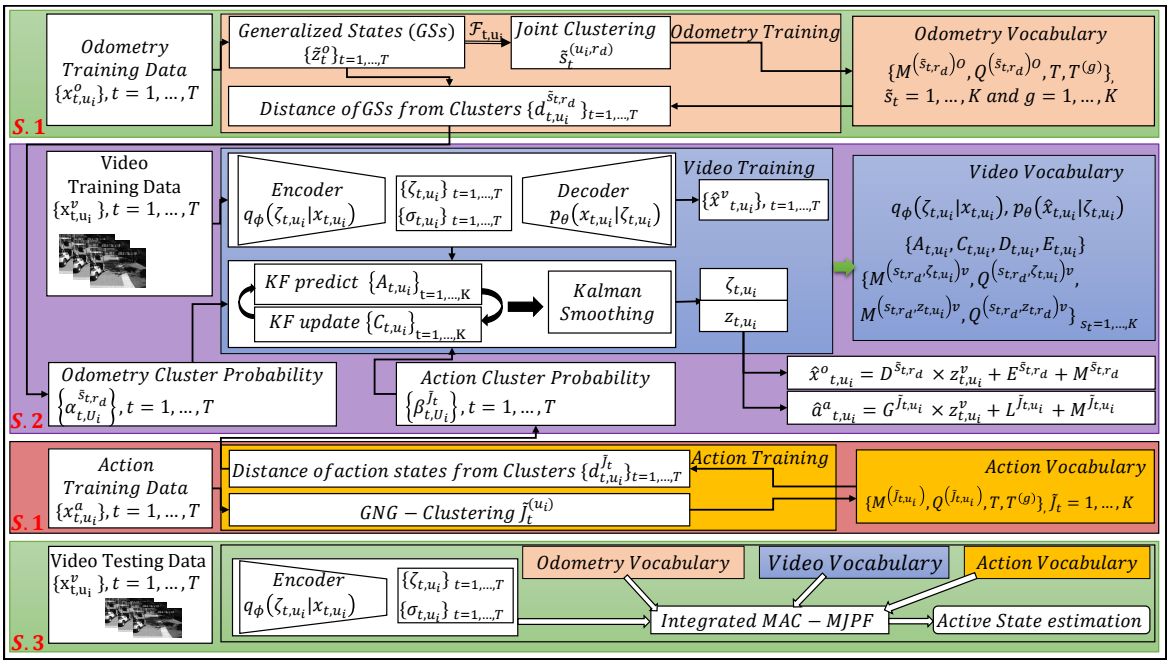


Figure 8.1: Integrated self-representation learning and decision-making from exteroceptive information driven by learned vocabulary of proprioceptive information. Lower-dimensional learning stage is represented as $S.1$, higher-dimensional learning stage is represented as $S.1$, and $S.3$ is online learning, inference and decision making process. © 2024 IEEE.

messages. The other quantity, i.e., EFE, models the rule of action selection, which is a policy predicting the sequences of feature states and observations for each possible action (policy), by computing the expected free energy (EFE) and selecting the policy that has the minimum value of EFE corresponding to those predicted future states and observations.

In this work we follow a different approach than the original one proposed for VFE and EFE (Friston et al., 2015). Here, we consider the learning and decision-making processes in the active stage using Coupled Markov Jump Particle Filters detailed in Alg. 5 to model perception and action selection procedures through VFE and EFE. In the original work, action is treated as a discrete state which is only optimal for simple tasks. We instead model both continuous and discrete action states. Action representation, inference, and selection is a process of inference where an agent selects actions by inferring a distribution over control states by using particle filters (estimating discrete action states) and Kalman filters(inferring continuous action states).

8.2 Method Description: Active Learning and Inference

This chapter describes the MASAA Active First-Person model and online and incremental learning subsystems. All the other subsystems discussed in chapter 7 are integrated in this chapter. The learning stage has two main stages. The first stage is offline learning as detailed in chapters 6 and 7 with the addition of the action offline learning subsystem. The second stage is the online and incremental learning phase which updates every module in an integrated way.

The learning algorithm starts from an intensive offline learning procedure considering a given set of data. These include video data acquired from an onboard sensor (First Person Viewpoint - FPV), the corresponding odometry data, and the sequences of action data from a moving vehicle driven by an expert driver. The proposed method consists of three stages (see fig. 8.1): stage 1 includes the lower dimensional odometry and action vocabulary learning; stage 2 the higher dimensional vocabulary learning; and stage 3 the active learning stage.

8.2.1 Odometry Learning

The general assumption in odometry and action learning is to consider the generalized coordinates, which are a set of parameters that can specify the configuration of the dynamics of a physical system as a time derivative of its state (as also in chapters 3, 6 and 7). Generalized coordinates are used to develop the equation of motion of a system as a function of time. They are simplifications of the motion equation in the assumption of Lagrangian mechanics with the principle of least action (Friston et al., 2008). As detailed in chapter 2, in generalized coordinate systems, having positional stationary points $[x_t, y_t]$, one can apply the Null Force Filter (Iqbal et al., 2021), which computes the n^{th} order derivatives of the positional data. As detailed in the previous chapter (chapter 7), the odometry learning is the same as the WM learning in section 7.7. As a result, the output of Odometry learning is shown as a clustered positional trajectory of a rectangular motion in fig. 8.2.

8.2.2 Action Learning

In stage one, the second network, which deals with the action vocabulary learning, is learned from the output of control information. This comprises throttle, steering angle, and brake information. The action datasets are extracted from the CARLA (Dosovitskiy et al., 2017) simulator which is programmed to record the action information being sent to the actuators of the interacting agents. One thing to note here is that the video, odometry, and action datasets are recorded from this simulator being synchronized to record this information. This means

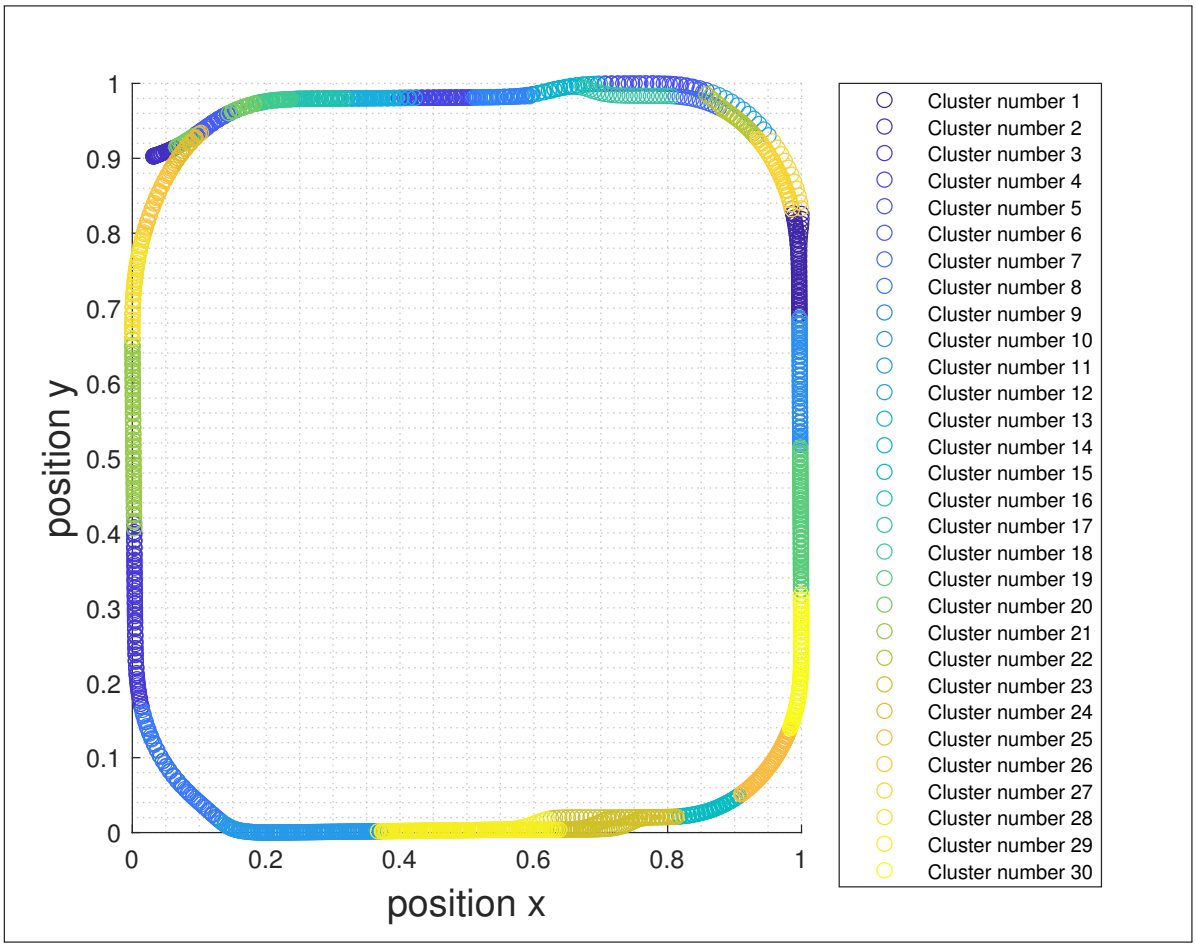


Figure 8.2: Clustered trajectory of an overtaking agent showing the output of odometry cluster information. © 2024 IEEE.

that, at each point in time, it records a video frame, an odometry point, and the action taken at that point.

The sequence of action data is clustered using the M-GNG algorithm (Iqbal et al., 2021) to account for the discrete action space. This allows us to create the action vocabulary which includes *i)* action cluster mean $M^{(\tilde{J}_k)a}$; *ii)* action cluster covariance $Q^{(\tilde{J}_k)a}$; *iii)* action transition matrix T defining the probability of transition from one cluster to the other; *iv)* action transition time $T^{(g)}$ defining the time spent in each cluster before switching (moving) to the next cluster.

Learning lower dimensional data helps the process of learning the higher dimensional video data to establish a relation at the semantic label. The video cluster learning is guided by the odometry vocabulary that generates a unified network for both video and odometry modalities.

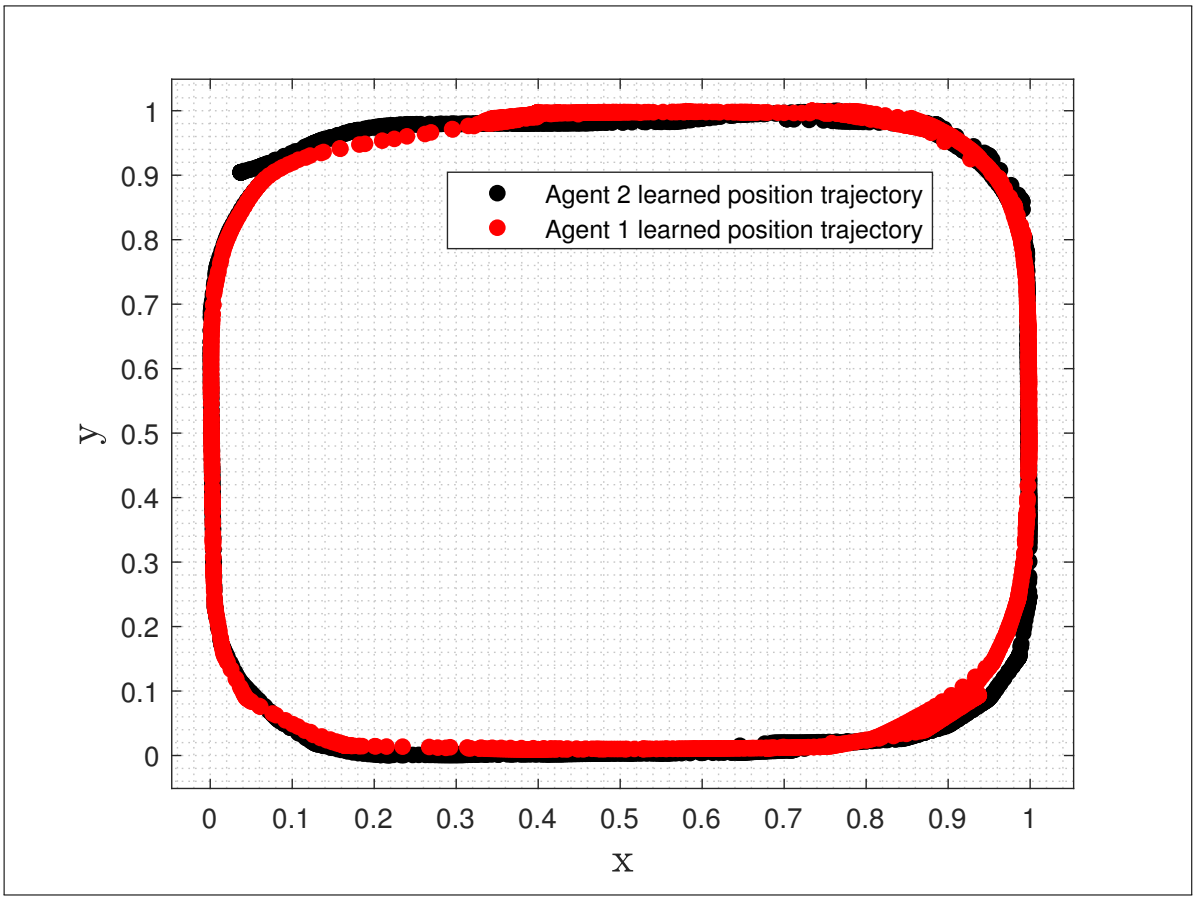


Figure 8.3: Learned trajectory from stage 2 of fig. 8.1 (video learning). Cluster guided trajectory © 2024 IEEE.

8.2.3 Video Learning

The video vocabulary learning model is controlled by the odometry vocabulary to focus the attention of learning towards the best features that represent both the content of the video and the motion, at each consecutive steps. This process is the same process that learns the MMP model, presented in chapter 7. The novelty in this chapter is the action modality introduced to learn MMP model based on a guiding action vector. Hence, we have two MMPs as a multi-modal perception. The first MMP is learned through the odometry attention focusing process, and the second MMP is learned through the action based attention focusing process. Learning the MMP in this way gives an advantage when the environment is new. Since the action space is totally proprioceptive information, if it is coupled with enough maneuvering data it can generalize well in any environment within the action parameters. Instead of referring to the previous chapter for odometry based MMP learning, we reused and write it here for clear difference with the action based MMP.

Odometry based MMP: the learning step involving the odometry modality, we use the learned vocabulary to train the video models. For this purpose, the Kalman Variational Autoencoder (KVAE) (Fraccaro et al., 2017) is employed to obtain the latent states $a_t^{(u_i)v}$ corresponding to the video of the training data $x_t^{(u_i)v}$. we have two hidden states, i.e., $a_t^{(u_i)v}$ and $z_t^{(u_i)v}$, which represent the content of an image and its corresponding dynamics, respectively. While training, we learned other pseudo observation matrices, i.e., D and E, from the latent distribution and the odometry vocabulary. These matrices encode the information of the location parameters of interacting agents; therefore, we used four matrices (A, C, D, and E). The trained odometry vocabulary $\tilde{s}_k^{(u_i)O}$ is employed to guide the model to assign a semantic category to the latent states. The two latent states ζ_{t,u_i} and z_{t,u_i} are modeled according to eq. (7.4) and eq. (7.5) of Chapter 7, respectively. Note that, due to the introduction of the action variable, the content-related video latent state variable a, is replaced with ζ, which is also evident in figs. 8.1 and 8.4. As a result, the learned odometry vocabulary guided trajectory from video sequences is depicted in fig. 8.3. The losses introduced in chapters 6 and 7 for video learning are used here as well.

Action based MMP: Similarly the video training can be guided, using the action vocabulary, through a probabilistic guiding vector:

$$1/\beta_{t,u_i}^{\tilde{j}_k} = \left(d_{t,u_i}^{\tilde{j}_k}\right)^n \times \sum_{k=1}^{K} \frac{1}{\left(d_{t,u_i}^{\tilde{j}_k}\right)^n}, \tag{8.1}$$

where $d_{t,u_i}^{\tilde{J}_k} = \mathscr{D}_M((z_{t,u_i}^v),(M^{(\tilde{J}_k)a},Q^{(\tilde{J}_k)a}))$ is the Mahalanobis distance ($\mathscr{D}_M$) between the video latent state and the mean of the action cluster. The action based latent states are mathematically modeled in the same way as in eq. (7.4) and eq. (7.5) which are detailed in Chapter 7. The difference here is the odometry guiding vector $\alpha_{t,u_i}^{\tilde{s}_k}$ is changed to action guiding vector $\beta_{t,u_i}^{\tilde{j}_k}$. Additionally, the content related latent state variable a is changed to ζ, as in this chapter a is an action variable.

When we couple action and video in the training, we learned relational matrices, i.e., G and L, from the latent distribution and the action vocabulary. From these matrices we can build the action that is taken on that particular video frame. Mathematically:

$$\hat{a}_{t,u_i}^a = G^{\tilde{J}_{t,u_i}} \times z_{t,u_i}^v + L^{\tilde{J}_{t,u_i}} + M^{\tilde{J}_{t,u_i}} \tag{8.2}$$

Relating video modality ro action modality gives a flexibility to choose an action even if the odometry signal is lost. In active inference when exploring in new environments, this relation is vital. If the odometry anomaly is very high, the agent is believed to be in a new region. In such regions the learned actions are not included in the training phase, but this

does not force the agent to take a random action, instead it computes eq. (8.2) and take this action, which will generate similar actions based on the motion of the video information.

Algorithm 3: Action Selection:

action selection is based on epsilon-greedy policy
$\alpha_t = \max_n W_{t,n}$
$\delta_1 \leftarrow threshold$ (0.25) based on odometry anomaly
$\delta_2 \leftarrow threshold$ (0.4) based on video anomaly
$\varepsilon_t = 1 - \alpha_t$
if $\varepsilon_t < \delta_1$ **then**
    $n_{selected} \leftarrow arg\min G^{(u_i)}_{\theta_{t,n}}$ according to eq. (8.3)
    $\tilde{a}^{(u_i)a}_t = \tilde{a}^{(u_i)a}_{t,n_{selected}}$
else
    if $\varepsilon_t < \delta_2$ **then**
        Sample from replay memory: $R(x^{(u_i)v}, x^{(u_i)o}, a^{(u_i)a})$
        Predict states using alg. 5, with the sampled states
        $n_{selected} \leftarrow arg\min G^{(u_i)}_{\theta_{t,n}}$ using eq. (8.3)
        $\tilde{a}^{(u_i)a}_t = \tilde{a}^{(u_i)a}_{t,n_{selected}}$
    else
        Uniform sample based on min and max action space of the agent
        $\tilde{a}^{(u_i)a}_t \leftarrow Random(\max_a, \min_a)$
    end if
end if
Environment and action set up (see Alg. 4) $next_{obs}, next_{odom}, reward, done = env.step(action)$
$update_{ReplayMemory}(state, odom, action, nextState, nextOdom, reward)$

8.2.4 Active Learning and Inference

In this stage, which is the last one, the agent is equipped with an environmental model from the learned vocabularies. As it can be seen in fig. 8.4, the inference related to the perception model can be carried out in two ways. The first one is the prediction from the same state in different time steps, meaning from time t to time $t+1$ for each state and super state (from perceptual state to perceptual state at a different time), while the second inference is the prediction from the action state at the same time instant (the effect of the predicted action on the perceptual states). These two ways of inferences can give the learned model the chance to have different signals for anomaly measurements from multi-level inference mechanisms. This can be useful to select an action by checking the impact of the selected action on the current states.

In our models, to integrate action and perception models, we use the Multi-Agent Coupled Markov Jump Particle Filters (MAC-MJPFs, see Alg. 5). Inferring and integrating these learned models with decision-making is also blended in this algorithm. We chose these filters because they are integrated and well-suited to adapt particle estimations by fast reinitialization when the agent changes its motion. As a single integrated multi-filter, this

Algorithm 4: Environment Configuration:

$control_{throttle} = action_{throttle}$
$control_{steer} = action_{steer}$
$control_{brake} = action_{brake}$
Acceleration regulation:
if $action_{throttle} > 0$ **then**
    $control_{throttle} = min(control_{throttle}, maxThrot)$
    $control_{brake} = 0.0$
else
    $control_{throttle} = 0.0$
    $control_{brake} = min(abs(control_{throttle}), maxBrake)$
end if
Steering regulation:
if $control_{steer} > 0$: **then**
    $control_{steer} = \min(maxSteer, control_{steer})$
else
    $control_{steer} = \min(-maxSteer, control_{steer})$
end if
$vehicle_{control}(control)$
$vel = vehicle_{velocity}$
$Kmh = 3.6 * \sqrt{vel.x^2 + vel.y^2 + vel.z^2}$
if $collision$: **then**
    done = True
    reward = -1
else
    $done = False$
    $reward = 1$
end if
if $SECONDS\_PER\_EPISODE < time.time()$: **then**
    $done = True$
end if
Output: image, vehicle location, reward, done

filter allows updating each estimate after each measurement to get an updated state and state covariance, by combining the most likely models based on their weights.

Given the input of both vocabularies, the agent sends video information to the active learning model. The model first gets the video latent state and initializes each super state (S_t, J_t and D_t) based on the probabilistic distance vectors (α_t and β_t). After the initialization, it predicts the next super states using the particle filter, and the next continuous states using the Kalman filter. It then updates each prediction after the action selection. After the update phase, since we are using a particle filter to account for particle degeneracy problem (Elfring et al., 2021), we apply a resampling step with a replacement, based on a threshold, after each update step. More specifically, we do not resample after each update step if the number of particles that have nearly zero weight is less than 25% of the total number of particles. This allows us to improve the overall performance. To perform the update step we need to receive a new observation, and for that we need to apply an action.

To test the predictability of action information, we tested the action sequence prediction (see fig. 8.5 (b)) and update (see fig. 8.6 (b)) phases with the expert action demonstration. Comparing this predictability of action sequences with the actual expert sequence of actions

Algorithm 5: AIF for Interaction:

Input: $M^{(\tilde{s}_k)o}, Q^{(\tilde{s}_k)o}, T, T^{(g)}, M^{(s_k,a_t)v}, M^{(s_k,z_t)v}\ Q^{(s_k,a_t)v}, Q^{(s_k,z_t)v}, x_t^{(u_1)v}, M^{(\tilde{j}_k)a}, Q^{(\tilde{j}_k)a}$
for $t = 1,\ldots,\tau \leftarrow$ *Time evolution* **do**
  Obtain image latent state using VAE: $a_t^{(u_1)v}$, $\sigma_t^{(u_1)v}$ and
  Obtain latent state covariance
  Calculate probabilistic *guiding vector*
  Begin Filtering
  if $t == 1$ **then**
    INITIALIZATION OF PARTICLES:
    for $n = 1,\ldots,N \leftarrow$ *number of Particles* **do**
      Initialize cluster assignment $\tilde{s}_{t=1,n}$ of particle n
      using cluster probabilities $\alpha_{v,t}^{(\tilde{s}_k^{u_i})}$ with $\tilde{s} = 1,\ldots,K$
      Initialize n^{th} particle video value as:
      $z_{t=1,n}^{v} \sim \mathcal{N}(M^{(s_{1,n},z)v}, Q^{(s_{1,n},z)v})$
      Initialize n^{th} particle odometry value as:
      $z_{t=1,n}^{o} \sim \mathcal{N}(M^{(s_{1,n},z)o}, Q^{(s_{1,n},z)o})$
      Initialize n^{th} particle action value as:
      $a_{t=1,n} \sim \mathcal{N}(M^{j_{1,n},a}, Q^{j_{1,n},a})$
      Initialize n^{th} particle configurator value as:
      $c_{t=1,n} \sim \mathcal{N}(M^{c_{1,n},c}, Q^{c_{1,n},c})$
    end for
  else
    UPDATE:
    for $n = 1,\ldots,N \leftarrow$ *number of Particles* **do**
      Perform video update:
      $z_{t|t,n}^{(u_i)v}, \Sigma_{t|t,n}^{(u_i)v} =$
      $KF_{update}^{Video}(z_{t|t-1,n}, \Sigma_{t|t-1,n}, C_{t,n}^{s_t}, a_{t,n}, \mathscr{R}_{t,n}^{v})$
      Obtain odometry estimation from video:
      $\tilde{z}_{t|t,n}^{(u_i)o} = D^{(\tilde{s}^O,O)} * z_t + E^{(\tilde{s}^O,O)} + M^{(\tilde{s}^O,O)}$
      Perform odometry update:
      $\tilde{z}_{t|t,n}^{(u_i)o}, \Sigma_{t|t,n}^{(u_i)o} =$
      $KF_{update}^{Odom}(\tilde{z}_{t|t-1,n}, \Sigma_{t|t-1,n}, \hat{\tilde{z}}_{t|t-1,n}, \mathscr{R}_{t,n}^{o})$
      Perform action update:
      $\tilde{a}_{t|t,n}^{u_i}, \Sigma_{t|t,n}^{u_i} =$
      $KF_{update}^{Action}(\tilde{a}_{t|t-1,n}, \Sigma_{t|t-1,n}, \hat{a}_{t|t-1,n}, \mathscr{R}_{t,n}^{a})$
    end for
    Calculate anomalies.
    Re-weight *v*,*o*,*a* particles weights based on anomalies
    Re-sample if:
    $\sum_n 1/w_n^2 <$ threshold (we used 0.25)
  end if
  PREDICTION:
  for $n = 1,\ldots,N \leftarrow$ *number of Particles* **do**
    Predict vocabulary of video and odometry:
    $s_{t+1,n}^{(u_i)v,o} = s_{t+1,n}^{(u_i)v,o} | s_{t,n}^{(u_i)v,o} \sim T(s_{t,n}^{v,o})$
    Predict vocabulary of action:
    $J_{t+1,n}^{(u_i)a} = J_{t+1,n}^{(u_i)a} | J_{t,n}^{(u_i)a} \sim T(J_{t,n}^{a})$
    Predict vocabulary of configurator:
    $D_{t+1,n}^{(u_i)c} = D_{t+1,n}^{(u_i)c} | D_{t,n}^{(u_i)c} \sim T(D_{t,n}^{c})$
    Predict Video state:
    $z_{t+1|t,n}^{(u_i)v}, \Sigma_{t+1|t,n}^{(u_i)v} = KF_{pred}^{V}(z_{t|t,n}^{(u_i)v}, \Sigma_{t|t,n}^{(u_i)v}, s_{t+1,n}^{(u_i)v,o})$
    Predict odometry state:
    $\tilde{z}_{t+1|t,n}^{(u_i)o}, \Sigma_{t+1|t,n}^{(u_i)o} = KF_{pred}^{O}(z_{t|t,n}^{(u_i)v}, \Sigma_{t|t,n}^{(u_i)v}, \tilde{s}_{t+1,n}^{(u_i)v,o})$
    Predict action state:
    $\tilde{a}_{t+1|t,n}^{(u_i)a}, \Sigma_{t+1|t,n}^{(u_i)a} = KF_{pred}^{A}(a_{t|t,n}^{(u_i)a}, \Sigma_{t|t,n}^{(u_i)a}, D_{t+1,n}^{(u_i)c})$
    $\tilde{a}_{t+2|t+1,n}^{(u_i)a}, \Sigma_{t+2|t+1,n}^{(u_i)a} = KF_{pred}^{A}(a_{t+1|t+1,n}^{(u_i)a}, \Sigma_{t+1|t+1,n}^{(u_i)a}, D_{t+2,n}^{(u_i)c})$
  end for
  Action selection: See Alg. 3
end for
Output: Optimal Policy

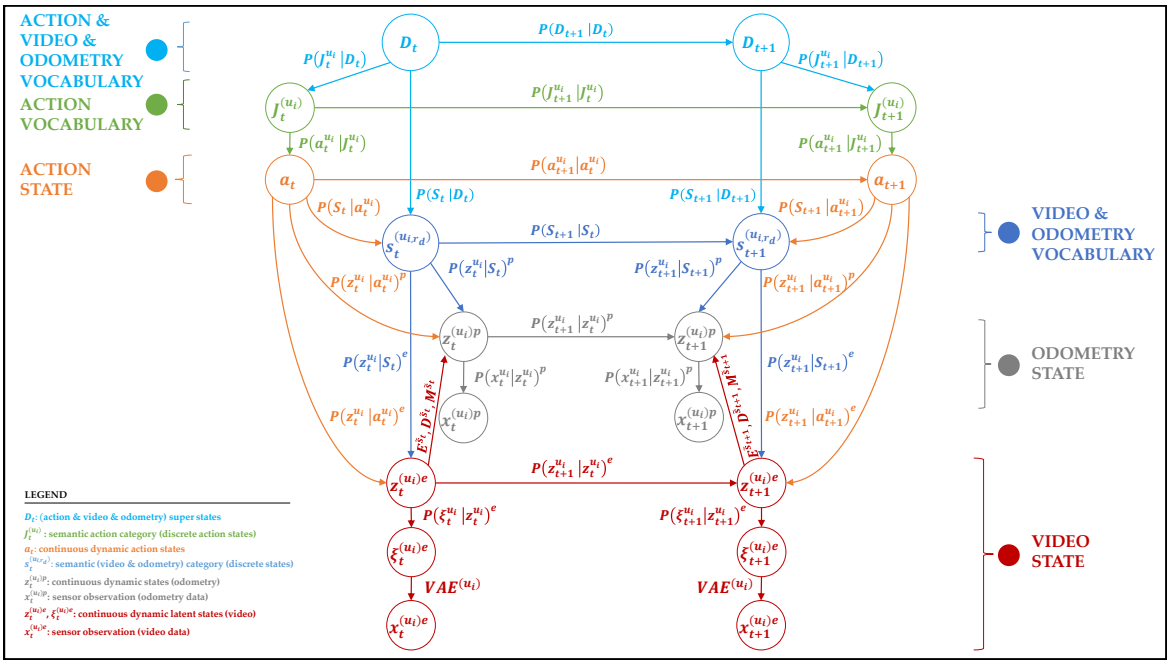
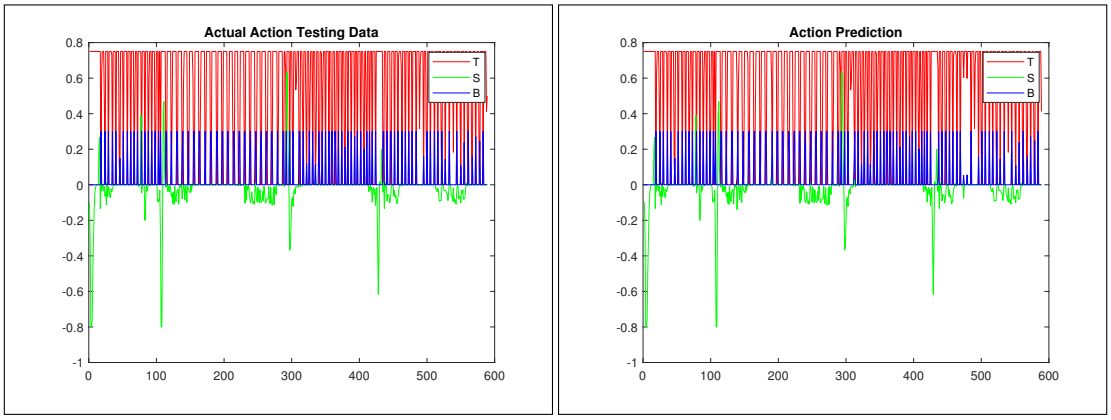


Figure 8.4: Learned HC-DBN. The agent now has a combined vocabularies of action, odometry, and video, that are mapped by the configurator network. © 2024 IEEE.

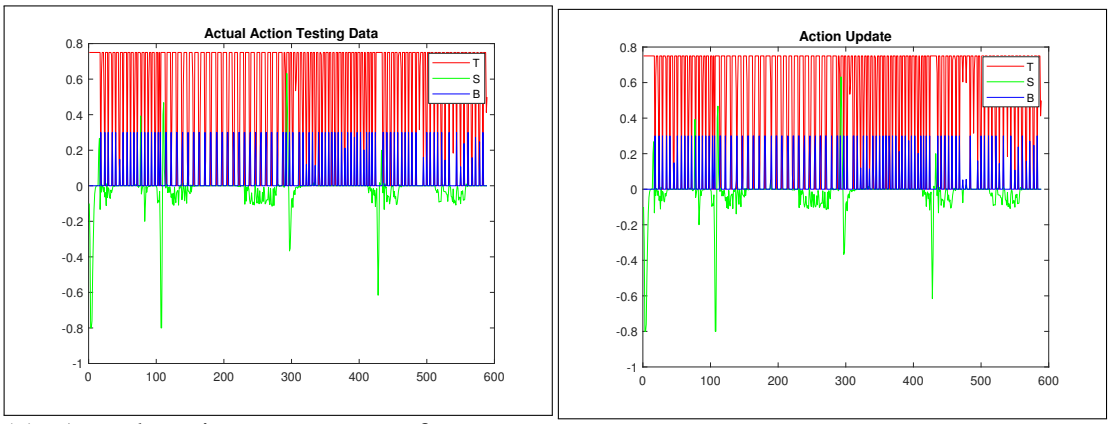


(a) Actual action sequence of expert agent.

(b) Predicted action sequence of ego agent.

Figure 8.5: Expert agent action sequences and predicted action sequences of an ego agent. © 2024 IEEE.

shown in figs. 8.5 (a) and 8.6 (a) (both are the same sequences), we can notice that our system exhibits a very low RMSE of 0.0010 when tested offline. This low error asserts the predictability of actions, and hence, we used the new action sequences shown in fig. 8.8 (a) to test the online policy learning and action selection performance.



(a) Actual action sequence of expert agent.

(b) Updated action sequence of ego agent.

Figure 8.6: Expert agent action sequences and updated action sequences of an ego agent. © 2024 IEEE.

Action selection (Alg. 3) is performed using the free energy principle. To select an action, the configurator network should first combine the predicted action super state, and the predicted combined odometry and video super states, to select the most probable state. Then it can select the higher weighted particle and associate it with the continuous action. This maximum weight is used to choose between exploration (new actions) and exploitation (the learned action).

In active inference, to select an action and update the states, Friston (Friston et al., 2015) used, the sum of the scalar product between the logarithmic value of the preferred observation and the predicted state, as an EFE measure. In our case, we use the learned trajectory from video learning stage as a preferred state and observation, and for each prediction of the particles, we measure the Mahalanobis distance between the preferred generalized state and the predicted generalized state:

$$G_{\theta_{t,n}}^{(u_i)} = D_M(\hat{z}_t^{(u_i)o}, \tilde{z}_{t,n}^{(u_i)o}) \tag{8.3}$$

where $n = 1, \cdots, N$ is the total number of particles at that time instant. To select the action, we use the minimum index of $G_{\theta_{t,n}}^{(u_i)}$ and select that particle from the predicted action particles.

8.3 Multi-Level Messages

Messages in H-DBNs can be grouped in two major categories: 1) predictive messages and 2) diagnostic messages. As shown in fig. 8.7, $\pi(S_{k+1})$ and $\pi(\tilde{z}_{k+1})$ are predictive messages while $\lambda(S_{k+1})$ and $\lambda(\tilde{z}_{k+1})$ are diagnostic messages. Predictive messages are messages that

are used to predict next time super states ($\pi(S_{k+1})$), and generalized states ($\pi(\tilde{z}_{k+1})$). The diagnostic messages are key to measure the predictive capacity of a model and the sensory observations. The predicted super states ($\pi(S_{k+1})$) and generalized states ($\pi(\tilde{z}_{k+1})$) will be ascertained by the observed super states ($\lambda(S_{k+1})$) and generalized states ($\lambda(\tilde{z}_{k+1})$).

These messages are the essences of anomaly signal measurements, as they are hierarchically integrated to each other. Anomaly signals, in each hierarchy, measure the mismatch (amount of surprise) in each prediction and update combined steps. Based on the type of distributions we use different probabilistic distance measures. In fig. 8.7 assuming S to be a discrete state, $\pi(S_{k+1})$ and $\lambda(S_{k+1})$ can be used to measure discrete-level anomaly. Taking $\tilde{z}$ as a generalized continuous state, $\pi(\tilde{z}_{k+1})$ and $\lambda(\tilde{z}_{k+1})$ can indicate continuous-level anomaly signals. In fig. 8.7, some probabilistic distances can be evidenced:

(A) A Bhattacharyya distance between predicted state and observed state (observation mapped to generalized state) with the corresponding covariance matrix:
$D_B(\tilde{z}_{k+1}, \Sigma\tilde{z}_{k+1}, z_{k+1}, \Sigma z_{k+1})$.

(B) A Bhattacharyya distance between the predicted mean generalized super state and the generalized state with the corresponding covariance matrix:
$D_B(\tilde{z}_{k+1}, \Sigma\tilde{z}_{k+1}, \tilde{s}_{t+1}, \Sigma\tilde{s}_{k+1})$.

(C) A Symmetric Kullback–Leibler Divergence:
$D_{KL}(\pi(\tilde{s}_{k+1}) \,||\, \lambda(\tilde{s}_{k+1})) + D_{KL}(\lambda(\tilde{s}_{k+1}) \,||\, \pi(\tilde{s}_{t+1}))$.

8.4 Online Decision Making

Bayesian inference represents the optimal way to infer posterior beliefs within a generative model. As we discussed it in chapter 3, Bayes theorem is computationally intractable since it requires the summation of observations under all possible states in the case of discrete states, while in the continuous states it requires the evaluation of integrals that may not have analytical solutions. To solve these kinds of problems approximation methods are necessary. The approximation process enables us to produce many proposal distributions with sample parameters from the prior, and to select the best representation of the original distribution using different probabilistic distance measurements. VFE is one of the measurement methods for selecting the best fitted distribution from the proposal functions that has minimum VFE value. One common way to measure this requires introducing an information-theoretic quantity known as self-information or surprisal. Surprisal reflects a deviation between observed outcomes and those predicted by a model. It is typically written as the negative

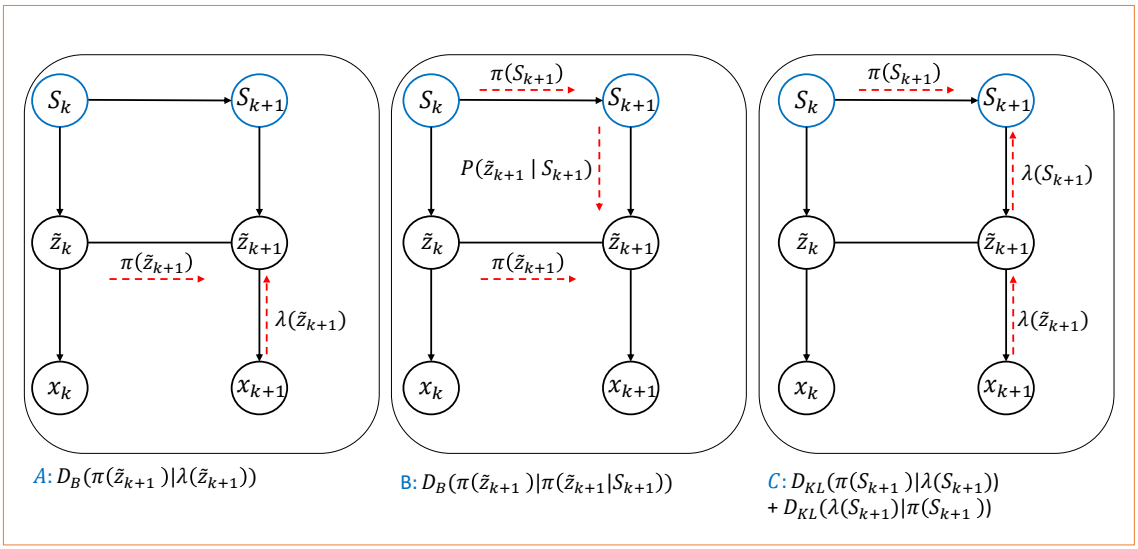
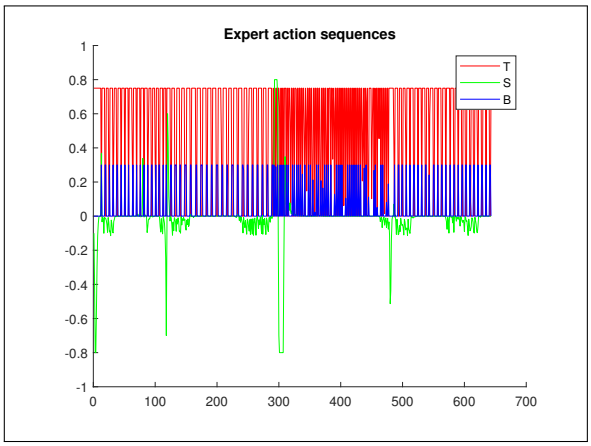


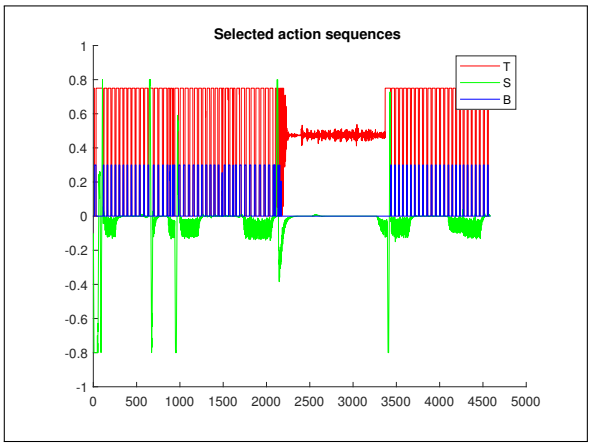
Figure 8.7: Multi-level anomaly signal measurement models: A) anomaly signal indicator at the observations and the predicted generalized states; B) anomaly signal indicator between predictive message; C) anomaly signal indicator at cluster level between predictive and diagnostic messages;

log-probability of that observation, i.e., $-\ln p(o|m)$ (where o, is the observation, and m is the model). The active inference approach has two stages. Stage one is modeling of categorical or continuous inference, while stage 2 is modeling inference of optimal actions. In (Friston et al., 2015) policies or sequences of actions are represented by π, changing the generative model that is the joint distribution indicating all possible combination of states and observations $p(o,s) = p(o|s)p(s)$ to $p(o,s,\pi) = p(o|s,\pi)p(s|\pi)p(\pi)$.

In active inference there is no reward or value but instead there is a distribution known as *prior preference distribution* $p(o_{preferred})$. One thing to note here is that policy in active inference framework is different from the model free reinforcement learning which maps state to action. Active inference only infers possible allowable sequences of actions based on the last-time action. The value of each policy in active inference is selected from a policy distribution which has minimum expected free energy.



(a) Throttle,steering angle and brake sequences of an expert agent.



(b) Throttle, steering angle, and brake sequences of online selected actions.

Figure 8.8: Preferred expert agent action sequences vs learned action sequences of an ego agent.

8.4.1 Variational Free energy

In Bayesian online decision making the best fit model is the one having low approximation error (VFE). The approximate distribution is an arbitrary distribution over the state that is updated to match the true posterior distribution. To measure the similarity, different probabilistic distance measurements can be used based on the type of the distribution, as we highlighted in section 8.3. For discrete distribution *Kullback–Leibler (KL) divergence (D_{KL})* and for continuous distributions *Bhattacharyya distance (D_B)* can be used.

A probabilistic distance can be utilized to evaluate how much a prediction model fits the observation sequence. It is important to evaluate how much the real observations support the predictions performed at the continuous level, which leads to detect any abnormal behavior occurred in the surrounding environment (Krayani et al., 2020).

VFE can be derived using probabilistic distance measures:

$$F_\theta = \mathbb{E}_{q(s|\theta)}\left[\ln \frac{q(s|\theta)}{p(o,s|\theta)}\right] \tag{8.4}$$

Eq. (8.4) is the same as eq. (4.4) in section 4.6, but here it is conditioned on policy parameter θ. VFE represents the measures of prediction errors, and can be expressed as a log difference:

$$F_\theta = \mathbb{E}_{q(s|\theta)}\left[\ln q(s|\theta) - \ln p(o,s|\theta)\right] \tag{8.5}$$

This is KL divergence between the approximate posterior and the generative model as a log difference. To marginalize the likelihood using the product rule of probability, VFE can be expressed as:

$$F_\theta = \mathbb{E}_{q(s|\theta)}\left[\ln q(s|\theta) - \ln p(s|\theta)\right] - \mathbb{E}_{q(s|\theta)}\left[\ln p(o|s,\theta)\right] \tag{8.6}$$

Assuming the independence between the likelihood and the policy parameter θ, eq. (8.6) is the KL divergence between prior and posterior beliefs including the predictive accuracy associated with the probability of observation and model evidence. It is an optimization phase between minimizing prediction error with minimum complexity in changing belief.

$$F_\theta = D_{KL}\left[q(s|\theta)||p(s|\theta)\right] - \mathbb{E}_{q(s|\theta)}\left[\ln p(o|s)\right] \tag{8.7}$$

Rearranging 8.4 it is possible to show that VFE is an upper bound on surprisal:

$$F_\theta = \mathbb{E}_{q(s|\theta)}\left[\ln q(s|\theta) - \ln p(s|o,\theta)\right] - \ln p(o|\theta) \tag{8.8}$$

VFE is a combination of the average mismatch between the posterior and prior beliefs discounted by the mismatch between the predicted and observed outcomes.

8.4.2 Expected Free energy

The expected free energy is the dot product between the natural logarithm of the preferred observation (probability of observation at each time step) and the probability of the expected observation. In MJPF the probabilities are weights of each particle. The policy with the highest posterior probability (given preferred outcomes) is typically chosen, which couples the agent back to the generative process by changing the true state of the world through actions.

In active inference, since the developed models try to optimize perceptual error, one way of achieving it is to select best actions that will bring to a preferred observation associated with the state. Since there is no punishment or reward, best actions are selected based on prior expectations over observations. Hence an action expected to produce preferred observations is the one that minimizes EFE and maximizes model accuracy:

$$G_\theta = \mathbb{E}_{q(o,s|\theta)}\left[\ln q(s|\theta) - \ln p(o,s|\theta)\right] \tag{8.9}$$

As in eq. (4.6) in section 4.6, the EFE is expressed as the expected difference between the approximate posterior and the generative model:

Figure 8.9: Explaining different sequences of actions in terms of anomaly signals. Online performance signal measured from the video latent state information. © 2024 IEEE.

$$G_\theta = \mathbb{E}_{q(o,s|\theta)}\left[\ln q(s|\theta) - \ln p(s|o,\theta)\right] - \mathbb{E}_{q(o|\theta)}\left[\ln p(o|\theta)\right] \tag{8.10}$$

Using the product rule of probability, EFE can be expressed as information seeking to account for observation preference instead of the policy parameter θ:

$$G_\theta \approx \mathbb{E}_{q(o,s|\theta)}\left[\ln q(s|\theta) - \ln q(s|o,\theta)\right] - \mathbb{E}_{q(o|\theta)}\left[\ln p(o|o_{pref})\right] \tag{8.11}$$

In eq. (8.11) the first term is the expected information gain conditioned on expected observation. For the intuitiveness of the expected information gain, eq. (8.9) can be rearranged as:

$$G_\theta = -\mathbb{E}_{q(o,s|\theta)}\left[\ln q(s|\theta) - \ln q(s|o,\theta)\right] - \mathbb{E}_{q(o|\theta)}\left[\ln p(o|o_{pref})\right] \tag{8.12}$$

EFE can be also expressed in terms of the KL divergence:

$$G_\theta = D_{KL}\left[q(o|\theta)||p(o|o_{pref})\right] + \mathbb{E}_{q(s|\theta)}\left[H\left[p(o|s)\right]\right] \tag{8.13}$$

8.4.3 Action prediction and selection

Action selection can be based on an action whose next action is not coherent with the next state, which tests the internal coherence of the learned models. This can be quantified using different anomaly measurement indicators in each level of the H-DBNs (see fig. 8.9). As described in algorithms 5 and 3, inference in continuous action is more complicated since we predict infinitely many combinations of acceleration and steering angle that are not yet effected to the actuators of a vehicle. One advantage we have in H-DBNs is that it is possible to infer $t+1, t+2, t+3, ..., t+n$, without taking the action. Since there is a computational gap one can not predict this way, so it should be restricted to the immediate (next) time prediction and measure the EFE based on learned model outcomes. This is done in our case using a particle filter which gives us a reasonable number of actions that minimizes the continuous action space. Figs. 8.5 (a) and 8.6 (a) indicate the expert agent action sequences, as a visual comparison with the predicted (fig. 8.5 (b)) and updated (fig. 8.6 (b)) action sequences using Alg. 5. Even if the agent predicted the action of expert agent well, that action may not be taken due to minor errors in the GSs. To clarify this we presented in fig. 8.8 (a) the expert agent actions and in fig. 8.8 (b) the number of predicted actions needed to finish the overtaking task in the online model for the learning agent. The number of actions that are predicted and tried are nearly seven times larger than the expert agent's actions.

8.5 Online Incremental Learning and Decision Making

Online incremental learning is a way of updating knowledge through the interaction of environments. It is a process of updating and expanding an AI model's capabilities over time. It mimics the way humans incrementally learn and accumulate knowledge throughout their lives, which is why sometimes it is called life-long learning (Pierre, 2018). It enables AI models to learn and update their capabilities without completely retraining them. This saves time and resources, as compared to traditional approaches that involve training separate models for each new requirement. It is essential to learn continually since the learned knowledge can not represent completely the environment. Even if some parts of it are represented, due to noisy sensory signals, the agent will eventually drift from the learned environmental models. Drifting from the predicted trajectory represents a stability issue of control commands that requires dynamic, non-equilibrium maneuvers (Weber and Gerdes, 2024).

Before applying an online incremental learning process, anomaly detection and characterization is the primary objective. To update any system, the new information should be first available. Checking the input information and deducing it as an increment (new knowledge),

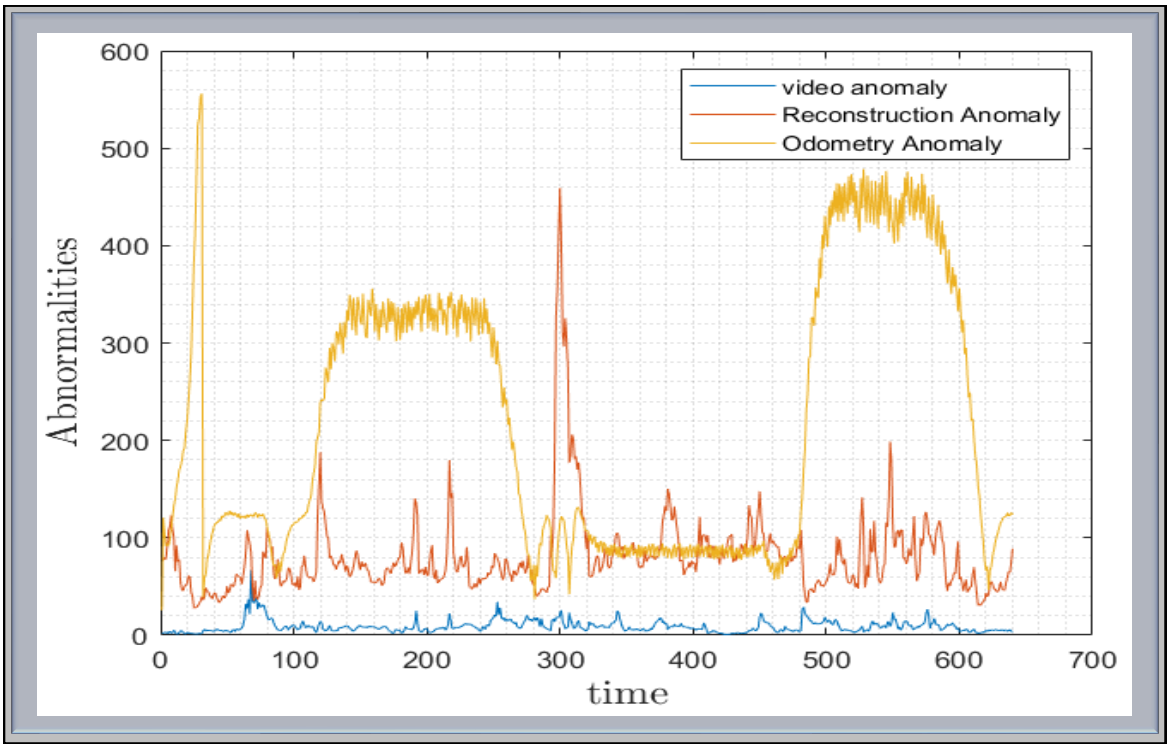


Figure 8.10: Anomaly signal detection. Combined video and odometry anomaly signals. Reconstruction anomaly is related to the video content and video anomaly is regarding the motion aspect of the video.

or system malfunction is the first step. For these reasons we test the developed system in learned and new environmental regions. As shown in fig. 8.10, when the agent is in new regions of the environment, the odometry anomaly is very high, instead, compared to the odometry anomaly, the video anomalies are small. The reason is that most of the road video data are similar with some variabilities. The anomalies in fig. 8.10 are part of the learned and the new regions starting from zone 1 to zone 3 that are indicated in fig. 8.13. To characterize the anomaly signals, we used the video modality as presented in fig. 8.11.

As a testing scenario in fig. 8.12, we show an experiment prototype where an agent passes through the learned regions and completely new regions. In this figure the black line indicates already learned trajectories while the red dashed line indicates a desired trajectory. This creates exploration zones as highlighted in fig. 8.13 allowing the agent to exploit known information and requiring it to explore new behavior in the exploration zones.

In our models we use dynamic deep VAEs which face catastrophic forgetting (Hong et al., 2022) when target distributions are different from the source distribution. To mitigate this issue in new environments, we update the batch normalization layers of the VAE by

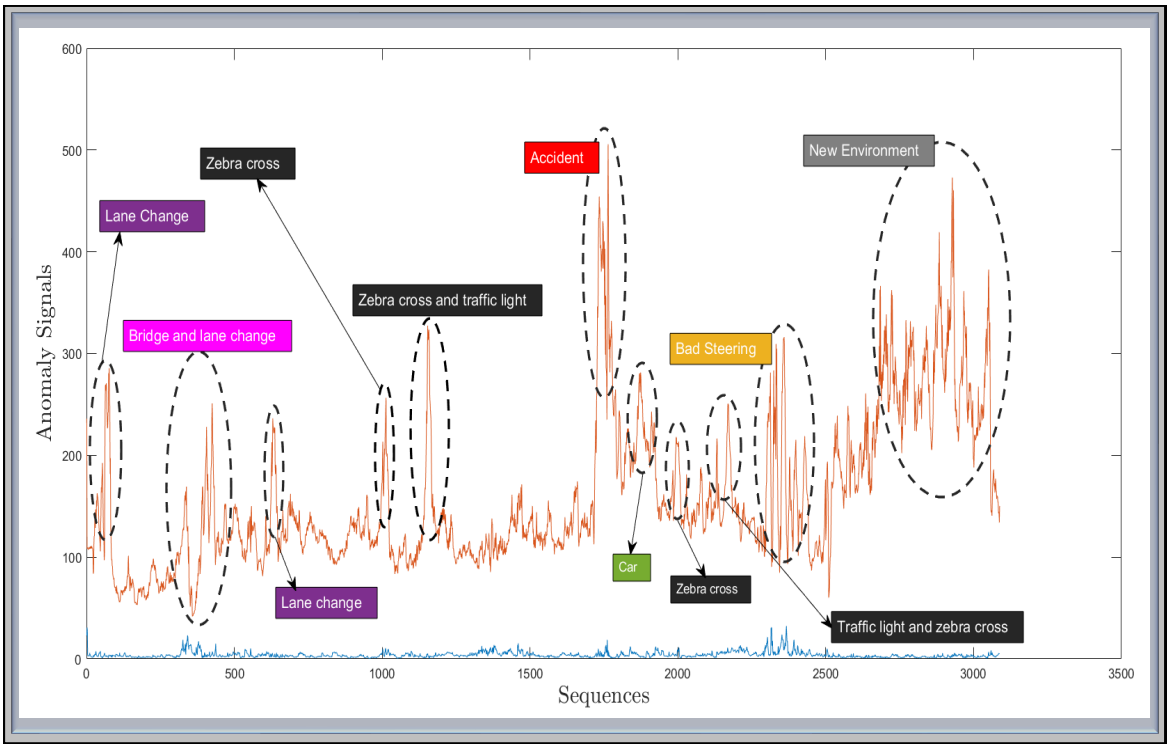


Figure 8.11: Anomaly signal characterization. Characterizing anomaly helps to know the threshold of new and learned environments. The orange line is the content related video encoding and the light blue signal is motion related anomaly signal of the video encoding.

modifying the feature representations of the learned models as a domain adaptation process. The domain adaptation process is performed based on Dynamic Unsupervised Adaptation (DUA) (Mirza et al., 2022) approach. DUA continuously updates the statistics of the batch normalization layers by modifying the feature representations of the model.

Batch normalization (Ioffe and Szegedy, 2015) calculates the mean and variance of each sample X from the training data and normalizes each incoming sample x as:

$$\hat{x} = \frac{x - \mathbb{E}[X]}{\sqrt{Var[X] + \varepsilon}} \cdot \gamma + \beta \tag{8.14}$$

where γ and β are the scale and shift parameters, and ε is used to account for division by zero. The expectation and variance are computed over the training data set as:

$$\begin{aligned} \hat{\mu}_t &= (1-\rho) \cdot \hat{\mu}_{t-1} + \rho \cdot \mu_t \\ \hat{\sigma}_t^2 &= (1-\rho) \cdot \hat{\sigma}_{t-1}^2 + \rho \cdot \sigma_t^2 \end{aligned} \tag{8.15}$$

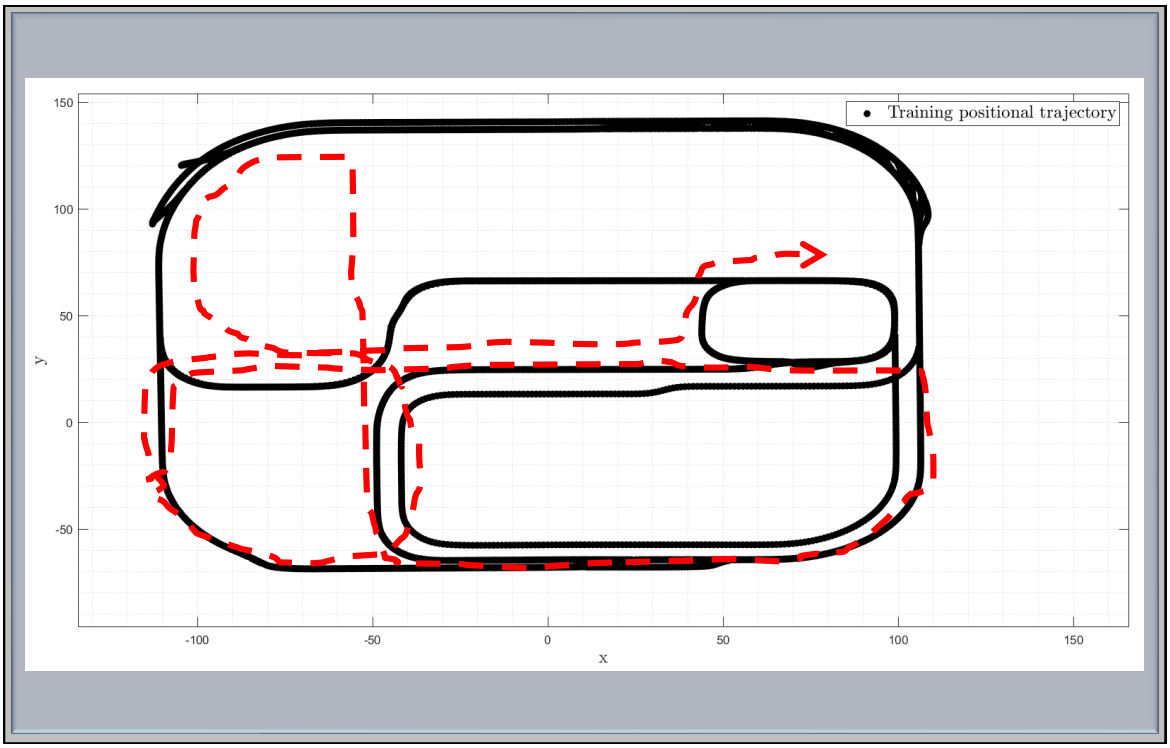


Figure 8.12: Training trajectories from Carla simulator. The red arrow shows the intended trajectory that the agent follows to both exploit and explore learned and new regions in the environment.

where $\hat{\mu}_t$ and $\hat{\sigma}_t^2$ are the estimated mean and variance from the training data, whereas μ and σ^2 represent the mean and variance of the current batch. The hyperparameter ρ is the momentum term, in most of NN architectures its default value is $\rho = 0.1$, while t denotes each training step.

In DUA (Mirza et al., 2022) excluding the running mean and variance every parameter of the source network are kept fixed. $\mathbb{E}[X]$ and $Var[X]$ are adapted based on eq. (8.14) to the new statistics X_{tar}, using the training statistics from X_{src}. Instead of using a constant value, adaptive momentum is followed.

The advantage of DUA is that it does not require to freeze some layers as in the common transfer learning (Bengio, 2011) approaches. Moreover, since it does not require additional training examples, it minimizes the long-tail data distribution problem (Jiang et al., 2022). Hence, it has a clear advantage for incremental online learning. Since the agent performs the updating process sequentially after each video frame, there is no substantial time gap between learning and decision making. Due to these advantages and a reasonable performance in the MMP anomaly signal output (see fig. 8.16), we adopted DUA in this chapter.

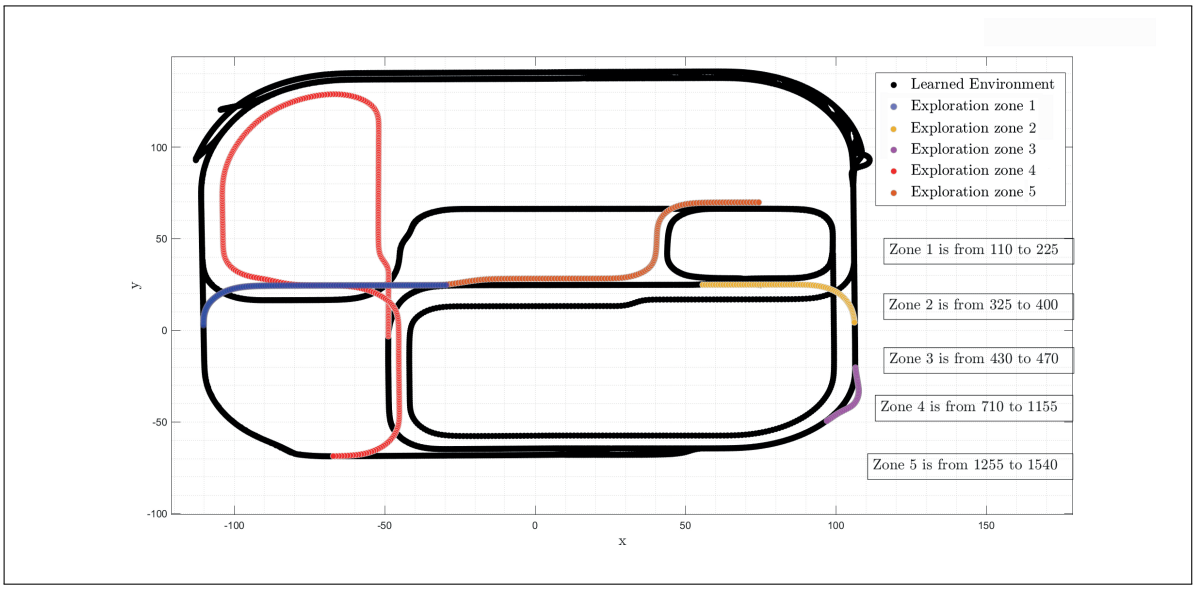


Figure 8.13: Experienced regions and possible exploration zones. These zones are based on a testing data containing positional sequences that are taken as time steps.

To update the entire developed model, we set up an experiment having five different exploration zones (see fig. 8.13). In these regions we suppose that there is no odometry observation. In this case, an agent needs to explore the new environment based on the video and action learned modalities. To predict the new odometry we use the learned vocabularies and combine each super state to compute the rotation angle between each cluster. Using cluster rotation angles an agent can predict future odometry state, given the current (the odometry state before transition to the new region) odometry state as:

$$\tilde{s}_{t,r} = \begin{pmatrix} \cos(\tilde{s}_{t,\theta}) & -\sin(\tilde{s}_{t,\theta}) \\ \sin(\tilde{s}_{t,\theta}) & \cos(\tilde{s}_{t,\theta}) \end{pmatrix} \tag{8.16}$$

Using eq. (8.16) as a transition model in new regions and the current state, the odometry state can be predicted using the MJPF.

8.6 Results

Characterizing each incident (anomaly) signal is crucial for continual learning. It is very important to know false positive signals, that might occur due to noisy measurements. For example in fig. 8.11 the bridge and lane change, zebra cross, overtaking a car in new regions are not new to the model. An accident due to bad steering (the vehicle flipped), and new regions in the environment are new signals to the model. Hence characterizing false positive

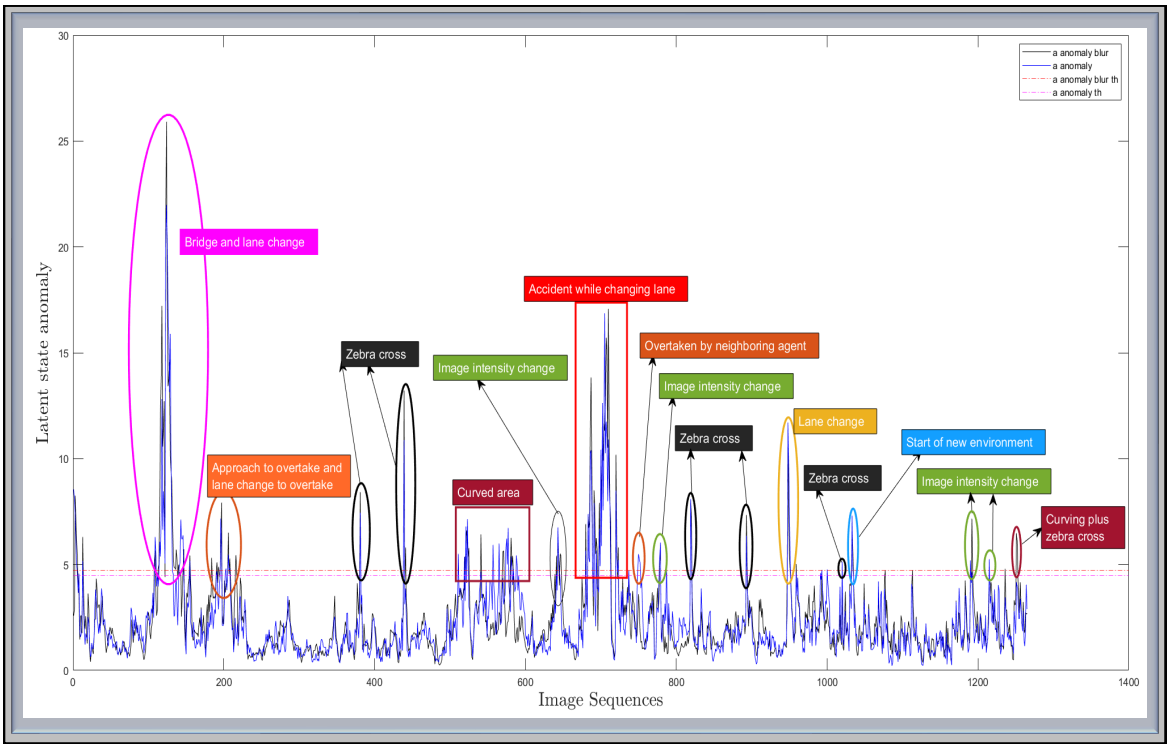


Figure 8.14: Anomaly signal thresholding. The threshold used here is by taking the mean of the anomaly signal and by scaling it with the standard error ($\mu + 2.5\sigma$). The blur experiment is to check how the threshold is affected if we minimize the luminosity of each video frame. The legends *anomaly blur th* and *anomaly th* are short forms for anomaly blur threshold and anomaly threshold respectively. These signals are computed for the content related latent states of the VAE encodings.

signals is mandatory to update the learned model. Different thresholding techniques can be used to characterize different anomaly signals.

The threshold can be set heuristically by considering different factors that may affect driving. As presented in fig. 8.14 we used mean of the anomalies scaled by the standard error: $th = \mu + 2.5\sigma$.

The online learning, inference, and decision making algorithm (Alg. 5), implementing online learning, inference, and decision making, is modified to account for updating (light training) the MMP model. After the latent states are obtained the algorithm checks for a threshold. If the threshold is very high the momentum updating schedule will be executed.

To evaluate the generative nature of the MMP model, we test the CARLA trained network with video frames of a vehicle in a UC3M corridor. In the training, we cautiously trained the network without the introduction of pedestrians. The MMP output is reasonable as shown in fig. 8.15.

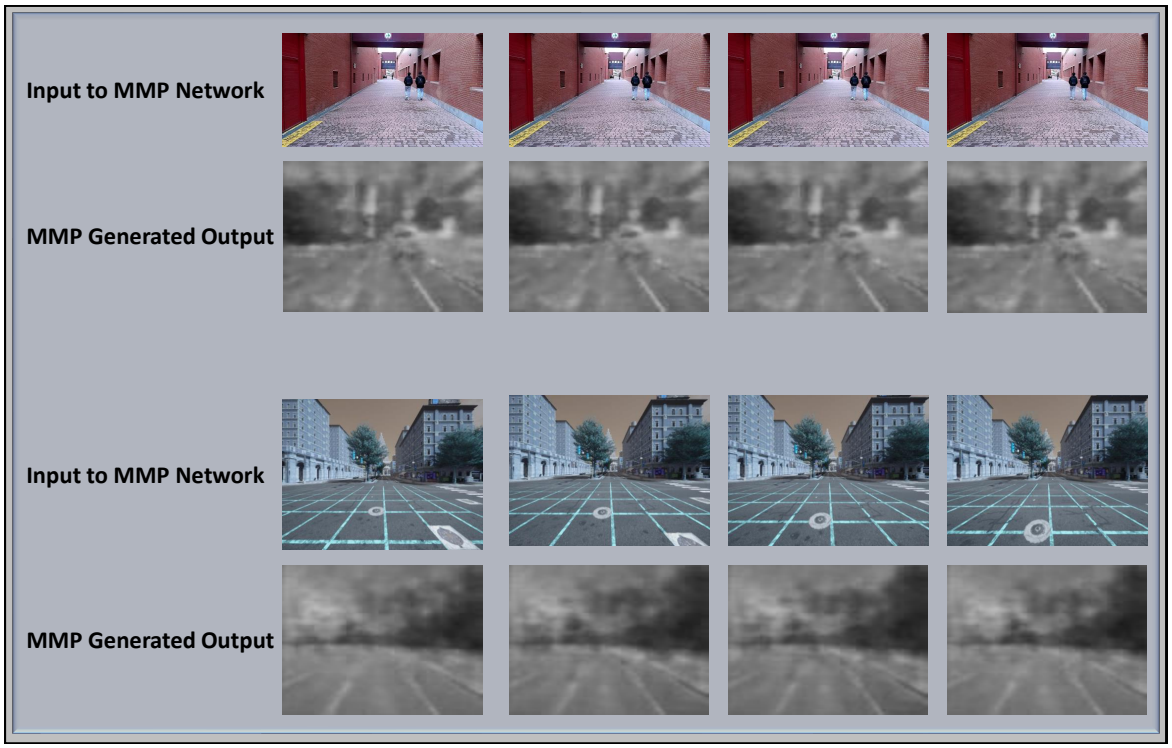


Figure 8.15: MMP generalizability. The input images are used to test the generalizability of the learned network.

The first input images in fig. 8.15, are from a real vehicle, taken in UC3M corridor. The second input images in this figure are taken from new regions of CARLA, which are not experienced by the model. The trained model does not have any of those type of inputs. We provide these testing images to check if the MMP is able to generalize well. If one sees closely the reconstructed images from the real image inputs, the model considers the two pedestrians as a vehicle. Based on these outputs, one can say the generalizability of the model is week. But considering the training experience, as the model has zero experience about those input image, being able to generate a plausible outputs, instead, validates a good characteristics of the learned model.

The continuous odometry anomaly signals are depicted in figs. 8.17 (a) and 8.17 (b). In fig. 8.17 (a), the signal is obtained by measuring the error between the predicted GS and the model evidence after observation, and fig. 8.17 (b) shows the anomaly signal between the predicted GS of odometry and the current super state. In fig. 8.17 (a), the highest rise is due to a combined effect of the new exploration region and the change of motion direction. Since the super states are not involved in this anomaly signal measurement, it only indicates how the GSs in the prediction and update phases are behaving. Hence, the measure

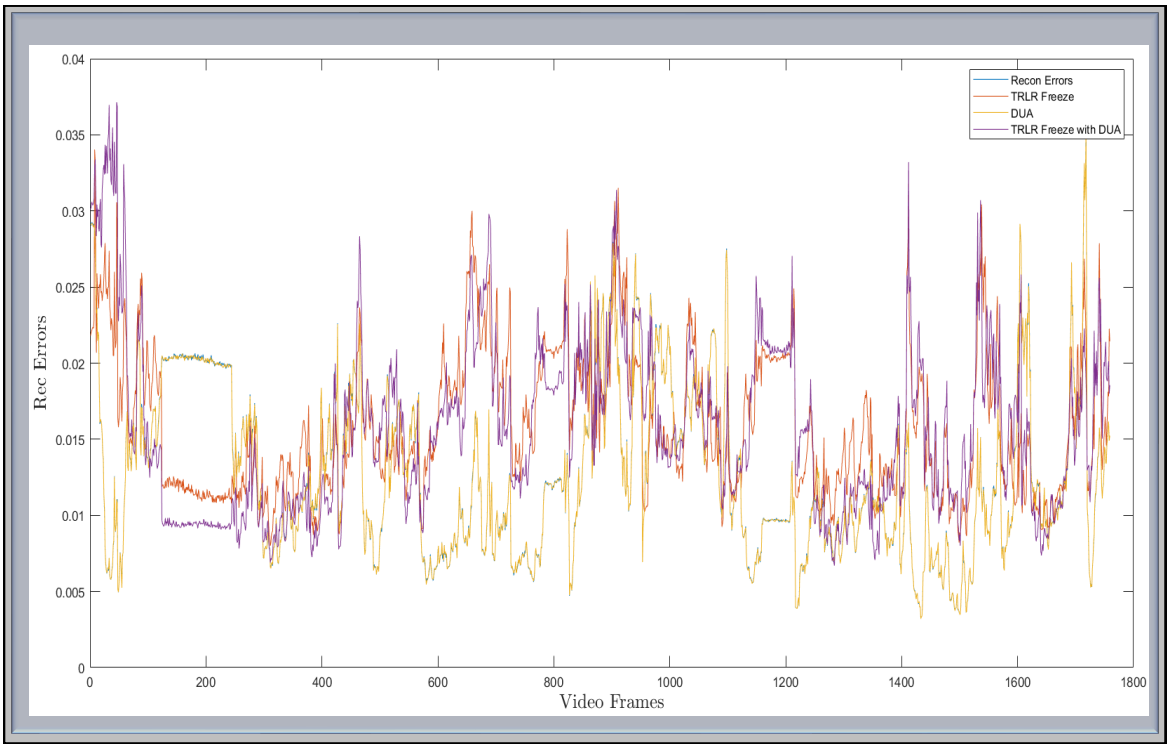
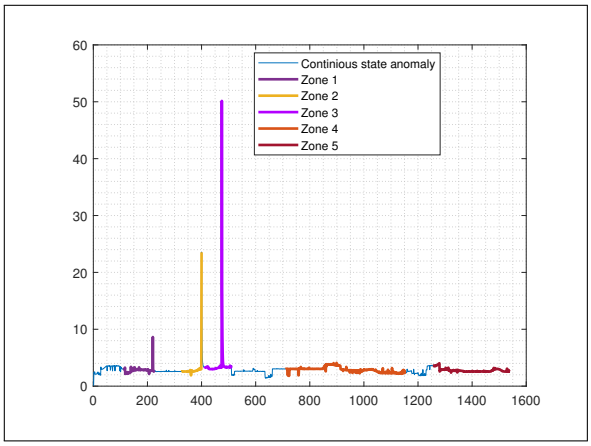
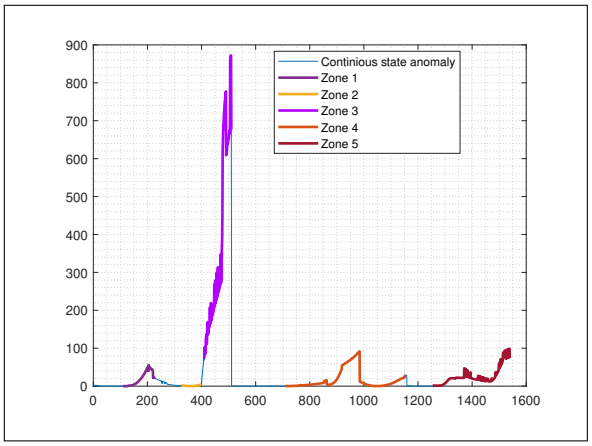


Figure 8.16: MMP incremental learning video reconstruction anomaly. The signals show how different transfer learning approaches affects the MMP output.



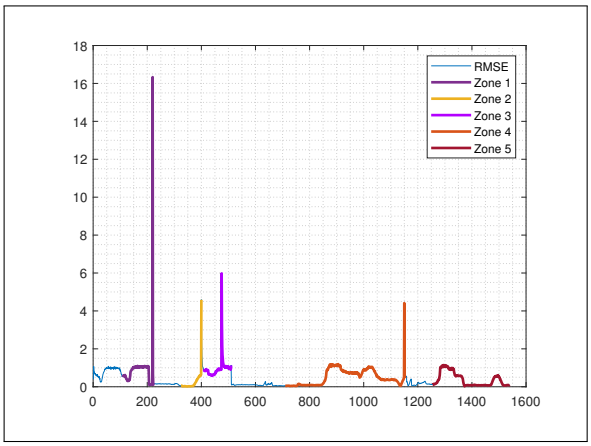
(a) predicted state vs observation.



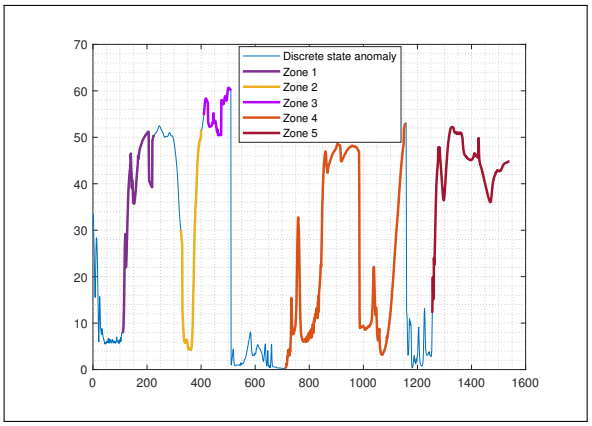
(b) predicted state vs current super state.

Figure 8.17: Odometry prediction anomaly in the continuous GSs. Exploration zones are shown to indicate how the anomaly signals are varying in each hierarchical generative states.

is a Bhattacharyya distance between the predicted and the observed states, including their corresponding covariance matrices. In fig. 8.17 (b), the anomaly signals are generally very high in new regions as those areas are unknown to the learned super states. Anomaly signals



(a) predicted vs updated states normalized RMSE.



(b) Discrete states (cluster prediction vs update) anomaly.

Figure 8.18: Filter performance anomaly in learned and explored zones. Exploration zones are shown to indicate how the anomaly signals are varying in each hierarchical generative states.

related to the overall performance measure of the filter and the discrete state level anomaly measures are shown in figs. 8.18 (a) and 8.18 (b), respectively. The anomalies are annotated based on fig. 8.13.

The inference performance of the MAC-MJPF is displayed in fig. 8.18 (a). It shows the normalized RMSE signal between the prediction and update stages of the filter. The discrete state anomaly signal (see fig. 8.18 (b)) shows higher values in the new regions. These peaks are expected as the anomaly signal indicator is designed to measure how the piece-wise linear vocabularies capture the high-level semantics in unexplored regions.

To have a better anomaly measure, we can combine both the continuous and discrete state anomaly signal measures.

8.7 Summary

We introduced online and incremental learning model of MASAA. We used a data-driven approach for the Coupled Bayesian learning and decision-making of interacting agents through multi-sensorial data. We demonstrated the online decision-making method in the learned regions of the CARLA simulator. As the agent is familiar with the environment, the next challenge is to maintain the agent on the learned regions. The reason why this is challenging is that simple distribution changes due to faulty actuation affect the decision making process. If the decision-making is altered, it creates new observations that are new to

the model, which affect the perception module. This creates unrecognizable inputs, that the agent is not able to address.

The natural process in this type of situation is to make the agent return to a well experienced region. The active inference framework follows human reasoning and decision-making processes that make it practically close to human cognition. The expected free energy (EFE) minimization approach tackles this problem by predicting many actions and evaluating them according to the current state of the agent and its intention (preferred observation) or goal. The action with the minimum EFE will be selected. Then the variational free energy (VFE) plays a role to predict the upcoming observations. These observations have to be measured according to the agents predicted state and the one which has lower VFE will be selected. This is how the perception action cycle, that we discussed in Chapter 4, is achieved. In doing so, this work discussed how the combined Markov Jump Particle Filters (MAC-MJPFs) were able to perform the perception action cycle in the active inference framework.

The challenge does not stop there. As new regions are explored, understanding them and taking new or near similar actions become another issue. As we are using neural networks, the source domain has issues in generalizing the new situations well. As demonstrated in fig. 8.15, if the target domain slightly shifts from the source domain, the VAE network struggles to generalize well. In such cases, domain adaptation through retraining or carefully modifying the network parameters can overcome the performance drop. If one chooses the retraining strategy, the testing sample must be adequate to accommodate the source domain as well. If the testing data has a shortage in representing the source domain, its performance in the training samples may drop as the network changes the learned parameters due to retraining with unbalanced testing data, which is termed catastrophic forgetting. If domain adaptation without retraining is chosen, the parameter weights must be transferred or adapted to the target domain. We adopted a zero-forgetting unsupervised domain adaptation strategy to modify the batch normalization layer in the online testing phase. This domain adaptation strategy is applied on the target domain and the results show that the network did not face catastrophic forgetting.

Chapter 9

Conclusion and Future Work

9.1 Conclusions

The increasing demand to create autonomy that is able to help mankind to transform the unfavorable environment into a simple, conducive and comfortable one has been always an inspiration for many discoveries and scientific breakthroughs. Controlled autonomy in areas like self-driving, cognitive radio, and health-related fields, are invaluable. Artificial intelligence is becoming an integral part of modern science. Enabling self-expressive and self-awareness capabilities in agents is a key objective of a general artificial intelligence. This thesis represents a step in this direction through the fusion of different sensors and models to enhance the self-awareness capabilities of an artificial agent.

Inspired by general artificial intelligence and controlled hallucination theory, in this thesis we introduced and developed a novel cognitive architecture, Multi-Agent Self-Awareness Architecture (MASAA), with all models that are learning hierarchy of representations, for hierarchical prediction and decision making tasks. Each component of MASAA is learned by fusing multi-sensory data in a Bayesian framework. It integrates generative world models abstracted in a lower-dimensional mental-space, and uses this abstracted knowledge to learn higher-dimensional knowledge through selective attention methodology. The key features in learning are adaptability and incident (anomaly) detection to make new decisions in new situations. The online incremental learning ability in the MASAA framework can naturally achieve this capability through utility functions that encode various possible outcomes. It can be focused or guided to preferences by maximizing expected utility functions.

In this thesis, in the first four chapters, we discussed frameworks that merge ideas from generative latent variable models representing a generative veridical world model to a generative approximated model. These latent variable models are representations of a physical systems in a state-space model. The generative state space models are robust

enough to describe the characteristics of a physical agent evolving in a dynamical state at any instant. These dynamic systems have an abstract phase state, i.e., generalized variables representing the coordinates in the state space. The time evolution of these systems are modeled by a dynamic rule that governs future values of all state vector variables, given current values of them. In these type of dynamic systems we reviewed the background studies on linear dynamical state space models, non-linear dynamical state space models, and their combinations. We further analyzed the representation and inference algorithms in these state space models.

To model a state-space from the fusion of multi-sensory data, a robust fusion algorithm is essential to reduce the computational complexity of a model. Dimensionality reduction and Clustering methods that were introduced in Chapter 2 and 3 are viable solutions in this regard. An unsupervised clustering algorithm is used to learn salient features in the lower-dimensional learning stage, and in the MASAA framework, it is called world model (WM) vocabulary learning. The Modified Growing Neural Gas unsupervised clustering algorithm is used to create the WM model. From the WM learned model, an attention vector is used to guide the higher-dimensional Multi-Modal Perception (MMP) model to grasp the environmental situation as a situation awareness. This situation model is transformed to a first-person model by the Active First-Person model. In the active first-person model the agent performs online maneuvering in the learned environment. This is a way of coupling multi-sensorial experiences in a controlled manner to execute a learned policy. It is well known that environments are not static, hence, changes in the environment must be adapted incrementally. The online and incremental learning module of MASAA is responsible for this mission. Before updating the learned models, these changes have to be first detected and characterized. Anomaly detection and characterization is, instead, the primary objective of the cost module. The cost module can also be extended to train WM and MMP models. These all process is stored in a unified autobiographic memory (AM), called the Short Term memory, that facilitates the communication between each developed models.

In this thesis starting from Chapter 6 to 8, we presented three papers that build novel interaction models. Chapter 5, instead, introduces the datasets that are experiences of expert agents. The representations, inferences, and decision making process are exerted to learn a structure or policy from them.

Chapter 6 describes collaborative interaction modeling between two self-driving vehicles, by introducing a relative distance-based interaction to localize both agents from video information. Multi-agent and multi-sensorial modalities are represented in the form of a Multi-agent Hierarchical Dynamic Bayesian Network (MAH-DBN). MAH-DBN is proven to be a robust representation probabilistic model. To make inferences of the future states

over the hierarchical representation of a MAH-DBN, a particular filter called Markov Jump Particle Filter (MJPF), which comprises a bank of Kalman Filter (KF) connected to a Particle Filter (PF). KF and PF are used to make inferences at the continuous and discrete levels of H-DBN models, respectively. The novelties in this chapter are:

(a) we describe a data-driven approach with a novel integration to a Bayesian inference model that localizes collaborative interacting agents only from video data. This gives an assertion of how a best coupling of sensory modalities in the learning stage can invoke memory representations without the observations of the other modality.

(b) the learned multi-sensorial modalities, i.e., odometry and video, are represented as a multi-agent hierarchical DBN (MAH-DBN). Since we are developing generative self-awareness systems, a novel way of representation using DBNs gives flexibility for anomaly detection and incremental learning.

(c) we validate the proposed system by developing a modified MJPF. We tested the system with complex real and simulated based experiments of vehicles performing an overtaking maneuver.

Interaction is a step that any learning algorithm has to encounter. A physical system has to interact with the world in which it resides. This world is mostly stochastic by nature as there are many other actors that are also residing (inhabiting) in this world. Chapter 6 is the triggering idea in establishing the MASAA architecture. Modeling interacting agents in a unified architecture is introduced and developed in Chapter 7. MASAA architecture can be assessed for its effectiveness in achieving scalability, adaptability, reliability and conceptual integrity in the online and incremental learning setup that we have tested. Accordingly, each characteristic attained in our scenario can be explained as follows:

1. *Scalable*. H-DBN based representations are generative, and their posterior distribution has an approximation that can be addressed using neural networks, which are scalable to higher-dimensional datasets. Scalability in architectural design also allows an agent to update its current understanding of the environment with new representations, that are also capable of modeling multi-ego heterogeneous agents.

2. *Adaptability.* An architecture is a blue-print to build a system. This blue-print should be independent of learning frameworks or developing methodologies. MASAA is adaptable to the most methodologies of self-awareness learning frameworks.

3. *Reliability.* Multi-sensory based systems should assert fail-safe scenarios. The guiding principles (business rules) of MASAA are flexible enough to assert fail-safe scenarios.

4. *Conceptual integrity.* MASAA is independent of its implementation algorithms. It has clear components that have input output relations that represents an overall idea of reasoning.

In this architecture, we build an interaction model for self-driving cars. Each model in MASAA is based on the integration of probabilistic self-expressive Bayesian models and Deep Learning approaches. We showed how lower-dimensional WM knowledge can guide the learning of higher-dimensional latent representations of interaction experiences. This experience further transformed into active first-person models for online inference of observations.

Finally in Chapter 8 we discussed online and incremental learning, and decision making, in the active inference framework, which updates all module of MASAA. We introduced six H-DBNs integrated as a single unified model for learning, inference, and decision making of an autonomous agent:

1. Video State Network
2. Odometry State Network
3. Action State Network
4. Coupled Video and Odometry Vocabulary Network
5. Action Vocabulary Network and
6. Configurator (Coupled Action, Video, and Odometry) Vocabulary Network

Among these networks, the Action State Network, the Action Vocabulary Network and the Configurator Network are the innovative contributions unified with the other networks, discussed in Chapter 7. The active network, which includes both the action state and action vocabulary nodes, is developed based on the Active Inference framework. We demonstrated the decision-making stage of MASAA in both learned and new environments. We used the CARLA simulator to test the integrated knowledge, allowing an agent to localize itself and navigate the environment by minimizing the variational free energy (VFE) and expected free energy (EFE), for inference and decision-making in a self-supervised approach.

9.2 Open questions and Future Work

Predicting odometry from video sequences in a new environment is still challenging, as the learned parameters are specific to the experienced environments. Transferring the parameter

weights without loss of information is difficult. The methodologies used in Chapter 8, related to unsupervised domain adaptation, should be explored widely to generalize well in new environments.

In a continuous action space, small variabilities can introduce major distribution shifts. These have a negative impact in the decision-making stage as they also shift the learned representations of odometry and video data. The active inference framework need to be explored better to effectively handle such complex real world scenarios.

Interaction modeling is a complex scenario which has to address number of data modalities and their dimensionality, interaction scenarios, type of Deep Learning network architecture, the hierarchy of the DBN and other parameter. Hence more research is needed in this direction to set up optimal parameters.

Additional anomaly signal measures should also be introduced to analyze each network hierarchy that are developed in Chapter 8. Action-based MMP and odometry-based MMP network relations need a thorough analysis. Inter and intra model coherence needs further research to reduce the complexity of the model.

More components might be also introduced specifically in the actuation subsystem of MASAA for drones. The hardware integration and analysis in real vehicles are also an open end for further research.

In addition, experience merging, model distillation, and architectural optimization that can modify or enhance MASAA can be a possible line of research.

In MASAA, heterogeneous agent interaction for disaster recovery scenarios where a drone and a self-driving vehicle interaction scenario can be integrated easily. In this type of scenario, a drone can recharge its battery after a flight mission, to be ready for the next mission. We model this scenario with a real DJI drone and iCab semi-autonomous vehicle (see fig. 5.7).

Landing experiments by a DJI drone were performed on a moving and static iCab agent. This interaction model can verify the heterogeneity of the framework that we discussed in Chapter 7 as a future work plan.

Bibliography

M. Abdou, H. Kamal, S. El-Tantawy, A. Abdelkhalek, O. Adel, K. Hamdy, and M. Abaas. End-to-end deep conditional imitation learning for autonomous driving. In *2019 31st International Conference on Microelectronics (ICM)*, pages 346–350, 2019. doi: 10.1109/ICM48031.2019.9021288.

A. S. Alemaw, G. Slavic, H. Iqbal, L. Marcenaro, D. M. Gomez, and C. Regazzoni. A data-driven approach for the localization of interacting agents via a multi-modal dynamic bayesian network framework. In *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–8, 2022. doi: 10.1109/AVSS56176.2022.9959648.

A. S. Alemaw, P. Zontone, L. Marcenaro, P. Marin, D. M. Gomez, and C. Regazzoni. Integrated learning and decision making for autonomous agents through energy based bayesian models. In *2024 27th International Conference on Information Fusion (FUSION)*, pages 1–8, 2024. doi: 10.23919/FUSION59988.2024.10706431.

A. S. Alemaw, G. Slavic, P. Zontone, L. Marcenaro, D. M. Gomez, and C. Regazzoni. Modeling interactions between autonomous agents in a multi-agent self-awareness architecture. *IEEE Transactions on Multimedia*, pages 1–16, 2025. doi: 10.1109/TMM.2025.3543110.

T. Aminosharieh Najafi, A. Affanni, R. Rinaldo, and P. Zontone. Driver attention assessment using physiological measures from eeg, ecg, and eda signals. *Sensors*, 23(4), 2023a. ISSN 1424-8220. doi: 10.3390/s23042039. URL https://www.mdpi.com/1424-8220/23/4/2039.

T. Aminosharieh Najafi, A. Affanni, R. Rinaldo, and P. Zontone. Drivers' mental engagement analysis using multi-sensor fusion approaches based on deep convolutional neural networks. *Sensors*, 23(17), 2023b. ISSN 1424-8220. doi: 10.3390/s23177346. URL https://www.mdpi.com/1424-8220/23/17/7346.

Andreas Geiger. Self-driving cars lecture notes University of Tubingen. https://uni-tuebingen.de/fakultaeten/mathematisch-naturwissenschaftliche-fakultaet/fachbereiche/informatik/lehrstuehle/autonomous-vision/lectures/self-driving-cars/, 2022. (Online; accessed 25-October-2024).

B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2008.10.024. URL https://www.sciencedirect.com/science/article/pii/S0921889008001772.

M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002. doi: 10.1109/78.978374.

M. Bain and C. Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, 1995. URL https://api.semanticscholar.org/CorpusID:10738655.

P. Baldi. Autoencoders, unsupervised learning and deep architectures. In *Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop -*

Volume 27, UTLW'11, page 37–50. JMLR.org, 2011.

M. Baydoun, D. Campo, V. Sanguineti, L. Marcenaro, A. Cavallaro, and C. Regazzoni. Learning switching models for abnormality detection for autonomous driving. In *2018 21st International Conference on Information Fusion (FUSION)*, pages 2606–2613, 2018. doi: 10.23919/ICIF.2018.8455592.

M. Baydoun, D. Campo, D. Kanapram, L. Marcenaro, and C. S. Regazzoni. Prediction of multi-target dynamics using discrete descriptors: an interactive approach. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3342–3346, 2019. doi: 10.1109/ICASSP.2019.8682272.

T. Bayes and n. Price. Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, f. r. s. communicated by mr. price, in a letter to john canton, a. m. f. r. s. *Philosophical Transactions of the Royal Society of London*, 53:370–418, 1763. doi: 10.1098/rstl.1763.0053. URL https://royalsocietypublishing.org/doi/abs/10.1098/rstl.1763.0053.

M. Beal, Z. Ghahramani, and C. Rasmussen. The infinite hidden markov model. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL https://proceedings.neurips.cc/paper_files/paper/2001/file/e3408432c1a48a52fb6c74d926b38886-Paper.pdf.

Y. Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop - Volume 27*, UTLW'11, page 17–37. JMLR.org, 2011.

P. Bernhard and M. Deschamps. Kalman 1960: The birth of modern system theory. *Mathematical Population Studies*, 26(3):123–145, 2019. doi: 10.1080/08898480.2018.1553393. URL https://inria.hal.science/hal-01940560.

Y. Bicer, A. Alizadeh, N. K. Ure, A. Erdogan, and O. Kizilirmak. Sample efficient interactive end-to-end deep learning for self-driving cars with selective multi-class safe dataset aggregation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2629–2634, 2019. doi: 10.1109/IROS40897.2019.8967948.

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, Apr. 2017. ISSN 1537-274X. doi: 10.1080/01621459.2017.1285773. URL http://dx.doi.org/10.1080/01621459.2017.1285773.

M. Buehler, K. Iagnemma, and S. Singh. *The 2005 DARPA Grand Challenge: The Great Robot Race*. Springer Publishing Company, Incorporated, 1st edition, 2007. ISBN 3540734287.

C. Cappelle, M. E. El Najjar, D. Pomorski, and F. Charpillet. Multi-sensors data fusion using dynamic bayesian network for robotised vehicle geo-localisation. In *2008 11th International Conference on Information Fusion*, pages 1–8, 2008.

L. E. Chai, S. K. Loh, S. T. Low, M. S. Mohamad, S. Deris, and Z. Zakaria. A review on the computational approaches for gene regulatory network construction. *Computers in Biology and Medicine*, 48:55–65, 2014. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2014.02.011. URL https://www.sciencedirect.com/science/article/pii/S0010482514000420.

V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), July 2009. ISSN 0360-0300. doi: 10.1145/1541880.1541882. URL https://doi.org/10.1145/1541880.1541882.

T. Chen, F. Faniyi, R. Bahsoon, P. R. Lewis, X. Yao, L. L. Minku, and L. Esterle. The handbook of engineering self-aware and self-expressive systems. *ArXiv*, abs/1409.1793, 2014. URL https://api.semanticscholar.org/CorpusID:14708381.

Z. Chen and X. Huang. End-to-end learning for lane keeping of self-driving cars. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1856–1860, 2017. doi: 10.1109/IVS.2017.7995975.

F. Codevilla, M. Miiller, A. López, V. Koltun, and A. Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, page 1–9. IEEE Press, 2018. doi: 10.1109/ICRA.2018.8460487. URL https://doi.org/10.1109/ICRA.2018.8460487.

F. Codevilla, E. Santana, A. Lopez, and A. Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9328–9337, 2019. doi: 10.1109/ICCV.2019.00942.

J. A. F. Costa and R. S. Oliveira. Cluster analysis using growing neural gas and graph partitioning. In *2007 International Joint Conference on Neural Networks*, pages 3051–3056, 2007. doi: 10.1109/IJCNN.2007.4371447.

L. Da Costa, T. Parr, N. Sajid, S. Veselic, V. Neacsu, and K. Friston. Active inference on discrete state-spaces: A synthesis. *Journal of Mathematical Psychology*, 99:102447, 2020. ISSN 0022-2496. doi: https://doi.org/10.1016/j.jmp.2020.102447. URL https://www.sciencedirect.com/science/article/pii/S0022249620300857.

A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

A. Doucet, S. J. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10:197–208, 2000. URL https://api.semanticscholar.org/CorpusID:16288401.

J. Elfring, E. Torta, and R. van de Molengraft. Particle filters: A hands-on tutorial. *Sensors*, 21(2), 2021. ISSN 1424-8220. doi: 10.3390/s21020438. URL https://www.mdpi.com/1424-8220/21/2/438.

J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, 2018. doi: 10.1109/TPAMI.2017.2658577.

L. Esterle, N. Dutt, C. Gruhl, P. R. Lewis, L. Marcenaro, C. Regazzoni, and A. Jantsch. Self-awareness in cyber-physical systems: Recent developments and open challenges. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6, 2023. doi: 10.23919/DATE56975.2023.10137197.

A. Ferdowsi, U. Challita, and W. Saad. Deep learning for reliable mobile edge analytics in intelligent transportation systems: An overview. *IEEE Vehicular Technology Magazine*, 14 (1):62–70, 2019. doi: 10.1109/MVT.2018.2883777.

R. Fernandez-Matellan, D. Puertas-Ramirez, D. M. Gomez, and J. G. Boticario. Fusion of physiological signals for modeling driver awareness levels in conditional autonomous vehicles using semi-supervised learning. In *2024 27th International Conference on Information Fusion (FUSION)*, pages 1–8, 2024. doi: 10.23919/FUSION59988.2024.10706517.

J. Forlizzi and C. DiSalvo. Service robots in the domestic environment: a study of the roomba vacuum in the home. HRI '06, page 258–265, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595932941. doi: 10.1145/1121241.1121286. URL https://doi.org/10.1145/1121241.1121286.

Z. Fountas, N. Sajid, P. A. Mediano, and K. Friston. Deep active inference agents using monte-carlo methods. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3604–3613, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

K. Friston, N. Trujillo-Barreto, and J. Daunizeau. Dem: A variational treatment of dynamic systems. *NeuroImage*, 41(3):849–885, 2008. ISSN 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2008.02.054. URL https://www.sciencedirect.com/science/article/pii/S1053811908001894.

K. J. Friston, F. Rigoli, D. Ognibene, C. D. Mathys, T. H. B. FitzGerald, and G. Pezzulo. Active inference and epistemic value. *Cognitive Neuroscience*, 6:187 – 214, 2015. URL https://api.semanticscholar.org/CorpusID:9129391.

K. J. Friston, T. H. B. FitzGerald, F. Rigoli, P. Schwartenbeck, and G. Pezzulo. Active inference: A process theory. *Neural Computation*, 29:1–49, 2017. URL https://api.semanticscholar.org/CorpusID:1976729.

B. Fritzke. A growing neural gas network learns topologies. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7. MIT Press, 1994. URL https://proceedings.neurips.cc/paper_files/paper/1994/file/d56b9fc4b0f1be8871f5e1c40c0067e7-Paper.pdf.

Q. Fu, H. Yu, X. Wang, Z. Yang, Y. He, H. Zhang, and A. Mian. Fast orb-slam without keypoint descriptors. *IEEE Transactions on Image Processing*, 31:1433–1446, 2022. doi: 10.1109/TIP.2021.3136710.

A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

A. F. Genovese. The interacting multiple model algorithm for accurate state estimation of maneuvering targets. *Johns Hopkins Apl Technical Digest*, 22:614–623, 2001.

L. Girin, S. Leglaive, X. Bie, J. Diard, T. Hueber, and X. Alameda-Pineda. Dynamical variational autoencoders: A comprehensive review. *Foundations and Trends® in Machine Learning*, 15(1–2):1–175, 2021. ISSN 1935-8245. doi: 10.1561/2200000089. URL http://dx.doi.org/10.1561/2200000089.

A. Gupta, A. S. Khwaja, A. Anpalagan, and L. Guan. Safe driving of autonomous vehicles through state representation learning. In *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pages 260–265, 2021. doi: 10.1109/IWCMC51323.2021.9498960.

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, 2006. doi: 10.1109/CVPR.2006.100.

P. Han, R. Mu, and N. Cui. Active and dynamic multi-sensor information fusion method based on dynamic bayesian networks. In *2009 International Conference on Mechatronics and Automation*, pages 3076–3080, 2009. doi: 10.1109/ICMA.2009.5246072.

T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

S. Haykin. *Cognitive Dynamic Systems: Perception-action Cycle, Radar and Radio*. Cognitive Dynamic Systems: Perception–action Cycle, Radar, and Radio. Cambridge Uni-

versity Press, 2012. ISBN 9780521114363. URL https://books.google.it/books?id=GMDdQEVm74UC.

I. Higgins, L. Matthey, A. Pal, C. P. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

G. E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders. In *Proceedings of the 21th International Conference on Artificial Neural Networks - Volume Part I*, ICANN'11, page 44–51, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 9783642217340.

Y. Hong, M. Mundt, S. Park, Y. Uh, and H. Byun. Return of the normal distribution: Flexible deep continual learning with variational auto-encoders. *Neural Networks*, 154: 397–412, 2022. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2022.07.016. URL https://www.sciencedirect.com/science/article/pii/S0893608022002702.

J. Hu, H. Kong, T. Liu, and Y. Meng. Autonomous motion decision-making based on deep reinforcement learning for autonomous driving. In *2022 6th CAA International Conference on Vehicular Control and Intelligence (CVCI)*, pages 1–6, 2022. doi: 10.1109/CVCI56766.2022.9964721.

S. Ioffe and C. Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.

H. Iqbal, D. Campo, M. Baydoun, L. Marcenaro, D. M. Gomez, and C. Regazzoni. Clustering optimization for abnormality detection in semi-autonomous systems. In *1st International Workshop on Multimodal Understanding and Learning for Embodied Applications*, MULEA '19, page 33–41, New York, NY, USA, 2019a. Association for Computing Machinery. ISBN 9781450369183. doi: 10.1145/3347450.3357657. URL https://doi.org/10.1145/3347450.3357657.

H. Iqbal, D. Campo, M. Baydoun, L. Marcenaro, D. M. Gomez, and C. Regazzoni. Clustering optimization for abnormality detection in semi-autonomous systems. In *1st International Workshop on Multimodal Understanding and Learning for Embodied Applications*, MULEA '19, page 33–41, New York, NY, USA, 2019b. Association for Computing Machinery. ISBN 9781450369183. doi: 10.1145/3347450.3357657. URL https://doi.org/10.1145/3347450.3357657.

H. Iqbal, D. Campo, L. Marcenaro, D. Martin Gomez, and C. Regazzoni. Data-driven transition matrix estimation in probabilistic learning models for autonomous driving. *Signal Processing*, 188:108170, 2021. ISSN 0165-1684. doi: https://doi.org/10.1016/j.sigpro.2021.108170. URL https://www.sciencedirect.com/science/article/pii/S0165168421002085.

J. Janai, F. Güney, A. Behl, and A. Geiger. Computer vision for autonomous vehicles: Problems, datasets and state of the art, 2021. URL https://arxiv.org/abs/1704.05519.

A. H. Jazwinski. Stochastic processes and filtering theory. 1970. URL https://api.semanticscholar.org/CorpusID:122340800.

Z. Jiang, M. Liu, Z. Guo, S. Zhang, Y. Lin, and D. Pan. A tale of eda's long tail: Long-tailed distribution learning for electronic design automation. In *Proceedings of the 2022 ACM/IEEE Workshop on Machine Learning for CAD*, MLCAD '22, page 135–141, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450394864. doi: 10.1145/3551901.3556485. URL https://doi.org/10.1145/3551901.3556485.

S. Julier and J. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004. doi: 10.1109/JPROC.2003.823141.

R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960. ISSN 0021-9223. doi: 10.1115/1.3662552. URL https://doi.org/10.1115/1.3662552.

D. T. Kanapram, M. Marchese, E. L. Bodanese, D. M. Gomez, L. Marcenaro, and C. Regazzoni. Dynamic bayesian collective awareness models for a network of ego-things. *IEEE Internet of Things Journal*, 8(5):3224–3241, 2021. doi: 10.1109/JIOT.2020.3043199.

M. Kaneko, K. Iwami, T. Ogawa, T. Yamasaki, and K. Aizawa. Mask-slam: Robust feature-based monocular slam by masking using semantic segmentation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 371–3718, 2018. doi: 10.1109/CVPRW.2018.00063.

O. Kathe, V. Turkar, A. Jagtap, and G. Gidaye. Maze solving robot using image processing. In *2015 IEEE Bombay Section Symposium (IBSS)*, pages 1–5, 2015. doi: 10.1109/IBSS.2015.7456635.

J. Kephart and D. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003. doi: 10.1109/MC.2003.1160055.

J. Kim, S. Moon, A. Rohrbach, T. Darrell, and J. Canny. Advisable learning for self-driving vehicles by internalizing observation-to-action rules. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9658–9667, 2020. doi: 10.1109/CVPR42600.2020.00968.

S. Kim. Autonomous cleaning robot: Roboking system integration and overview. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 5, pages 4437–4441 Vol.5, 2004. doi: 10.1109/ROBOT.2004.1302416.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013. URL https://api.semanticscholar.org/CorpusID:216078090.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

T. Kohonen. The self-organizing map. *Neurocomputing*, 21(1):1–6, 1998. ISSN 0925-2312. doi: https://doi.org/10.1016/S0925-2312(98)00030-7. URL https://www.sciencedirect.com/science/article/pii/S0925231298000307.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009. ISBN 0262013193.

A. Krayani, M. Baydoun, L. Marcenaro, A. S. Alam, and C. Regazzoni. Self-learning bayesian generative models for jammer detection in cognitive-uav-radios. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pages 1–7, 2020. doi: 10.1109/GLOBECOM42002.2020.9322583.

A. Krayani, K. Khan, L. Marcenaro, M. Marchese, and C. Regazzoni. Self-supervised path planning in uav-aided wireless networks based on active inference. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 13181–13185, 2024. doi: 10.1109/ICASSP48485.2024.10446575.

D. Lai, Y. Zhang, and C. Li. A survey of deep learning application in dynamic visual slam. In *2020 International Conference on Big Data Artificial Intelligence Software Engineering (ICBASE)*, pages 279–283, 2020. doi: 10.1109/ICBASE51474.2020.00065.

Y. LeCun. *A Path Towards Autonomous Machine Intelligence.* OpenReview Archive, 2022.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

S. Lefèvre, D. Vasquez, and C. Laugier. A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH Journal*, 1, 2014. URL https://api.semanticscholar.org/CorpusID:16655281.

P. R. Lewis, A. Chandra, F. Faniyi, K. Glette, T. Chen, R. Bahsoon, J. Torresen, and X. Yao. Architectural aspects of self-aware and self-expressive computing systems: From psychology to engineering. *Computer*, 48(8):62–70, 2015. doi: 10.1109/MC.2015.235.

R. Li, S. Wang, Z. Long, and D. Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7286–7291, 2018. doi: 10.1109/ICRA.2018.8461251.

S. Li, F. Xue, X. Wang, Z. Yan, and H. Zha. Sequential adversarial learning for self-supervised deep visual odometry. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2851–2860, 2019. doi: 10.1109/ICCV.2019.00294.

S. Li, D. Zhang, Y. Xian, B. Li, T. Zhang, and C. Zhong. Overview of deep learning application on visual slam. *Displays*, 74:102298, 2022. ISSN 0141-9382. doi: https://doi.org/10.1016/j.displa.2022.102298. URL https://www.sciencedirect.com/science/article/pii/S0141938222001160.

Y. Li and R. E. Turner. Rényi divergence variational inference. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/7750ca3559e5b8e1f44210283368fc16-Paper.pdf.

Z. Li. A hierarchical autonomous driving framework combining reinforcement learning and imitation learning. In *2021 International Conference on Computer Engineering and Application (ICCEA)*, pages 395–400, 2021. doi: 10.1109/ICCEA53728.2021.00084.

Q. Liu, X. Li, S. Yuan, and Z. Li. Decision-making technology for autonomous vehicles: Learning-based methods, applications and future outlook. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 30–37, 2021. doi: 10.1109/ITSC48978.2021.9564580.

P. Marín-Plaza, J. Beltrán, A. Hussein, B. Musleh, D. Martín, A. de la Escalera, and J. M. Armingol. Stereo vision-based local occupancy grid map for autonomous navigation in ros. In *International Conference on Computer Vision Theory and Applications*, volume 4, pages 701–706. SciTePress, 2016.

P. Mazzaglia, T. Verbelen, and B. Dhoedt. Contrastive active inference. *ArXiv*, abs/2110.10083, 2021. URL https://api.semanticscholar.org/CorpusID:239024470.

D. McFarlane, V. Giannikas, and W. Lu. Intelligent logistics: Involving the customer. *Computers in Industry*, 81:105–115, 2016. ISSN 0166-3615. doi: https://doi.org/10.1016/j.compind.2015.10.002. URL https://www.sciencedirect.com/science/article/pii/S0166361515300488. Emerging ICT concepts for smart, safe and sustainable industrial systems.

M. J. Mirza, J. Micorek, H. Possegger, and H. Bischof. The norm must go on: Dynamic unsupervised domain adaptation by normalization. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14745–14755, 2022. doi: 10.1109/CVPR52688.2022.01435.

M. Mitchell. Self-awareness and control in decentralized systems. In *Metacognition in Computation*, 2005. URL https://api.semanticscholar.org/CorpusID:17241174.

J. Mo, M. J. Islam, and J. Sattar. Fast direct stereo visual slam. *IEEE Robotics and Automation Letters*, 7(2):778–785, 2022. doi: 10.1109/LRA.2021.3133860.

R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. doi:

10.1109/TRO.2015.2463671.

K. P. Murphy. *Dynamic bayesian networks: representation, inference and learning*. University of California, Berkeley, 2002.

K. P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. MIT Press, 2023. URL http://probml.github.io/book2.

R. Novianto and M.-A. Williams. The role of attention in robot self-awareness. In *RO-MAN 2009 - The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pages 1047–1053, 2009. doi: 10.1109/ROMAN.2009.5326155.

S. Nozari, A. Krayani, P. Marin-Plaza, L. Marcenaro, D. M. Gómez, and C. Regazzoni. Active inference integrated with imitation learning for autonomous driving. *IEEE Access*, 10:49738–49756, 2022. doi: 10.1109/ACCESS.2022.3172712.

On-Road Automated Driving (ORAD) Committee. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, sep 2016. URL https://doi.org/10.4271/J3016_201609.

T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters. An algorithmic perspective on imitation learning. *ArXiv*, abs/1811.06711, 2018. URL https://api.semanticscholar.org/CorpusID:53670210.

K. Ozasa, Y. Toda, and T. Matsuno. Growing neural gas based traversability clustering for an autonomous robot. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, 2023. doi: 10.1109/IJCNN54540.2023.10191416.

A. Pentland and T. Choudhury. Face recognition for smart environments. *Computer*, 33(2): 50–55, 2000. doi: 10.1109/2.820039.

J. M. Pierre. Incremental lifelong deep learning for autonomous vehicles. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3949–3954, 2018. doi: 10.1109/ITSC.2018.8569992.

D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988. URL https://proceedings.neurips.cc/paper_files/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf.

M. Prabhushankar and G. AlRegib. Stochastic surprisal: An inferential measurement of free energy in neural networks. *Frontiers in Neuroscience*, 17, 2023. ISSN 1662-453X. doi: 10.3389/fnins.2023.926418. URL https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2023.926418.

A. Prakash, A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger. Exploring data aggregation in policy learning for vision-based urban autonomous driving. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11760–11770, 2020. doi: 10.1109/CVPR42600.2020.01178.

M. Ravanbakhsh, M. Baydoun, D. Campo, P. Marin, D. Martin, L. Marcenaro, and C. Regazzoni. Learning self-awareness for autonomous vehicles: Exploring multisensory incremental models. *IEEE Transactions on Intelligent Transportation Systems*, 22(6):3372–3386, 2021. doi: 10.1109/TITS.2020.2984735.

C. Regazzoni and I. Pitas. Perspectives in autonomous systems research [in the spotlight]. *IEEE Signal Processing Magazine*, 36(5):148–147, 2019. doi: 10.1109/MSP.2019.2925436.

C. S. Regazzoni, L. Marcenaro, D. Campo, and B. Rinner. Multisensorial generative and descriptive self-awareness models for autonomous systems. *Proceedings of the IEEE*, 108 (7):987–1010, 2020. doi: 10.1109/JPROC.2020.2986602.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page II–1278–II–1286. JMLR.org, 2014.

D. Ridel, E. Rehder, M. Lauer, C. Stiller, and D. Wolf. A literature review on the prediction of pedestrian behavior in urban scenarios. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3105–3112, 2018. doi: 10.1109/ITSC.2018.8569415.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning internal representations by error propagation*, page 318–362. MIT Press, Cambridge, MA, USA, 1986. ISBN 026268053X.

J. Schlatow, M. Moostl, R. Ernst, M. Nolte, I. Jatzkowski, M. Maurer, C. Herber, and A. Herkersdorf. Self-awareness in autonomous automotive systems. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, pages 1050–1055, 2017. doi: 10.23919/DATE.2017.7927145.

J. Schulz, C. Hubmann, J. Löchner, and D. Burschka. Interaction-aware probabilistic behavior prediction in urban environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3999–4006, 2018. doi: 10.1109/IROS.2018.8594095.

A. Seth. *Being You: A New Science of Consciousness*. Dutton, 2021.

A. Signa, A. Chella, and M. Gentile. Cognitive robots and the conscious mind: A review of the global workspace theory. *Current Robotics Reports*, 2021. doi: 10.1007/s43154-021-00044-7.

R. R. Sims. Kolb's experiential learning theory: A framework for assessing person-job interaction. *Academy of Management Review*, 8:501–508, 1983. URL https://api.semanticscholar.org/CorpusID:145543746.

Sivic and Zisserman. Video google: a text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1470–1477 vol.2, 2003. doi: 10.1109/ICCV.2003.1238663.

G. Slavic, A. S. Alemaw, L. Marcenaro, and C. Regazzoni. Learning of linear video prediction models in a multi-modal framework for anomaly detection. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 1569–1573, 2021. doi: 10.1109/ICIP42928.2021.9506049.

G. Slavic, M. Baydoun, D. Campo, L. Marcenaro, and C. Regazzoni. Multilevel anomaly detection through variational autoencoders and bayesian models for self-aware embodied agents. *IEEE Transactions on Multimedia*, 24:1399–1414, 2022. doi: 10.1109/TMM.2021.3065232.

G. Slavic, A. S. Alemaw, L. Marcenaro, D. Martín Gómez, and C. Regazzoni. A kalman variational autoencoder model assisted by odometric clustering for video frame prediction and anomaly detection. *IEEE Transactions on Image Processing*, 32:415–429, 2023. doi: 10.1109/TIP.2022.3229620.

R. Smith, K. J. Friston, and C. J. Whyte. A step-by-step tutorial on active inference and its application to empirical data. *Journal of Mathematical Psychology*, 107:102632, 2022. ISSN 0022-2496. doi: https://doi.org/10.1016/j.jmp.2021.102632. URL https://www.sciencedirect.com/science/article/pii/S0022249621000973.

K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/8d55a249e6baa5c06772297520da2051-Paper.pdf.

R. S. Sutton and A. G. Barto. Reinforcement learning: an introduction, 2nd edn. adaptive computation and machine learning, 2018.

D. H. Terman and E. M. Izhikevich. State space. *Scholarpedia*, 3(3):1924, 2008. doi: 10.4249/scholarpedia.1924. revision #137545.

The Evening News. Radio driven car hits sedan but keeps going. *The Evening News*, July 28 1925. https://www.newspapers.com/article/the-evening-news-radio-driven-car-hits-s/147714843/ (Online; accessed October 25th, 2024).

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, Dec. 2010. ISSN 1532-4435.

R. Wallace, A. T. Stentz, C. Thorpe, H. Moravec, W. R. L. Whittaker, and T. Kanade. First results in robot road-following. In *Proceedings of 9th International Joint Conference on Artificial Intelligence (IJCAI '85)*, volume 2, pages 1089 – 1095, August 1985.

E. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, 2000. doi: 10.1109/ASSPCC.2000.882463.

T. P. Weber and J. C. Gerdes. Modeling and control for dynamic drifting trajectories. *IEEE Transactions on Intelligent Vehicles*, 9(2):3731–3741, 2024. doi: 10.1109/TIV.2023.3340918.

G. Welch and G. Bishop. Welch & bishop , an introduction to the kalman filter 2 1 the discrete kalman filter in 1960. In *International Conference on Computer Graphics and Interactive Techniques*, 1994. URL https://api.semanticscholar.org/CorpusID:14848980.

Wikipedia contributors. History of self-driving cars — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=History_of_self-driving_cars&oldid=1252760289, 2024. (Online; accessed 25-October-2024).

S. Xiang, F. Nie, and C. Zhang. Learning a mahalanobis distance metric for data clustering and classification. *Pattern Recognition*, 41(12):3600–3612, 2008. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2008.05.018. URL https://www.sciencedirect.com/science/article/pii/S0031320308002057.

Y. Xiao, F. Codevilla, C. Pal, and A. M. Lopez. Action-based representation learning for autonomous driving, 2020. URL https://arxiv.org/abs/2008.09417.

W. Zhan, L. Sun, Y. Hu, J. Li, and M. Tomizuka. Towards a fatality-aware benchmark of probabilistic reaction prediction in highly interactive driving scenarios. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3274–3280, 2018. doi: 10.1109/ITSC.2018.8569785.

Y. Zhang, Y. Chen, J. Wang, and Z. Pan. Unsupervised deep anomaly detection for multi-sensor time-series signals. *IEEE Transactions on Knowledge and Data Engineering*, 35(2):2118–2132, 2023. doi: 10.1109/TKDE.2021.3102110.

P. Zontone, A. Affanni, A. Piras, and R. Rinaldo. Stress recognition in a simulated city environment using skin potential response (spr) signals. In *2021 IEEE International Workshop on Metrology for Automotive (MetroAutomotive)*, pages 135–140, 2021. doi: 10.1109/MetroAutomotive50197.2021.9502867.