

Models from Models: Modeling of Complex Systems by Combining Model-Based and Data-Driven Learning

Ashraf Sharifi

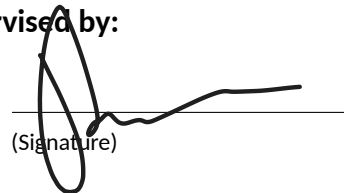
Department of Computer Science
University of Verona

PhD Cycle:

XXXVIII Cycle (38th Cycle)

Supervised by:

Prof. Davide Quaglia

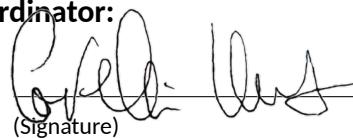

(Signature)

Prof. Sara Migliorini


(Signature)

PhD Coordinator:

Prof. Umberto Castellani


(Signature)

ISBN: 978-88-6925-184-9

August 11, 2026

PhD Thesis by Ashraf Sharifi is licensed under **CC BY-NC-ND 4.0**

For a copy of the license, please visit:

<https://creativecommons.org/licenses/by-nc-nd/4.0/>

In loving memory of my father, whom I miss beyond words.

To my mother, for her endless care and love.

To my husband, whose presence turned every struggle into a shared journey.

And, to the cherished lives sacrificed for the freedom of my Iran, during the days of silence.

Declaration

I declare that this thesis is entirely my own work and, except where stated, describes my own research.

Ashraf Sharifi
University of Verona

Acknowledgements

I would like to express my sincere gratitude to **Prof. Davide Quaglia**, my supervisor, for his constant guidance, generous support, and, above all, for the trust he placed in me throughout this research.

I would also like to express my sincere appreciation to **Prof. Sara Migliorini** for her valuable supervision, constructive feedback, insightful comments, and the many helpful discussions that greatly enriched this research.

Abstract

Developing a reliable Digital Twin (DT) for complex systems poses unique challenges due to their nonlinear dynamics, environmental sensitivity, and limited observability. This thesis addresses these challenges by designing an integrated DT framework that unifies data-driven Machine Learning (ML) with model-driven biological and physical principles. The objective is to create DTs that are not only accurate but also interpretable, robust, and applicable across a wide range of complex systems, with agriculture and aquaculture used as representative case studies in this thesis to demonstrate their practical utility. These domains exemplify the challenges inherent to complex systems, nonlinear interactions, environmental variability, and partial observability, which make them ideal testbeds for evaluating the effectiveness and generality of the proposed DT framework.

The first contribution focuses on controlled-environment agriculture, where a virtual-sensing architecture has been developed to reconstruct greenhouse microclimates in areas without installed physical sensors. By incorporating knowledge of the greenhouse structure into the learning models, this system improves spatial generalization and reduces reliance on extensive sensor networks. Additionally, an autonomous rover employs Deep Reinforcement Learning (DRL) to optimize its sampling strategy, demonstrating how intelligent data acquisition can further enhance the reliability of the DT. Together, these contributions establish the environmental perception and data collection layers of the DT.

The framework is then validated through two biological case studies that exemplify the integration of data-driven and model-driven methods. In aquaculture, a hybrid bioenergetic-Deep Learning (DL) model captures Rainbow Trout growth by combining mechanistic metabolic descriptors with multimodal sensor data. In plant phenotyping, a DT for *Salicornia* development integrates nonlinear growth models with image-based structural traits to derive biologically meaningful indicators such as growth velocity, acceleration, and developmental transitions. A special low-cost *Salicornia* image dataset is developed to support this process, enabling the system to translate raw imagery into reliable biomass estimates.

Across all domains, the thesis demonstrates that integrating mechanistic insights with ML yields DTs that are more robust, data-efficient, and interpretable than those from purely black-box approaches. This framework offers a generalizable method for developing biologically aligned DTs that can aid in decision-making for greenhouses, aquaponics, and other complex biological systems.

List of publications

The following is a list of articles and presentations that were published as a result of the research carried out for this thesis. Each publication is accompanied by a reference to the chapter(s) or section(s) in which its corresponding methods and results are discussed.

Articles

1. A. Sharifi, S. Migliorini, and D. Quaglia, "Optimizing the Trajectory of Agricultural Robots in Greenhouse Climatic Sensing with Deep Reinforcement Learning," 2024 International Conference on Control, Automation and Diagnosis (ICCAD), Paris, France, 2024, pp. 1-6. doi:10.1109/ICCAD60883.2024.10553772.
 - Discussed in **Section 4.3.3**, **Section 2.3.3**, and preliminary results in **Section 4.4.1**.
2. A. Sharifi, S. Migliorini, and D. Quaglia, "Combining Convolutional and Recurrent Neural Networks to Improve Greenhouse Microclimate Mapping," 2024 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor), IEEE, 2024.
 - Discussed in **Section 3.3.1.2** and preliminary results in **Section 3.5.1**.
3. A. Sharifi, S. Migliorini, and D. Quaglia, "Optimizing Trajectories for Rechargeable Agricultural Robots in Greenhouse Climatic Sensing Using Deep Reinforcement Learning with Proximal Policy Optimization Algorithm," Future Internet, 2025, 17, 296. <https://doi.org/10.3390/fi17070296>
 - Discussed in **Section 4.3.4**, **Section 2.3.4**, and preliminary results in **Section 4.4.2**.
4. E. Brentarolli, A. Sharifi, M. Ruaro, and S. Migliorini, "The Power of TimeLLM for Greenhouse Microclimate Mapping," 2025 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor), IEEE, 2025.
 - Discussed in **Section 3.3.1.3** and preliminary results in **Section 3.5.2**.
5. A. Sharifi, S. Migliorini, D. Quaglia, and R. Pastres "Merging Deep Learning and Mechanistic Approaches in the Biological Domain with Application to Rainbow Trout Weight Prediction," Under review in Expert Systems with Applications, 2026.
 - Discussed in **Chapter 5**.
6. A. Sharifi, M. T. Hokmabadi, S. Migliorini, D. Quaglia, and R. Pastres "Dataset on Longitudinal Image-Based Phenotyping and Water Quality Measurements for Salicornia Growth in a Marine Aquaponic System" Submitted to Data in Brief, 2026.
 - Discussed in **Section 6.4**.
7. A. Sharifi, M. T. Hokmabadi, S. Migliorini, D. Quaglia, and R. Pastres, "Multimodal Aquaponic Dataset for Salicornia: RGB Imaging, Morphological traits, Biomass, and Water-Quality Telemetry" Zenodo Repository, 2026. <https://doi.org/10.5281/zenodo.21298089>
 - Dataset associated with **Section 6.4** and supporting the Data in Brief submission.

8. A. Sharifi, S. Migliorini, D. Quaglia, and R. Pastres "A Digital Twin Framework for Modeling *Salicornia Growth*" To be submitted to *Expert Systems with Applications*, 2026.

- Discussed in **Sections 6.4 and 6.5**.

Presentations

1. M. T. Hokmabadi, A. Sharifi, D. Scutelnic, E. Fasani, D. Quaglia, and C. Daffara, "*Comparison of Optical Techniques for Morphological Reconstruction Aimed at Phenotyping and Detection of Crop Stress*," Poster and oral presentation at the XV SOI Scientific Days, Carmignani Educational Center, Pisa, Italy, June 2025.

- Discussed in **Section 6.3**.

Table of contents

Table of contents	ix
List of figures	xi
List of tables	xiv
Glossary	xv
1 Introduction	1
1.1 Introduction	2
1.2 Thesis Structure and Chapters Overview.....	3
1.3 Methodological Progression Across the Thesis Chapters.....	6
1.4 Motivating Example	7
2 Theoretical Background	8
2.1 Neural Network (NN) Foundations	9
2.2 Neural Network Architectures	10
2.3 Reinforcement Learning (RL)	16
2.4 Quantitative Metrics for Model Evaluation	23
3 Virtual Sensing and Modeling of Greenhouse Temperature	25
3.1 Introduction	26
3.2 Related Work	26
3.3 Problem Formalization and Proposed Solution	28
3.4 Application Setup	34
3.5 Experimental Results.....	36
3.6 Discussion and Comparative Analysis	41
3.7 Conclusion	43
4 Modeling of Autonomous Rover for Greenhouse Temperature Monitoring	44
4.1 Introduction	45
4.2 Related Work	45
4.3 Problem Formalization and Proposed Solution	47
4.4 Experimental Results.....	57
4.5 Discussion and Comparative Analysis	61
4.6 Conclusion	63
5 Modeling the Growth of Rainbow Trout for the Digital Twin of an Aquaculture Facility	65
5.1 Introduction	66
5.2 Related Work	70
5.3 Problem Formalization and Proposed Method	72
5.4 Application Setup and Data Description	76
5.5 Experimental Results.....	77
5.6 Conclusion	88
6 A Digital Twin Framework for Modeling <i>Salicornia</i> Growth	90
6.1 Introduction	91
6.2 Related Work	92
6.3 Phase 1: The Perception Validation Layer	95
6.4 Phase 2: Hybrid Growth Modeling for <i>Salicornia</i>	107
6.5 Experimental Results.....	131
6.6 Conclusion	142
7 Conclusion	143

7.1 Conclusion	144
Appendix A Mathematical Formulations of Nonlinear Growth Models	147
A.1 Logistic Model	147
A.2 The Gompertz Model	148
A.3 The Grey Verhulst Model	148
Appendix B Nonlinear Model Fitting: Optimization Procedures and Parameter Constraints	150
Bibliography	151

List of figures

1.1	A scalable aquaponic facility with the corresponding unified digital twin.	7
2.1	Architecture of Feedforward Neural Network (FNN)	11
2.2	Architecture of the Recurrent Neural Network (RNN)	12
2.3	Vanishing and Exploding gradients: A major challenge in training Recurrent Neural Networks.	13
2.4	Architecture of the Long Short-Term Memory (LSTM) cell.	13
2.5	Architecture of the Convolutional Neural Network (CNN).	14
2.6	Schematic representation of the Reinforcement Learning framework.	17
2.7	Exemplification of the Proximal Policy Optimization functioning, inspired by [1].	22
3.1	Architecture of the baseline BiLong Short-Term Memory (LSTM) model for virtual sensing... ..	30
3.2	Architecture of an LSTM followed by Convolutional Neural Network (CNN).	31
3.3	Architecture of a CNN followed by LSTM.	31
3.4	Architecture with an LSTM and a CNN in parallel.	32
3.5	Time-Large Language Model (LLM) pipeline: from raw time series to LLM-based prediction.	33
3.6	Example of domain-specific prefix prompt used in Time-LLM.	33
3.7	Application setup in north-eastern Italy, showing the two adjacent greenhouse tunnels used for data collection.	34
3.8	Sensor placement within the greenhouse.	35
3.9	Examples of extreme weather condition variations observed within a one-month period of the experimental dataset.	35
3.10	Evaluation of virtual-sensing performance for the baseline LSTM model and the proposed CNN-LSTM hybrid architecture, showing predicted versus actual temperature values.	37
3.11	Real temperatures and Time-LLM predictions for Point of Interest (POI) 1 with sequence and prediction length equal to 6 (i.e., 1.5 hours).	39
3.12	Comparison of observed temperatures and predictions from a global Time-LLM model trained with and without POI 7.	41
4.1	Greenhouse layout highlighting the fixed navigation path and the spatial distribution of sampling POIs.	48
4.2	Temperature measurements at two POIs	49
4.3	Greenhouse configuration where the proposed methodology identifies poi_2 and poi_5 as optional POIs to skip during the rover's trajectory.	50
4.4	Diagram of the proposed DRL approach, based on Deep Q-Learning (DQL) for context-aware trajectory planning.	51
4.5	Example of enriched model environment where some POIs have been identified as charging stations and some direct paths have been identified.	52
4.6	Example of environment represented through the labeled graph introduced in Def. Definition 4.4, a generic value c_i denotes a capacity value greater than 0.	54
4.7	Setup of the real-world case scenario considered for the experiments.	57
4.8	Greenhouse environment considered for experiments.	58
4.9	A sample of suggested trajectory	59
4.10	Application of the approach in different environments	60
5.1	Purely data-driven machine learning workflow.	67
5.2	Loss function of knowledge-informed machine learning	67
5.3	Enrichment of the input dataset implemented by the proposed hybrid approach	68
5.4	A cyclical representation of PFF inspired by[2].	70
5.5	LSTM followed by CNN Model Architecture.	75
5.6	Pilot trout farm in Preore (Trento), instrumented during the GAIN project for environmental and biomass monitoring.	76
5.7	Biomass Daily System functioning.	77

5.8	Hourly weight measurements for Campaign 2 acquired by the Biomass Daily (BD) system. ...	79
5.9	Smoothed daily weight estimates for Campaign 2 after noise reduction and aggregation. ...	80
5.10	Line plot comparing the average weight predicted by the proposed hybrid model and the ground truth values produced by the BD system.	84
5.11	Scatter plot comparing the average weight predicted by the proposed hybrid model and the ground truth values produced by the BD system.	84
5.12	Residual plot illustrating the difference between the BD measures and the fish weight estimated by the hybrid model along the full sequence prediction.....	85
5.13	SHapley Additive exPlanations (SHAP) Mean absolute SHAP feature importance comparing the proposed hybrid method with the pure data-driven approach.	87
6.1	Overview of Chapter 6 Structure, Showing the Dual-Phase Pipeline for Perception Validation and Growth Modeling.	92
6.2	Semantic segmentation results produced by Pixellib.....	96
6.3	Instance segmentation results produced by Pixellib.	97
6.4	Overview of Chapter 6 Structure, Phase 1.	99
6.5	Representative instance segmentation output using Pixellib, showing the detected potted plant. Bounding boxes and masks define the Region of Interest for subsequent analysis.	100
6.6	Sample plant visualized in different color spaces (HSV, CIELAB, and CMYK) used for segmentation and feature extraction.	101
6.7	Segmentation using CIELAB a^* channel	102
6.8	PlantCV size-analysis output showing plant height estimated from the front view (2,312 pixel).	103
6.9	Top-view images used for plant width and convex hull computation: (a) original image and (b) corresponding binary mask.....	103
6.10	PlantCV size analysis output for top-view measurements (width: 2,131 px; convex hull area: 2,404,237.5 px ²).	104
6.11	Image-based estimation of pot diameter used for pixel-to-millimeter calibration in top-view images.	105
6.12	Phenospec 3D scanning system for high-precision plant trait measurement under controlled environments.	106
6.13	Overview of Chapter 6 Structure, Phase 2.	108
6.14	Main components of the marine aquaponic pilot plant are located at the scientific Campus of Ca' Foscari University: Fish Tank (FT); Mechanical Filter (MF); Green House (GH); BioFilter (BF).	109
6.15	View of the hydroponic greenhouse at the UNIVE pilot site, showing the aquaponic integration of fish tanks (Right), and plant cultivation beds (Left) used for continuous environmental monitoring.	110
6.16	<i>Salicornia</i> plants cultivated on net supports in the UNIVE hydroponic greenhouse.....	111
6.17	Camera installation used in the IP-UNIVE greenhouse for continuous <i>Salicornia</i> image acquisition.	112
6.19	Spatiotemporal segmentation and instance isolation pipeline.	116
6.20	Pattern sheet used for spatial calibration and pixel-to-millimeter ratio estimation.	117
6.21	Reference patch extraction pipeline.....	119
6.22	Feature extraction pipeline: instance segmentation and chromatic preprocessing (Part I)....	122
6.22	Feature extraction pipeline: binary masking, skeletonization, and geometric quantification (Part II).	123
6.23	Sigmoid plant growth curve showing lag, log, and stationary growth phases.	127
6.24	Illustration of critical growth points (inflection and rapid-growth interval) of sigmoidal models.	129
6.25	Critical points estimated from the fitted logistic growth model for the weight data of a sample plant.	130
6.26	Logistic growth curve fitting for the central plant observed by Camera 5. The top row shows fits for the Area and Convex Hull proxies, and the bottom plot shows the reference Weight fit. Goodness-of-fit metrics are reported with each curve.	136
6.27	Comparison of prediction fidelity for two representative plants using the Full Model. Left: high-fidelity case ($RMSE = 8.51$ g, $MAE = 5.36$ g) with strong temporal alignment. Right: moderate case ($RMSE = 24.22$ g, $MAE = 19.90$ g) showing increased scatter and trend deviation.....	138

6.28 SHAP analysis of feature importance, Mean absolute SHAP values indicate average impact on model output	139
6.29 Prediction comparison for the middle plant captured by Camera 2, across the three ablation scenarios: the full model, the model without environmental data, and the model without model-based features.	140
6.30 SHAP importance comparison under reduced feature configurations.	141
6.31 Predicted vs. observed biomass trends for two representative Phase 1 plants.	141

List of tables

3.1	Comparison of temperature-prediction errors for virtual sensing using the baseline LSTM and the proposed CNN-LSTM hybrid models.	36
3.2	Error comparison between the DLinear and Time-LLM models in the presence of univariate and multivariate time series and different sequence and prediction lengths.	39
3.3	Error comparison for all virtual POIs between Time-LLM, the LSTM + CNN model, and the baseline Bi-directional LSTM. The best result for each metric is highlighted in bold.	40
3.4	Performance of the global Time-LLM model trained on all virtual POIs.	40
4.1	Summarization of used symbols.	55
4.2	Example of stop pattern suggested by the proposed approach.	58
4.3	Comparison between the proposed approach and the baseline [3].	58
4.4	Comparison of estimation errors obtained by the LSTM model when trained using trajectories generated by: (i) the Proximal Policy Optimization (PPO)-based approach [4], (ii) the DQL-based approach [5], and (iii) the baseline complete trajectories [3].	61
5.1	Feature categories with descriptions and units	73
5.2	Metadata and sampling parameters for the three monitoring campaigns.	78
5.3	Comparison of different recurrent-convolutional architectures for fish weight prediction on Campaign 2, using the full feature set. Results are reported as mean $\pm$ standard deviation..	81
5.4	Comparison of performance metrics for the pure data-driven baseline and the proposed hybrid data-driven, model-driven approach across Campaigns 2 and 3.	82
5.5	Comparison of performance metrics for the mechanistic model, pure data-driven baseline, and the proposed hybrid approach across Campaigns 2 and 3.....	83
5.6	Performance comparison between the proposed hybrid method and the pure data-driven approach across training steps.....	85
5.7	Performance with different training set proportions.	86
6.1	Comparison of Convex Hull Area (mm ²) and Plant Height (mm) between the Proposed Method (PM) and <i>Phenospex</i> (PHX).	106
6.2	Mean $\pm$ SEM of absolute errors for Convex Hull Area and Plant Height across species.	106
6.3	Summary of <i>Salicornia</i> image acquisition periods across Phase I and Phase II.	113
6.4	Feature categorization for <i>Salicornia</i> growth analysis. [Var] denotes each environmental variable (Oxygen, Temperature, SAT, pH, Salinity).....	132
6.5	Numerical comparison of the Logistic, Gompertz, and Grey Verhulst growth models fitted to Convex Hull Area, Area, and Weight.	133
6.6	Synchronization performance between image-derived proxies and plant weight. Metrics report mean $\pm$ SEM delays of critical growth milestones and the correlation between the growth rates of morphological features and Weight across plants of Phase 2. Reliability quantifies the proportion of plants whose fitted proxy-weight dynamics achieve $r > 0.85$. .	134
6.7	Extracted biological milestones for Phase 2 plants using the Logistic growth model.	135
6.8	Hyperparameter Search Space for Grid Search	136
6.9	Optimal Gradient Boosting Parameters	137
6.10	Ablation study results for <i>Salicornia</i> biomass estimation using Gradient Boosting. Metrics are reported as Mean $\pm$ Standard Error (SEM) across 7 folds using Group K-Fold cross-validation.	137
6.11	Biomass estimation performance on aggregated Phase 1 + Phase 2 dataset using Group K-Fold cross-validation. Values represent Mean $\pm$ SEM across 5 folds.	141

Glossary

DT	Digital Twin
IoT	Internet of Things
NN	Neural Network
DL	Deep Learning
AI	Artificial Intelligence
ML	Machine Learning
ANN	Artificial Neural Network
MLP	Multilayer Perceptron
LLM	Large Language Model
CNN	Convolutional Neural Network
FNN	Feedforward Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
GBR	Gradient Boosting Regressor
SGD	Stochastic Gradient Descent
NLP	Natural Language Processing
RL	Reinforcement Learning
DRL	Deep Reinforcement Learning
DQL	Deep Q-Learning
DQN	Deep Q-Network
PPO	Proximal Policy Optimization
SciML	Scientific Machine Learning
KIML	Knowledge-Informed Machine Learning
SHAP	SHapley Additive exPlanations
CFD	Computational Fluid Dynamics
NLLS	Non-Linear Least Squares
R^2	Coefficient of Determination
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MSE	Mean Square Error

RMSE	Root Mean Square Error
SEM	Standard Error of the Mean
ROI	Region of Interest
POI	Point of Interest
BW	Black and White
px	Pixel
mm	Millimeter
WSN	Wireless Sensor Networks
BD	Biomass Daily
PFF	Precision Fish Farming
TAN	Total Ammonia Nitrogen
DO	Dissolved Oxygen

Chapter 1

Introduction

Chapter contents

1.1	Introduction	2
1.2	Thesis Structure and Chapters Overview	3
1.3	Methodological Progression Across the Thesis Chapters	6
1.4	Motivating Example	7

1.1 Introduction

A Digital Twin (DT) is commonly defined as a dynamic virtual representation of a physical system that is continuously updated with real-time data from its physical counterpart [6, 7]. In this context, modeling acts as the computational core that allows the DT to adapt its internal state and structure as new information becomes available [8].

DT modeling approaches are traditionally classified into two main categories:

1. **Model-Driven (Mechanistic) Approaches:** rely on domain knowledge and are typically expressed through physical or biological laws, often formalized as systems of differential equations [9]. These methods rely on established domain knowledge and are usually represented through physical or biological laws, often formalized as systems of differential equations. Such models provide high interpretability and strong explanatory power. However, they require extensive calibration and may struggle to accurately represent unknown or poorly understood dynamics [10, 11].
2. **Data-Driven Approaches:** In contrast, these methods infer system behavior directly from observations, learning patterns without explicit mechanistic assumptions. While data-driven approaches can model complex nonlinear relationships effectively, they generally require large datasets and may produce black-box predictions that do not adhere to biological or physical constraints [12, 11].

The primary objective of this thesis is to develop a robust integrated DT framework designed for complex systems, such as greenhouses and aquaponic facilities. The focus is on bridging traditional model-driven (mechanistic) approaches with modern data-driven (Machine Learning (ML)) techniques, addressing the limitations of purely “black-box” models that often lack biological grounding.

Within this broader objective, biological domains, specifically plant and aquaculture systems, serve as representative case studies of complex, dynamic environments. These domains provide rich, real-world testbeds for evaluating how an integrated DT can capture nonlinear development, environmental interactions, and system-level feedback, without implying that the thesis is limited to biological modeling alone.

While the concept of the DT has matured within industrial and manufacturing domains [13], extending it to biological systems introduces substantially greater complexity [14]. Unlike engineered components, which operate under stable and well-characterized physical laws, biological organisms exhibit nonlinear, heterogeneous, and context-dependent behaviors. Environmental conditions such as microclimate variability, nutrient availability, and radiation dynamics interact with biological processes in ways that are difficult to formalize mechanistically. As a result, maintaining a continuously updated virtual representation requires models that can accommodate uncertainty, spatial variability, and dynamic feedback between the organism and its environment.

Despite these challenges, high-fidelity biological DT offer significant advantages. They promote resource-efficient operations by enhancing the control of water, energy, and nutrient flows [15]. Additionally, these models support early detection of stress in plants or fish through virtual sensing and predictive analytics [16]. They also reduce the need for dense sensor networks by inferring unmeasured states from hybrid models [5, 17].

This work considers two biological domains, plant and fish, as use cases for the proposed hybrid framework. These domains have distinct architectural requirements for a DT due to their different interactions with the environment.

Vegetables are stationary organisms, meaning their physiological well-being primarily depends on localized microclimate conditions. Therefore, modeling these systems relies heavily on high-resolution

mapping of variables at specific locations. On the other hand, fish are mobile agents, and their metabolism and environmental exposure are closely linked to their movement within the aquatic environment. This mobility introduces dynamic mechanistic factors, such as varying oxygen demands and the distribution of metabolic waste, that require a higher temporal resolution than the relatively slow, monotonic changes observed in greenhouse conditions.

Although these distinctions necessitate specialized mechanistic formulations, they do not alter the overall structure of the integrated data-driven and model-driven DT framework proposed in this thesis. This framework is designed to be applicable across diverse biological systems.

1.2 Thesis Structure and Chapters Overview

This thesis develops an integrated DT framework through a progressive methodology that evolves from environmental perception to biologically grounded, multimodal systems. Conceptually, the proposed framework follows a layered architecture in which each stage introduces additional sources of knowledge and complexity.

At the foundation lies the “Environmental Perception Layer”, which focuses on microclimate mapping and modeling. Structural knowledge of the greenhouse, including its geometry and spatial organization, is combined with data-driven temporal models to reconstruct and predict environmental states. This layer provides a computationally efficient baseline for real-time operation, particularly in scenarios where detailed mechanistic climate models are unavailable or impractical. The integration of autonomous rovers further extends this capability through the “Dynamic Data Acquisition Layer”, where mobile sensing agents improve spatial coverage and enable more informative sampling compared with fixed sensor networks.

Building upon this environmental foundation, the framework expands toward biological processes through the Hybrid data-driven, model-driven modeling layer. This stage addresses the limitations of purely data-driven approaches in systems governed by complex physiological mechanisms. Mechanistic biological knowledge, such as bioenergetic models and nonlinear growth relationships, is integrated with learned representations to introduce physically meaningful descriptors and improve model interpretability. This hybrid approach is demonstrated through the Rainbow Trout growth case study, where biological dynamics are captured by combining domain knowledge with data-driven learning.

Finally, the framework extends to multimodal DTs, in which heterogeneous data sources, including images and sensor measurements, are jointly exploited. The *Salicornia* study represents this stage by integrating perception validation, phenotyping reliability analysis, and hybrid growth-dynamics estimation.

Together, these layers establish a unified DT framework that progressively balances scalability, flexibility, and interpretability. Starting with data-driven environmental reconstruction and extending to biologically informed and image-based modeling, the proposed methodology demonstrates how DTs can bridge observational data and domain-specific knowledge across complex agricultural and biological systems. The following parts outline the general role and contribution of each chapter within the overall DT framework.

- **Chapter 2- Theoretical Background:**

Chapter 2 provides the necessary methodological and theoretical background to navigate the integration of model-driven and data-driven paradigms explored in this thesis. It begins by formalizing the architectures of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) networks for analyzing temporal sequences, as well as Convolutional Neural Network (CNN) for extracting spatial features. Additionally, this chapter introduces the theoretical framework of Deep Reinforcement Learning (DRL), focusing on the Proximal Policy Optimization (PPO) and Deep Q-Learning (DQL) algorithms used for autonomous decision-making. By synthesizing these diverse methodologies, this

chapter serves as a conceptual bridge that supports the development of the hybrid, data-driven, model-driven frameworks and DT architectures discussed in the subsequent chapters.

- **Chapter 3- Virtual Sensing and Modeling of Greenhouse Temperature:**

Chapter 3 develops the “Environmental Perception Layer” of the DT by introducing a virtual sensing framework to reconstruct the greenhouse microclimate in areas where physical sensors are absent. The approaches discussed in this chapter address a key limitation of greenhouse monitoring systems: achieving complete spatial coverage with physical sensors alone is impractical because a greenhouse would require an extremely dense sensor network to capture its fine-scale microclimate variability. A high-fidelity DT requires dense environmental data throughout the entire greenhouse volume. However, deploying a sufficiently dense sensor network would lead to high installation and maintenance costs, increased energy consumption, and significant operational complexity. As a result, solely hardware-dependent solutions cannot provide the spatial resolution necessary for advanced greenhouse management and DT applications.

To address these limitations, this chapter presents a virtual sensing framework that reconstructs the microclimate at locations inside the greenhouse where no physical climate sensors are installed. This approach allows the DT to operate continuously across the environmental field without the need for a dense physical sensor deployment. The framework integrates the greenhouse’s structural model directly into the data used by data-driven models. Rather than treating sensor measurements as independent time series, the model inputs include information about the greenhouse’s physical layout, such as the geometric distances between Point of Interest (POI)s and climate stations, and their relative positions within the structural envelope. These spatial descriptors provide meaningful physical context that enhances learning, enabling better representation of the underlying mechanisms that govern heat, humidity, and airflow distribution.

Building on this structural foundation, the chapter presents a unified modeling pipeline that progressively advances from a baseline data-driven LSTM model introduced in [18] to more expressive hybrid models. In particular, the Hybrid CNN-LSTM [17] exploits convolutional layers to extract spatial patterns related to greenhouse geometry, while recurrent layers capture the temporal dynamics of the climate variables. The framework is further extended by investigating Time-Large Language Model (LLM) [19], which reformulates time-series data into prototype representations to leverage pre-trained knowledge and improve generalization across space and time.

- **Chapter 4- Modeling of Autonomous Rover for Greenhouse Temperature Monitoring:**

Chapter 4 introduces the “Dynamic Data Acquisition” layer. Unlike traditional virtual sensing pipelines, which rely on predefined, fixed POIs for training data, the methods discussed in this chapter enable autonomous rovers to collect microclimate measurements at dynamic, context-dependent locations while performing agricultural tasks. This approach alleviates the limitations of static POI grids and enables the decision-making DT to receive richer, more diverse, and representative environmental data.

Since the rover’s sensing points are determined by its movement and workload, the central challenge is to ensure that these dynamically sampled measurements provide sufficient information to support high-quality virtual sensing. The methods developed in this chapter tackle this challenge by progressively enhancing the rover’s decision-making strategy for selecting sampling locations to visit, using DRL. Starting from a deterministic baseline in which the robot follows a fixed cyclical path and treats all POIs equally [3], the methodology evolves toward strategies based on DQL and PPO that actively

guide the rover toward high-information areas [5, 4]. This guidance respects physical constraints, such as battery capacity, travel time, and access to charging stations.

By integrating model-driven knowledge of the robot's kinematics and energy constraints with data-driven learning of information value, the rover becomes an intelligent sensing agent capable of generating higher-quality, more informative data streams. Importantly, incorporating this improved data into the virtual sensing models discussed in chapter 3 led to significant enhancements in environmental state estimation accuracy compared to the baseline framework that relied on a fixed-trajectory robot. Through this integration, the decision-making system benefits from enhanced data richness and improved predictive performance, resulting in a more accurate, adaptive, and operationally efficient representation of the greenhouse microclimate.

- **Chapter 5- Modeling the growth of Rainbow Trout for the DRL of an aquaculture facility:**

The previous two chapters have focused on advancing data-driven environmental modeling in greenhouses, first through virtual sensing and then through autonomous, information-aware data acquisition. These chapters established how increasingly sophisticated data-driven models can be strengthened by embedding structural or physical knowledge. Building on this progression, this chapter extends the same principles to biological organisms. We will explore a hybrid approach that combines mechanistic, model-driven knowledge with data-driven methods. The aquaponics use case, which models Rainbow Trout growth, provides additional biological validation for the integrated DT framework.

This chapter discusses the practical implementation of a DT in aquaponics. Specifically, chapter 5 provides a comprehensive and operational example by creating a virtual replica of a rainbow trout aquaculture facility. This example demonstrates how physical biological processes can be mirrored, interpreted, and predicted using an integrated model-driven and data-driven framework.

The pilot operates in a real aquaculture facility, where in-situ sensors monitor key water-quality variables, while an automated vision system estimates fish size and growth without manual handling. These environmental and biological data streams create a multi-modal foundation that keeps the DT aligned with the real system

The key methodological contribution is a hybrid framework that combines data-driven and model-driven approaches, enriching the data-driven model with mechanistic bioenergetic features. By applying established biological equations, the system calculates latent variables such as energy intake, catabolic losses, and somatic tissue energy content, which capture the underlying metabolic processes driving growth.

These mechanistic descriptors are combined with raw sensor data and processed within a hybrid LSTM-CNN architecture. In this architecture, the LSTM layers learn long-term growth patterns, while the CNN layers model localized, nonlinear interactions between environmental stressors and metabolic states. This integration of mechanistic understanding with deep learning not only enhances predictive accuracy but also improves data efficiency; the hybrid model, trained on only half of the available dataset, outperformed a purely data-driven model trained on the entire dataset.

Ultimately, this pilot illustrates how a DT can unify various sensing modalities, biological knowledge, and ML to create a robust, interpretable, and continuously updating virtual representation of a living aquaculture system.

- **Chapter 6- A DT Framework for Modeling Salicornia Growth:**

Chapter 6 discusses the development of a DT framework designed for monitoring the growth of *Salicornia*. It illustrates how visual observations from the physical cultivation system are synchronized with a virtual representation that models the plant's continuous development. The work emphasizes integrating data-driven learning with biologically informed growth models, enabling the system to interpret raw images in the context of established physiological patterns.

The content of this chapter is organized into two complementary phases.

- Phase 1 establishes the perception validation layer, demonstrating that image-derived traits can reliably represent biological structure. The validated pipeline presented in this section is then used to extract structural traits from *Salicornia* in the following phase.
- Phase 2 applies these validated traits to construct a hybrid growth-modeling framework that synchronizes physical observations with a virtual representation of *Salicornia* development.

A key achievement of this study is the application of nonlinear growth functions to derive significant biological indicators, including growth velocity, acceleration, and developmental transitions. These indicators enable the DT to characterize growth stages and identify key physiological milestones, thereby facilitating explainable predictions that align with the organism's actual biological progression.

This chapter introduces a low-cost image dataset of *Salicornia*, specifically developed for this research. It aims to fill the gap in available visual resources for this crop by capturing the plant's morphological development across various growth stages and environmental conditions. By detailing the creation of this dataset, the chapter lays the groundwork for the DT, facilitating the transition from raw imagery to precise, biologically accurate biomass estimation.

1.3 Methodological Progression Across the Thesis Chapters

The proposed DT framework progresses from a data-driven environmental perception layer to explicitly coupled hybrid models, balancing predictive accuracy with the computational constraints of real-world deployment.

Although the overall goal of this thesis is to integrate model-driven and data-driven approaches, Chapters 3 and 4 intentionally adopts a purely data-driven strategy. These chapters establish the "Environmental Perception" and "Dynamic Data Acquisition Layer" of the DT, in which structural layout in terms of geometric distances and historical environmental information are embedded as contextual features for learning greenhouse microclimate relationships. This approach does not rely on explicit physical formulations such as energy-balance equations or Computational Fluid Dynamics (CFD) simulations; instead, the relationships between variables are learned implicitly from data.

This design reflects both practical and computational constraints. Detailed mechanistic greenhouse climate models are difficult to calibrate, highly sensitive to structural variability, and computationally unsuitable for deployment on edge devices. Incorporating structural descriptors as contextual features, therefore, provides a pragmatic way to embed physically meaningful information while maintaining scalability, robustness, and low computational cost.

Building on this foundation, Chapters 5 and 6 introduces explicit hybrid modeling where reliable biological knowledge is available. Chapter 5 integrates bioenergetic metabolic descriptors with a deep LSTM-CNN model for fish-growth prediction, while Chapter 6 combines multimodal plant phenotyping with nonlinear sigmoid growth models to derive biologically meaningful developmental indicators.

This progression demonstrates that the methodological difference between environmental and biological modeling is intentional. The environmental layer emphasizes context-aware, data-driven modeling

to satisfy practical deployment constraints, whereas the biological layer incorporates explicit mechanistic knowledge where well-established domain models are available. Together, these chapters present a coherent DT framework that strategically combines implicit structural feature engineering for efficient environmental monitoring with explicit hybrid model-driven learning for accurately representing complex living systems.

1.4 Motivating Example

To clarify how the different methods presented in this thesis are interconnected, we consider an extended, more complex version of the BeBlue aquaponic pilot (Section 6.4.1). In the BeBlue pilot, fish tanks, filtration units, and hydroponic plant beds operate within a closed, recirculating loop, serving as a demonstrator of sustainable circular food production suitable for coupling with its corresponding DT. As illustrated in Figure 1.1, in this expanded scenario, the greenhouse contains multi-tier vertical farming racks equipped with low-cost cameras that continuously monitor plant development. Readings from a small number of fixed climatic sensors are integrated with those from autonomous rovers that move between the racks for operational purposes, thereby providing fine-grain mapping of temperature, humidity, airflow, and other climate information. Since rovers are battery-powered, their trajectories must account for charging docks to ensure reliable long-term operation. The climatic measurements collected by the rovers feed the virtual sensing model in the DT, which reconstructs the climate in unmeasured areas to detect potential disease risks and provide additional multimodal input for plant growth analysis and modeling. The facility also includes fish tanks and recirculating aquaculture units, whose corresponding hybrid models in the DT predict fish biomass trends to inform appropriate feeding strategies and maintain system stability.

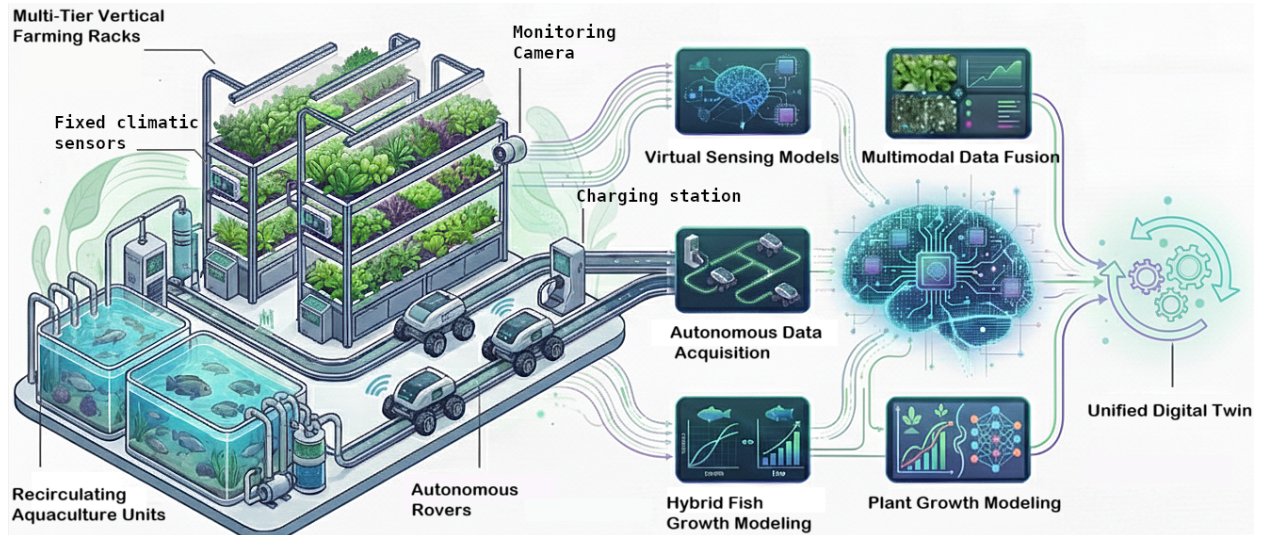


Figure 1.1: A scalable aquaponic facility with the corresponding unified digital twin.

Considering this unified example, the methods developed in the following chapters, i.e., virtual sensing (Chapter 3), autonomous data acquisition using rovers (Chapter 4), hybrid fish and plant growth modeling, and multimodal data fusion (Chapters 5 and 6), are not isolated topics. They form complementary components of a coherent methodology for building a DT architecture capable of managing a complex biological-environmental system at scale.

Chapter 2

Theoretical Background

Chapter contents

2.1	Neural Network (NN) Foundations	9
2.1.1	The Artificial Neuron	9
2.1.2	The Multi-Layer Perceptron (MLP)	9
2.1.3	Learning via Backpropagation	10
2.1.4	Optimization Algorithms	10
2.2	Neural Network Architectures	10
2.2.1	Feedforward Neural Network (FNN)	10
2.2.2	Long Short-Term Memory (LSTM) Neural Network	12
2.2.3	Convolutional Neural Network (CNN)	14
2.2.4	Transformer Models and Large Language Models (LLMs)	15
2.3	Reinforcement Learning (RL)	16
2.3.1	Key Elements of Reinforcement Learning	17
2.3.2	Practical Applications and Illustrative Examples of RL	18
2.3.3	Deep Q-Learning (DQL)	19
2.3.4	Proximal Policy Optimization (PPO)	21
2.4	Quantitative Metrics for Model Evaluation	23
2.4.1	Mean Absolute Error (MAE)	24
2.4.2	Mean Absolute Percentage Error (MAPE)	24
2.4.3	Mean Squared Error (MSE)	24
2.4.4	Root Mean Squared Error (RMSE)	24
2.4.5	Coefficient of Determination (R^2)	24

This chapter introduces the essential theoretical tools that will be utilized throughout the thesis. It begins with the foundational concepts of Neural Network (NN)s, including artificial neurons and Multilayer Perceptron (MLP). The discussion then expands to include temporal models such as RNNs and LSTMs, as well as spatial modeling with CNNs. In addition to these classical architectures, the chapter introduces Transformer models and LLMs, which have emerged as central components of modern machine learning for modeling complex, long-range dependencies. Additionally, the principles of DRL are explored, with a focus on PPO and DQL.

This section also introduces the quantitative metrics for model evaluation that will be used consistently throughout the thesis.

Together, these methodologies provide the foundation for the hybrid model-driven and data-driven DT framework that will be developed in the following chapters.

2.1 Neural Network (NN) Foundations

Artificial Neural Network (ANN) constitute the core computational framework of modern Deep Learning (DL) systems. Inspired by simplified models of biological neural processing, they serve as flexible, nonlinear function approximators that can model complex relationships across high-dimensional data domains. This section introduces the foundational components of NNs, including the artificial neuron, the multi-layer perceptron architecture, the backpropagation learning process, and the optimization algorithms that facilitate efficient training [20].

2.1.1 The Artificial Neuron

The basic computational unit of a NN is the artificial neuron, which was first formalized in Rosenblatt's perceptron model [21]. Given an input vector $\mathbf{x}$, the neuron calculates a weighted sum, adds a bias term, and then applies a nonlinear activation function:

$$y = \sigma \left(\sum_{i=1}^n w_i x_i + b \right) \quad (2.1)$$

In this context, w_i represents the learnable weights, b denotes the bias, n is the dimensionality of the input vector, and $\sigma(\cdot)$ indicates the activation function. Activation functions introduce nonlinearity into NNs, enabling them to model complex relationships that cannot be captured by purely linear transformations [20]. Several families of activation functions exist, each with distinct properties and behaviors. Classical choices include the sigmoid and hyperbolic tangent functions, which compress inputs into bounded ranges and were widely used in early NN architectures. More recent approaches favor unbounded functions that alleviate issues such as vanishing gradients. A particular example of these activation functions is the Rectified Linear Unit (ReLU), defined as:

$$\text{ReLU}(z) = \max(0, z)$$

ReLU outputs 0 for negative inputs and increases linearly for positive inputs. This activation function is widely adopted due to its computational simplicity and favorable gradient properties, which mitigate vanishing gradient effects and accelerate training [20].

2.1.2 The Multi-Layer Perceptron (MLP)

When neurons are arranged into an input layer, one or more hidden layers, and an output layer, the resulting structure is referred to as an MLP. Information flows in a unidirectional manner, moving from the input

layer to the output layer. The representational capacity of an MLP depends on its depth (number of layers) and width (number of neurons per layer). Deeper networks capture hierarchical abstractions, while wider networks model richer feature interactions [20].

2.1.3 Learning via Backpropagation

NNs learn by adjusting their parameters, which include weights and biases, to minimize a loss function. Common loss functions are Mean Square Error (MSE) for regression tasks and Cross-Entropy for classification tasks. A significant advancement that made effective training of multilayer networks possible is the backpropagation algorithm, introduced by Rumelhart, Hinton, and Williams in 1986 [22]. Backpropagation utilizes the chain rule of calculus to efficiently compute the gradients of the loss with respect to each weight, enabling iterative refinement during training. This algorithm remains the backbone of modern DL.

2.1.4 Optimization Algorithms

Once the gradients are computed, an optimizer updates the network's parameters. The most straightforward and most traditional method for this is Stochastic Gradient Descent (SGD), which adjusts the weights using small batches of data. However, modern DL often employs optimizers that automatically adapt the learning rate during training. One of the most popular choices is Adam (Adaptive Moment Estimation) [23], which tracks both the average and the variability of the gradients. This allows Adam to make stable, efficient updates, often resulting in faster, more reliable convergence than basic SGD.

2.2 Neural Network Architectures

The choice of NN architecture is primarily dictated by the nature of the data and the specific requirements of the underlying task. NNs are commonly categorized into Feedforward Neural Network (FNN) and RNN, distinguished by the way information flows through their architectures [24]. While FNNs are highly effective for mapping static inputs to outputs, tasks involving sequence analysis require models that account for the temporal dependencies and patterns inherent in time series data.

In the context of meteorological forecasting, the arrow of time is critical; a prediction for a future time step depends heavily on observations from previous hours or days. While modern attention-based architectures like “transformers” have gained popularity [25], RNNs and their variants remain essential for analyzing numerical sequences due to their emphasis on temporal persistence. This section will discuss the evolution of these architectures, beginning with the basic feedforward structure and then focusing on the mechanisms of recurrence.

2.2.1 Feedforward Neural Network (FNN)

The FNN is one of the earliest models in NN architecture [26]. These networks learn directly from input data to accomplish specific tasks. For instance, in an image classification problem, the objective is to assign each image to a predefined category. When presented with an input image, an FNN processes it through multiple layers and generates a set of scores, one for each possible category. The category that receives the highest score is considered the network's prediction. In this manner, the FNN effectively maps raw input data to a final decision about which label most accurately corresponds to the input.

In feedforward networks, as shown in Figure 2.1, information flows in a single direction: from the input layer, through one or more hidden layers, to the output layer. There are no cycles or feedback connections,

so once an activation leaves a layer, it does not return. The MLP described in Section 2.1.2 is a specific instance of FNN, characterized by fully connected layers and nonlinear activation functions.

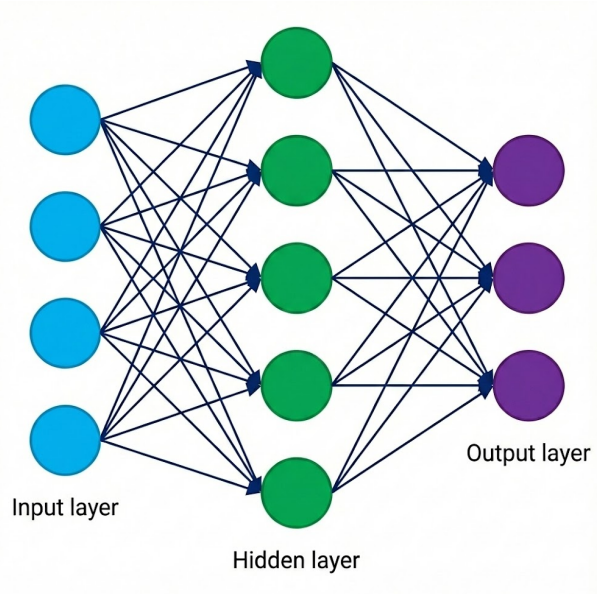


Figure 2.1: Architecture of Feedforward Neural Network (FNN)

While feedforward models are powerful universal approximators of static mappings [27], they lack the ability to represent sequential and temporal dependencies. To address this limitation, RNNs introduce feedback connections that allow information to persist across time steps, making them suitable for modeling sequences and temporal dynamics.

2.2.1.1 Recurrent Neural Network (RNN)

The RNN is a DL architecture designed to process sequential data, where the order and temporal dependencies between elements are essential [28]. Unlike FNNs, which assume independent inputs, RNNs introduce feedback connections that allow activations to cycle back into the network (Figure 2.2). This recurrent structure enables the model to retain information over time.

At each time step, the RNN maintains an internal hidden state h_t that acts as a memory of past inputs. This state is updated using the current input x_t and the previous hidden state h_{t-1} :

$$h_t = f(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (2.2)$$

and the output at time t is computed as:

$$y_t = g(W_{hy}h_t + b_y) \quad (2.3)$$

Where f and g are nonlinear activation functions. Without nonlinearity, multiple layers of an NN would effectively collapse into a single linear transformation, offering no advantage over straightforward linear regression. By incorporating $f(\cdot)$, the network is able to model nonlinear relationships, allowing it to better approximate the dynamics of meteorological systems.

The hidden state summarizes relevant information from previous time steps, allowing the network to incorporate temporal context when producing outputs.

A defining characteristic of the RNN is parameter sharing: the same weight matrices (W_{xh}, W_{hh}, W_{hy}) are applied at every time step. This design significantly reduces the total number of trainable parameters

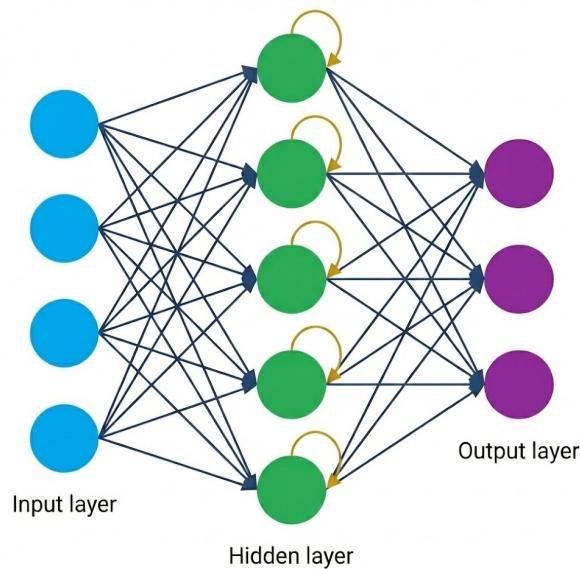


Figure 2.2: Architecture of the Recurrent Neural Network (RNN)

compared to a deep FNN. More importantly, it allows the model to generalize across time and learns to recognize a specific pattern (such as a sudden drop in pressure) regardless of whether it occurs at the beginning, middle, or end of the observation window.

Thanks to this recurrent structure, RNNs are explicitly architected to map sequences of arbitrary length. They are widely used in applications such as natural language processing, speech recognition, and time series forecasting [29, 30].

2.2.1.2 Vanishing Gradient Problem

Training RNNs uses Backpropagation Through Time (BPTT), where the network is unfolded across time, and gradients are propagated backward through each step. Because this process repeatedly multiplies many small or large terms, the gradient can quickly become extremely small (vanishing) or extremely large (exploding). This problem is called vanishing gradient, and as shown by Bengio et al. [31], and Pascanu et al. [32], this makes it difficult for standard RNNs to learn long-term dependencies, since information from distant time steps has almost no influence on learning.

In practice, this means that RNNs mainly capture short-term patterns and struggle with long-range temporal structure. This limitation motivated the development of gated architectures such as LSTMs [33], which help maintain more stable gradients over time.

2.2.2 Long Short-Term Memory (LSTM) Neural Network

LSTM networks are a type of RNN designed to capture long-range dependencies in sequential data [28]. As discussed in Section 2.2.1.2, standard RNNs struggle with vanishing and exploding gradients, making it difficult for them to learn from distant time steps. To address this, Hochreiter and Schmidhuber [33] introduced the LSTM architecture, which uses special memory cells to maintain a more stable flow of information. This allows LSTMs to learn relationships that span long sequences, overcoming the main limitations of conventional RNNs.

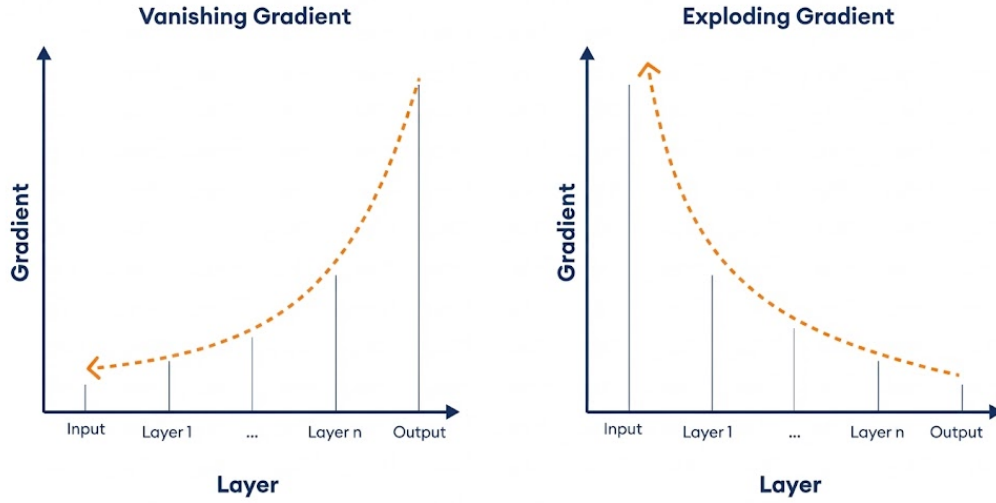


Figure 2.3: Vanishing and Exploding gradients: A major challenge in training Recurrent Neural Networks.

2.2.2.1 LSTM Architecture and Mathematical Foundations

As shown in Figure 2.4, an LSTM network is composed of memory cells connected in a recurrent chain. Each cell maintains two internal components at time t : a cell state C_t , which acts as long-term memory, and a hidden state h_t , which represents the output passed to the next time step.

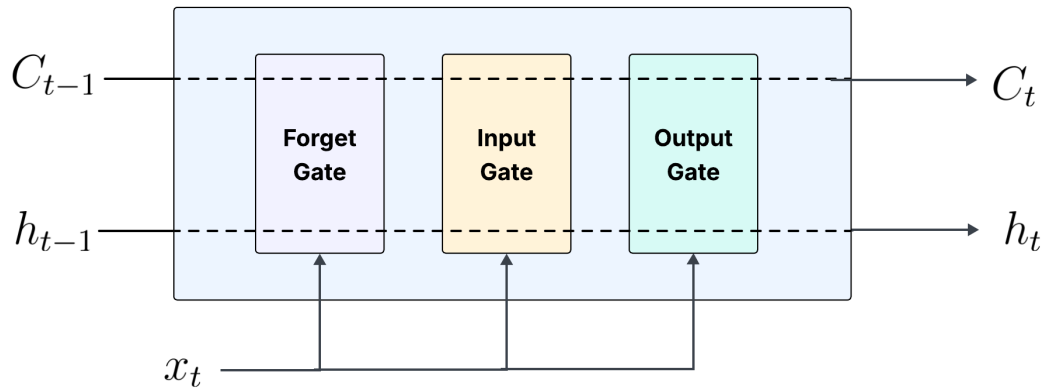


Figure 2.4: Architecture of the Long Short-Term Memory (LSTM) cell.

The key innovation of LSTM is a set of gates that control the flow of information through each cell. In an LSTM cell, information flows through three gates that regulate how the previous hidden state h_{t-1} , previous cell state C_{t-1} , and current input x_t are combined to produce the updated states:

- **Forget gate:** uses h_{t-1} and x_t to determine which parts of the previous cell state C_{t-1} should be retained or discarded.
- **Input gate:** processes h_{t-1} and x_t to control how much new information is written into the cell state, working together with a candidate update to incorporate relevant features from the current input.
- **Output gate:** combines the updated cell state C_t with h_{t-1} and x_t to compute the new hidden state h_t , determining which parts of the internal memory are exposed to the next layer or time step.

In addition to these gates, the LSTM cell computes a candidate update that encodes new information potentially relevant to the memory. The forget and input gates jointly determine how the previous cell state and the candidate update are combined to form the new cell state. The hidden state is then obtained by filtering this updated memory through the output gate.

Together, these mechanisms allow the LSTM to preserve information over long sequences. The cell carries relevant information forward while the gates learn to add or remove content as needed. This design enables LSTM to maintain stable gradients across many time steps, effectively addressing the vanishing gradient problem that limits standard RNN to process long-term sequences [33, 28].

2.2.3 Convolutional Neural Network (CNN)

CNNs are DL architectures designed to leverage the spatial structure of grid-like data, such as images or organized meteorological fields [34].

2.2.3.1 Architectural Foundations of CNN

The architecture of CNN, as shown in Figure 2.5, consists of three fundamental layers that work together to transform raw input into high-level abstract features. This layered structure enables the network to learn hierarchical features: the initial layers detect local patterns, such as edges and textures. In contrast, the deeper layers capture more complex shapes and semantic information.

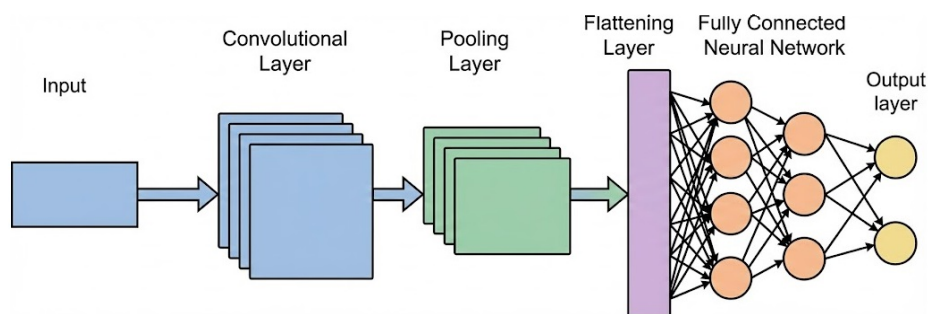


Figure 2.5: Architecture of the Convolutional Neural Network (CNN).

- **Convolutional Layers:** The core building blocks of the network are convolutional layers. These layers use a set of learnable kernels, or filters, to perform convolution operations on the input data. Unlike dense layers, which connect every input to every output, convolutional layers leverage local connectivity and weight sharing. This approach enables the network to detect the same physical feature, such as a specific moisture trend, regardless of its position within the look-back window.
- **Pooling Layers:** After convolution, pooling layers, such as Max Pooling, are used to reduce the spatial resolution of the feature maps. This down-sampling process reduces data size, lowers computational load, and reduces the risk of overfitting while providing the model with translation invariance. As a result, the feature detection becomes robust against slight temporal shifts.
- **Fully Connected Layers:** The final stage of the architecture flattens the multidimensional feature maps into a single vector. This vector is then sent through dense layers, which convert the extracted features into the final output.

2.2.3.2 Comparative Analysis: CNN vs. Recurrent Architectures

The choice between CNNs and recurrent models (RNNs/LSTMs) involves fundamental differences in how information is processed and how temporal dependencies are captured.

The most significant distinction lies in the processing mechanism. RNNs and LSTMs are inherently sequential; they process inputs one time step at a time, maintaining an internal state that carries information forward. While this allows them to model long-term dependencies, it slows down training. In contrast, CNNs process all elements within their receptive field simultaneously through parallelization. This makes CNNs more computationally efficient and faster to train on modern hardware, as they do not have to wait for the previous time step's computation to complete.

Furthermore, the two architectures differ in their memory focus. A CNN has a finite memory limited by the size of its filters; it is excellent at identifying local, short-term motifs within a specific window of time. LSTMs, however, possess a flexible memory governed by gating mechanisms. This allows an LSTM to decide which information from steps ago is still relevant, whereas a standard CNN would only see what is currently within its sliding filter.

2.2.4 Transformer Models and Large Language Models (LLMs)

The Transformer is an ANN architecture centered on the multi-head attention mechanism. The Transformer architecture represents a significant shift in the design of sequence models. These models were initially proposed for Natural Language Processing (NLP) tasks [25], and have since evolved into a versatile framework for modeling sequential, structured, and multimodal data. The key innovation of the Transformer is its attention mechanism, which enables it to model long-range dependencies without recurrent connections. This enables highly parallel computation and improved scalability compared to traditional architectures, such as RNNs and LSTMs.

2.2.4.1 Core Architectural Components of Transformers

A standard Transformer consists of stacked encoder and decoder blocks, each containing several key components that gradually transform raw input into contextualized representations.

- **Tokenizer:** The first stage involves converting raw input into discrete units known as tokens. In NLP, these tokens can represent either words or subwords. In time series applications, tokens may correspond to temporal segments or to discretized numerical values. Tokenization provides a structured representation that enables the model to process information effectively.
- **Embedding Layer:** Tokens are converted into dense numerical vectors using an embedding layer. These embeddings can be static or dynamically updated as they pass through deeper layers. Since Transformers do not naturally account for sequence order, positional encodings are added to the embeddings to provide temporal or structural context.
- **Attention Mechanism:** The attention mechanism is the key innovation of the Transformer model. It calculates relevance scores for all token pairs, enabling the model to selectively focus on the most informative parts of the sequence. This process involves learned projections for queries, keys, and values that produce weighted combinations of token representations.
- **Multi-Head Attention:** Instead of calculating a single attention distribution, the Transformer utilizes multiple attention heads in parallel. Each head captures various types of relationships, local, global, semantic, or structural, allowing the model to learn richer and more diverse patterns.

- **Feedforward Layers:** In each Transformer block, the attention module is followed by a position-wise feedforward network. These layers refine the contextualized representations and introduce additional nonlinearity. Residual connections and layer normalization help stabilize training and support deep layer stacking.

Transformer models, originally designed for NLP, have also shown capabilities in time series forecasting [35, 36]. Traditional recurrent models, such as RNNs and even more advanced variants like LSTMs, can face limitations when modeling very long sequences due to their inherently sequential processing and restricted effective receptive field. In contrast, Transformer architectures can attend to any position in the input sequence directly, enabling more flexible and scalable temporal modeling.

The Transformer family has inspired many variations aimed at improving computational efficiency and simplifying the architecture. Among these, DLinear [37] stands out as a lightweight baseline that demonstrates how strong forecasting performance can be achieved with minimal architectural complexity. Instead of relying on attention mechanisms, it uses simple linear operations applied to decomposed time series components, offering an efficient alternative for comparison. This highlights the flexibility of Transformer-based design principles. These innovations naturally extend to LLMs, which scale Transformer architectures to much larger capacities and leverage extensive pretraining to develop general reasoning and pattern-recognition abilities.

2.2.4.2 Large Language Models (LLMs)

LLMs represent the next stage in the evolution of Transformer architectures. By increasing the scale of parameters, the size of the training data, and the depth of Transformer blocks, LLMs gain impressive capabilities in reasoning, pattern recognition, and generalization. These models are trained on vast datasets and learn broad semantic and structural patterns that can be applied across various tasks.

Recently, LLMs have been adapted for time series analysis. Rather than training a Transformer model from scratch, approaches like Time-LLM [38] reformat numerical sequences into text-like prototypes. This allows pretrained LLMs to interpret temporal patterns, such as “a short rise followed by a decline”, without necessitating complete retraining. This methodology leverages the generalization capabilities of large Transformers to extract semantic structures from numerical data, creating new opportunities for hybrid modeling across forecasting, anomaly detection, and decision-making.

2.3 Reinforcement Learning (RL)

Reinforcement Learning (RL) is a field of ML concerned with how an agent should act in a dynamic environment to maximize a reward signal [39]. At its core, RL addresses closed-loop decision-making problems in which the agent's actions influence the future states and rewards it encounters.

The key features of RL include its feedback-driven approach, the lack of explicit action labels, and the focus on the long-term consequences of decisions. The primary goal is to model how an agent interacts with its environment to achieve a specific objective. This process involves perceiving the current state, selecting actions to alter that state, and adjusting behavior based on the outcomes defined by the environment (Figure 2.6).

A central challenge in RL, unlike in other forms of learning, is the balance between exploration and exploitation. To maximize rewards, an agent must exploit the actions it already knows to be effective. However, it must also explore new actions to discover potentially better strategies. If the agent relies solely on exploitation, it cannot improve, while relying only on exploration prevents it from consistently earning rewards. The solution lies in combining both approaches: the agent should try a variety of actions

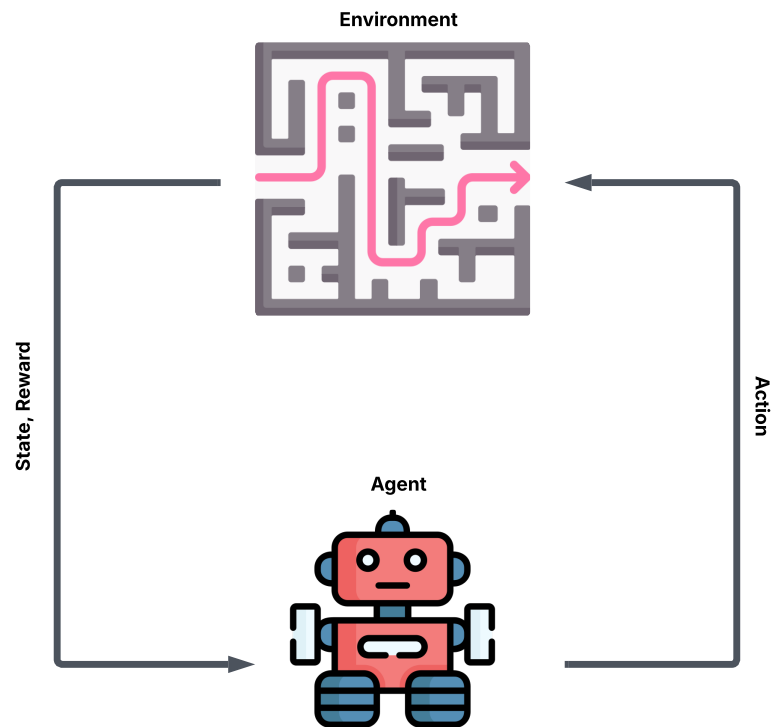


Figure 2.6: Schematic representation of the Reinforcement Learning framework.

while gradually favoring those that seem to yield the highest returns. In stochastic environments, actions may need to be repeated multiple times to accurately estimate their expected rewards. This challenge of balancing exploration and exploitation has been extensively studied in mathematics. However, it remains a fundamental issue specific to RL and does not occur in the pure forms of learning.

2.3.1 Key Elements of Reinforcement Learning

Beyond the core interaction between the agent and the environment, as represented in [40] an RL system is defined by four primary components:

- **Policy (π):** A policy represents the logic or strategy of an agent at a specific moment. Formally, it can be defined as a mapping from the perceived states of the environment to the actions that should be taken in those states (denoted as $\pi(a|s)$). Policies can vary significantly, ranging from simple lookup tables to complex NNs. They may be deterministic, meaning that a given state always results in the same action, or stochastic, in which the policy defines a probability distribution over possible actions. The policy is the central component of the system since it determines the observable behavior of the agent.
- **Reward Signal (r):** At each discrete time step, the environment provides the agent with a numerical value called a reward. The agent's objective is to maximize its cumulative reward over the long term. While the agent cannot change the reward function itself, it can influence the rewards it receives by adjusting its actions. The reward signal is the primary mechanism for adjusting the agent's policy; actions that lead to low rewards encourage the agent to modify its policy in favor of more beneficial actions in the future.

- **Value Function (V or Q):** While the reward signal indicates immediate success, the value function represents long-term desirability. The value of a state ($V(s)$) is the total amount of reward an agent can expect to accumulate in the future, starting from that state. In many cases, a state might yield a low immediate reward but possess a high value because it is a necessary precursor to high-reward states. Estimating values is significantly more challenging than receiving rewards; whereas rewards are provided directly by the environment, values must be iteratively estimated from the agent's lifetime of experience.
- **Environment Model:** This allows the agent to make inferences about how the environment will behave. Given a specific state and action, a model predicts the subsequent state and reward. Models are essential for planning, allowing the agent to consider potential future scenarios before they are physically experienced. Systems that incorporate models are termed model-based methods, whereas those that rely strictly on trial-and-error interaction are classified as model-free.

2.3.2 Practical Applications and Illustrative Examples of RL

The versatility of RL is clearly demonstrated by its successful applications across a range of fields, from strategic games to complex industrial systems and autonomous technologies. These examples highlight RL's capability to manage long-term planning, handle uncertainty, and adapt to changing circumstances.

- **Strategic Game Playing:** In games like Chess, a master player (the agent) selects moves by balancing immediate tactical advantages with long-term strategic planning, anticipating the opponent's counter-moves, and evaluating the strength of potential future positions [40].
- **Robotic Navigation and Resource Management:** A mobile robot must make sequential decisions, such as whether to continue a task (e.g., searching for objects) or return to a docking station based on fluctuating battery levels and its internal map of the environment.
- **Industrial Energy Optimization:** Google DeepMind utilized RL to optimize cooling systems in data centers, resulting in a 40% reduction in energy consumption. The agent autonomously adjusts cooling strategies in response to real-time sensor data while considering safety constraints [41].
- **Adaptive Manufacturing:** RL-based robotic arms are employed in assembly lines for tasks such as welding and precision assembly. Unlike traditional pre-programmed systems, these agents adapt to variations in material positioning or hardware wear, minimizing downtime and improving operational efficiency [42].
- **Autonomous Vehicle Control:** In self-driving systems, RL is used to derive optimal driving policies, including lane-change maneuvers and obstacle avoidance, by learning from both simulated environments and real-world feedback [43].
- **Quantitative Finance:** In portfolio management, RL models dynamically re-balance investment strategies based on market volatility and feedback to maximize long-term returns while mitigating risk [44].

Despite their differences, these scenarios share fundamental characteristics: an active decision-making agent interacts with an uncertain environment to achieve a specific goal. The agent's actions, whether it's a move on a chessboard or a change to a cooling valve, affect the environment's future state. Therefore, effective decision-making requires the agent to anticipate delayed consequences and account for the system's inherent unpredictability.

In addition, the boundary between the agent and the environment is often different from what we might intuitively expect. The agent includes the decision-making process, while the environment consists of everything external to that process, including the system's internal hardware states, such as a robot's battery level or memory. In all these scenarios, interaction is the key to learning, enabling the agent to refine its behavior through experience over time.

2.3.3 Deep Q-Learning (DQL)

DQL enhances classical RL techniques to handle environments with large or high-dimensional state spaces by utilizing Deep Q-Network (DQN) to approximate the action-value function. Instead of storing a specific value for each state-action pair, the DQN learns a parametric function that maps raw observations, such as images, sensor readings, or other complex features, to estimates of action quality. This method allows agents to function effectively in environments where traditional tabular methods would be computationally impractical. Additionally, the use of experience replay and target networks helps stabilize the learning process, making DQL applicable in complex, real-world settings [45].

2.3.3.1 Basic Q-Learning

As introduced in Section 2.3, RL agents acquire optimal behavior through repeated interaction with a stochastic environment. Q-learning is a foundational value-based method that seeks to learn the optimal action-value function $Q^*(s, a)$, defined as the maximum expected discounted return obtained by taking action a in state s and subsequently following the optimal policy π^* :

$$Q^*(s, a) = \mathbb{E} \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} \mid S_t = s, A_t = a, \pi^* \right] \quad (2.4)$$

The Bellman optimality equation, derived from Bellman's principle of optimality, expresses this relationship recursively by decomposing the expected return into immediate reward and the optimal future value:

$$Q^*(s, a) = \mathbb{E}_{s' \sim P} \left[r + \gamma \max_{a'} Q^*(s', a') \mid s, a \right] \quad (2.5)$$

In the tabular setting, Q-learning updates the value of each state-action pair using the temporal-difference rule:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right] \quad (2.6)$$

Here, s and a denote the current state and action, s' is the next state sampled from the environment's transition dynamics, and r is the immediate reward. The discount factor $\gamma \in [0, 1]$ controls the importance of future rewards, while α is the learning rate used in the temporal-difference update.

Q-learning is model-free and off-policy: it does not require knowledge of the transition dynamics and can learn an optimal policy independently of the behavior policy used to collect data [46].

2.3.3.2 Limitations of Tabular Q-Learning

Despite its theoretical guarantees, tabular Q-learning faces several limitations in practical applications:

- **Scalability:** In environments with large or continuous state spaces, the Q-table grows exponentially, making storage and exploration infeasible.
- **Lack of Generalization:** Tabular methods treat each state independently and cannot generalize across similar states, resulting in poor sample efficiency [47].

To scale Q-learning to high-dimensional domains, the Q-table is often replaced with a function approximator such as an NN. However, this introduces instability. Because the agent collects data sequentially, consecutive samples are highly correlated, causing the network to overfit recent experiences and forget earlier ones. Moreover, the same network is used both to estimate current Q-values and to compute learning targets, creating a “moving target” problem in which the learning objective changes as the network updates [45]. As an example, consider a mobile robot learning to navigate toward a charging station. Small changes in the robot’s position can lead to large changes in the estimated value of an action, causing the robot to alternate unpredictably between turning left and right rather than moving smoothly toward the goal. These effects can lead to oscillations in the learned behavior or even complete failure to converge.

The DQL framework addresses these challenges by combining Q-learning with deep NNs and introducing mechanisms that stabilize training [48].

2.3.3.3 The Deep Q-Learning Framework and Deep Q-Network Architecture

In the DQL framework, the Q-function is approximated by an NN, $Q(s, a; \theta)$, which outputs the estimated value of each action given a state. Stability is achieved through three key components:

- **Experience Replay:** Transitions (s, a, r, s') representing a single interaction step in the environment, consisting of the current state s , the action a taken by the agent, the reward r received, and the next state s' reached as a result, are stored in a replay buffer D , and training is performed on mini-batches sampled uniformly from this buffer:

$$(s, a, r, s') \sim U(D)$$

This breaks temporal correlations, improves sample efficiency, and allows the agent to reuse past experiences for multiple updates [45].

- **Target Network:** To mitigate the moving target problem, DQN maintains a separate target network with parameters θ^- . The online network with parameters θ is updated at every step, while the target network is updated only periodically. The loss function minimized by the online network is:

$$L(\theta) = \mathbb{E}_{(s,a,r,s') \sim D} [(y^{\text{DQL}} - Q(s, a; \theta))^2] \quad (2.7)$$

where the target value is computed using the frozen target network:

$$y^{\text{DQL}} = r + \gamma \max_{a'} Q(s', a'; \theta^-) \quad (2.8)$$

This separation provides a stable learning target and significantly improves convergence.

- **Exploration via ϵ -Greedy:** DQN balances exploration and exploitation using an ϵ -greedy policy:

$$\pi(a | s) = \begin{cases} \text{random action,} & \text{with probability } \epsilon, \\ \arg \max_a Q(s, a; \theta), & \text{with probability } 1 - \epsilon. \end{cases} \quad (2.9)$$

The value of ϵ is gradually annealed to encourage exploration early in training and exploitation later.

Together, these components enable DQN to learn stable and effective policies from high-dimensional sensory inputs, marking a major advancement in DRL.

While DQL and its extensions provide a stable framework for discrete action spaces, they encounter substantial challenges in more complex environments [49]. In particular, the requirement to compute the action maximization term $\max_a Q(s, a)$ becomes increasingly difficult as the action space grows. This limitation is especially pronounced in continuous control tasks, high-dimensional action spaces, and settings with multi-objective reward structures, where value-based maximization becomes computationally inefficient and unstable.

These limitations motivated a shift toward Policy Gradient and Actor-Critic architectures. Unlike value-based methods, these algorithms directly optimize a parameterized policy, allowing for smoother updates and the natural integration of stochastic behavior [50]. Among these, PPO [50] has emerged as a robust standard due to its clipped surrogate objective, which prevents destructively large policy updates while maintaining sample efficiency. The following section details the PPO framework and outlines its mechanism for balancing stability in complex decision-making tasks.

2.3.4 Proximal Policy Optimization (PPO)

PPO is a widely used DRL algorithm that improves a policy through repeated interactions with the environment. It trains a stochastic policy, denoted π_θ , that allows the agent to sample actions and observe their outcomes. During each training cycle, the agent collects new trajectories, sequences of states, actions, and rewards, using the current policy. These trajectories are then used to update the policy to maximize the expected return. As the agent collects more interactions over time, the policy shifts from being exploratory to becoming more precise in selecting high-reward actions.

PPO is a first-order optimization method that balances simplicity, stability, and performance. Unlike off-policy value-based methods like DQL, which can exhibit instability and divergence when combined with NNs, or standard policy gradient methods, which often suffer from high variance, PPO avoids large, destabilizing policy updates by introducing a clipping surrogate objective function. In particular, a clipping region is used to restrict the difference between the new and old policy to a small, controlled range. This proximal update ensures that learning remains stable and does not diverge, which is especially beneficial in complex environments or when training data is limited.

PPO is implemented with an actor-critic architecture. The actor network (or policy network) selects actions to take, while the critic network (or value network) estimates state values to compute the goodness of the choices taken. Figure 2.7 shows the architecture of the PPO algorithm using an Actor-Critic approach. As illustrated in the figure, the system operates in a continuous loop to refine both the actor's decisions and the critic's evaluations. This framework includes the following main components:

1. *Environment interaction and experience collection*: Given a modeled environment E , the agent selects an action a_t at time t and receives from E an “experience tuple” $\langle s_t, a_t, r_t, s_{t+1} \rangle$. Here, s_t is the current state, a_t the action taken, r_t the reward obtained for executing a_t in s_t , and s_{t+1} the resulting next state. Repeating this interaction over multiple time steps produces a sequence of such tuples, forming a trajectory of experiences.
2. *Rollout Storage (Trajectory Memory)*: The sequence of collected experience tuples is temporarily stored in a component called “Rollout Storage”. This component serves as a buffer for the sequence of experiences generated by the agent's current policy π_θ . It is an essential element of the PPO because the algorithm uses this collected data for multiple learning updates before discarding them and collecting a new trajectory.

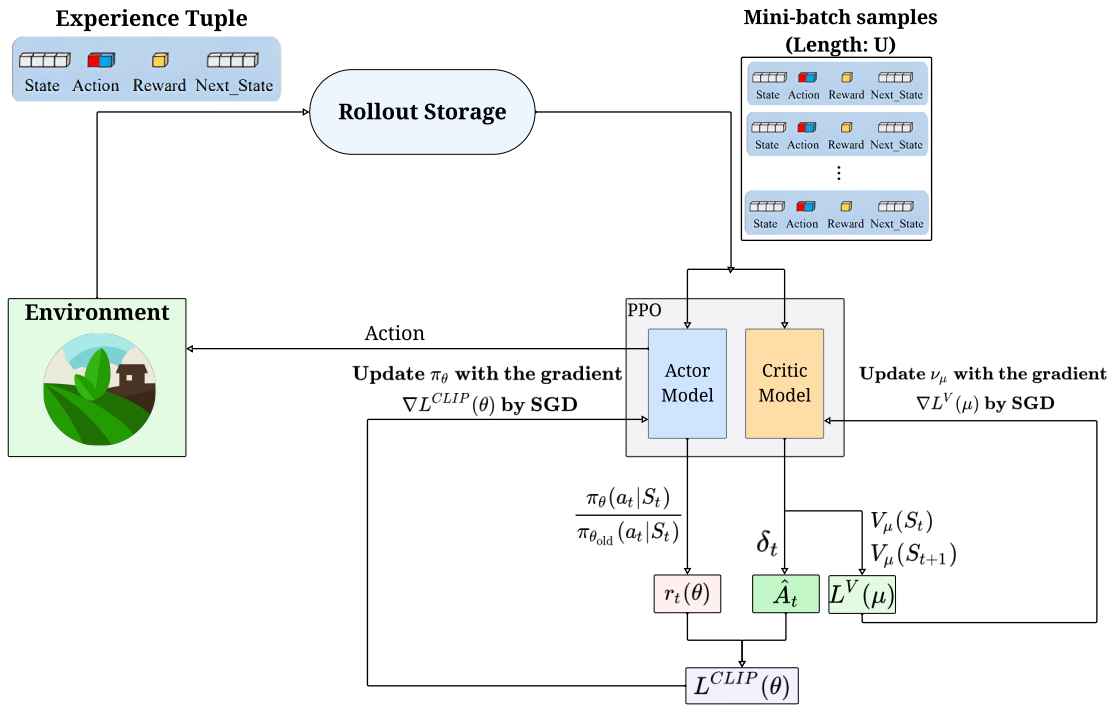


Figure 2.7: Exemplification of the Proximal Policy Optimization functioning, inspired by [1].

3. PPO Components: Actor and Critic Models:

The PPO block contains the two core learning models:

- **Actor model (or policy model):** It can be considered as the agent's decision-maker. It receives information about the current state s_t and decides which action a_t to take. Its goal is to develop a stochastic policy $\pi_\theta(a_t|s_t)$, which is essentially a set of rules or probabilities that dictate the best action for any given state. Indeed, given the state s_t as input, it produces the probabilities of taking each action a_t . The parameter θ represents the knowledge it has learned so far.
- **Critic model (or value model):** It acts as the agent's evaluator. It assesses how good a particular state s_t is. It learns to estimate the expected future reward ($V_\mu(s_t)$) achievable from the current state s_t . The parameter μ represents the learned evaluation knowledge.

4. **Data Preparation for Learning:** Once an iteration has been completed, the two models can be updated. This update exploits the data collected inside the "Rollout storage" component and requires the computation of the following parameters:

- **Policy ratio $r_t(\theta)$:** For each action a_t , the algorithm calculates the probability of taking that action under the current policy $\pi_\theta(a_t|s_t)$ and compares it to the probability under the old policy $\pi_{\theta_{\text{old}}}(a_t|s_t)$. This comparison produces the "importance sampling ratio ($r_t(\theta)$)", defined as:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \quad (2.10)$$

This ratio is a key factor in PPO stability, ensuring that updates do not deviate too far from the old policy.

- **Temporal Difference (TD) error δ_t :** The estimates done by the critic network are used to calculate the TD error, which measures the difference between the expected return and the actual value

observed. It is typically calculated as:

$$\delta_t = r_t + \gamma \cdot V_\mu(s_{t+1}) - V_\mu(s_t) \quad (2.11)$$

where γ is the discount factor used to modulate the role of future rewards with respect to the current one.

- *Advantage estimation $\hat{A}_t$* : Using the TD errors, the “Generalized Advantage Estimation (GAE)” is calculated, which provides a more accurate measure of how much better a chosen action was compared to the average expected outcome from that state. It is a crucial input for updating the actor model.

5. *Model Updates (Optimization)*: The prepared data is then used to update both the actor and critic models. A batch of samples, referred to as “Mini-batch Samples (length: U)”, is drawn from the “Rollout Storage” for these updates.

- *Critic Model Update*: The objective of the critic model is to minimize its value Loss, denoted as $L^V(\mu)$. This loss measures the squared difference between the predicted values ($V_\mu(s_t)$) and the actual returns. The parameter μ characterizing this network is then updated using its gradient, $\nabla L^V(\mu)$, typically through an optimization algorithm like SGD.
- *Actor Model Update*: The objective of the actor model is to minimize its *Clipped Policy Loss*, denoted as $L^{\text{CLIP}}(\theta)$. This is the core of PPO. It uses the “Importance Sampling Ratio” $r_t(\theta)$ and the “Advantage Estimation” $\hat{A}_t$ to construct a loss function that encourages beneficial actions while preventing large, destabilizing changes to the policy. In particular, the clipping mechanism within this loss computation ensures that the ratio $r_t(\theta)$ always stays within a certain range. The simplified policy loss can be seen as:

$$L^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t \right) \right] \quad (2.12)$$

where ϵ is the clipping hyperparameter. The parameter θ characterizing this network is updated using its gradient, $\nabla L^{\text{CLIP}}(\theta)$, typically through an optimization algorithm like SGD. An entropy bonus (not explicitly shown in the formula but often added) is also normally included to encourage exploration. This expectation ensures that the update reflects the average improvement across the collected experience.

This cycle of interaction, experience collection, data preparation, and model updates is repeated, allowing the agent to continuously learn and improve its performance in the environment.

2.4 Quantitative Metrics for Model Evaluation

Accurate evaluation of predictive models requires quantitative metrics that capture different aspects of error behavior. In this thesis, these metrics are introduced as part of the methodological background to ensure consistent assessment of models performance across experiments. The following measures are used throughout the thesis to evaluate prediction accuracy, robustness, and sensitivity to scale. In most experiments, these metrics serve as the primary basis for evaluating model performance; when additional or task-specific metrics are required, they are introduced and justified within the corresponding experimental section.

2.4.1 Mean Absolute Error (MAE)

The Mean Absolute Error (MAE) measures the average magnitude of the errors between the predicted and observed values, without considering their direction. It is defined as:

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i| \quad (2.13)$$

MAE is intuitive and expressed in the same units as the original variable, making it easy to interpret. It penalizes all deviations linearly, treating all errors equally.

2.4.2 Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error (MAPE) expresses prediction accuracy as a percentage and is defined as:

$$\text{MAPE} = \frac{100}{N} \sum_{i=1}^N \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (2.14)$$

It measures the relative error between predictions and observations. Since it is normalized by the actual value, MAPE is useful for comparing performance across datasets with different scales. However, it is sensitive to small true values (i.e., when $y_i \approx 0$), leading to disproportionately large percentage errors.

2.4.3 Mean Squared Error (MSE)

The MSE quantifies the average squared difference between predictions and ground truth:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (2.15)$$

Because the errors are squared before averaging, MSE penalizes larger deviations more strongly than MAE, making it useful when large errors are particularly undesirable.

2.4.4 Root Mean Squared Error (RMSE)

The Root Mean Square Error (RMSE) measures the average magnitude of prediction errors, expressed in the same units as the target variable. Because it is derived from squared residuals, it penalizes larger deviations more strongly:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}. \quad (2.16)$$

2.4.5 Coefficient of Determination (R^2)

The coefficient of determination, R^2 , quantifies the proportion of variance in the observed data that the model explains. Higher values indicate better explanatory power:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}. \quad (2.17)$$

Using all three metrics ensures that performance differences across methods are assessed from complementary perspectives, covering both absolute and relative error behavior.

Chapter 3

Virtual Sensing and Modeling of Greenhouse Temperature

Chapter contents

3.1	Introduction	26
3.2	Related Work	26
3.3	Problem Formalization and Proposed Solution	28
3.3.1	Model Architecture	29
3.4	Application Setup	34
3.4.1	Sensing Infrastructure and Data Collection	34
3.4.2	Evaluation Metrics	35
3.5	Experimental Results.....	36
3.5.1	Experiments Result of Hybrid LSTM and CNN Method	36
3.5.2	Experiments result of Time-LLM framework	37
3.6	Discussion and Comparative Analysis	41
3.6.1	Analysis of Predictive Accuracy	41
3.6.2	Computational Efficiency and Deployment Trade-offs	42
3.7	Conclusion.....	43

3.1 Introduction

Accurate climate mapping is essential for modern agriculture. Spatially precise information about greenhouse conditions supports agronomic decisions that directly influence plant health and growth [16]. Although greenhouses are typically “controlled” environments, their internal climates are far from uniform. Factors such as airflow patterns, sunlight distribution, and plant density create distinct microclimates in different areas of the structure. Ignoring these variations can lead to uneven growth and increased risk of plant disease. From the other side, capturing these spatial differences typically requires a dense network of physical sensors. However, installing and maintaining many devices can be costly and challenging due to wiring constraints, battery limitations, and the harsh, humid greenhouse environment.

To address these limitations, this research adopts virtual sensing, also known as “soft sensing”. A virtual sensor is a software model that uses data from a small set of permanent sensors to estimate the climate at other places where no physical sensors are installed. This chapter presents a three-tier progression of virtual sensing methods to improve microclimate mapping while reducing hardware requirements. The methods presented in this chapter are intentionally data-driven. They establish the environmental perception layer of the DT, providing a baseline that motivates the hybrid extensions introduced in Chapters 5 and 6.

After introducing these methods, we compare them using a common real-world dataset, evaluating predictive accuracy, computational complexity, and memory footprint to ensure a fair and practical assessment.

The methodology follows a clear evolution in both model complexity and deployment target. We begin with standard RNNs to establish a simple baseline. Then, we introduce a Hybrid CNN-LSTM model designed to run directly on small, low-power devices inside the greenhouse, where computational resources and energy are limited. Finally, we explore the upper bound of achievable performance using Time-LLM, a framework that adapts LLMs for time-series prediction. This three-level approach allows us to compare accuracy against the hardware constraints typical of smart-agriculture environments.

3.2 Related Work

Assuming uniform climate conditions within a greenhouse is unrealistic [16]. This simplification can lead to inaccurate predictions of plant growth and inefficient resource management, underscoring the need to understand the spatial and temporal variations in greenhouse microclimates. This issue was first addressed by Jolliet et al. (1994), who focused on optimizing greenhouse humidity and transpiration [51]. In this context, Van Pee et al. (1998) modeled climate gradients around plants, revealing interactions within greenhouse microclimates [52]. These studies emphasize the complexity of internal climate variations and the necessity of precise climate characterization for effective greenhouse management.

To illustrate the complexities associated with internal climate variations in greenhouses, Zhao et al. (2001) explored vertical temperature and humidity gradients in naturally ventilated greenhouses [53]. Boulard et al. (2002) emphasized the importance of considering temperature, humidity, and CO₂ gradients arising from fan ventilation [54]. Building on this exploration, Kittas et al. (2003) investigated temperature gradients in shaded greenhouses [55]. This was followed by Chen’s development of a mathematical model for predicting temperature and humidity gradients in evaporatively cooled greenhouses [56].

Collectively, these studies demonstrate that addressing spatial heterogeneity is essential for optimizing greenhouse climate control and improving crop performance. Further studies on challenges posed by spatial heterogeneity were conducted [57], and the relevance of considering specific climatic gradients in crop growth modeling was discussed in another study [58].

To manage complex, nonlinear microclimates, advanced climate control strategies have been developed. In the realm of greenhouse climate regulation, various data-driven methods aim to manage the nonlinear dynamics of microclimates while addressing constraints and disturbances inherent in greenhouse environments. For instance, the methodology outlined in Pinon et al. [59] integrates Feedback Linearization (FL) and Model Predictive Control (MPC) to address the challenges posed by the nonlinear dynamics of greenhouse temperatures. Additionally, the approaches discussed in Speetjens et al.'s study [60] and Hameed et al.'s work [61] apply the Kalman filter and the extended Kalman filter for greenhouse climate control.

The effective implementation of these control strategies requires accurate, real-time monitoring of greenhouse conditions, enabled by advances in sensing technology. Technology enables researchers to measure and control climate variation in greenhouses. For instance, by using various sensors, it is possible to monitor different parameters inside greenhouses, including air temperature and humidity [62], soil moisture [63], soil irrigation volume, light intensity, and carbon dioxide concentration [64].

Wireless Sensor Networks (WSN)s have emerged as powerful tools for capturing spatial and temporal variations, enabling real-time monitoring and control of greenhouse environments. In this context, WSN offer a helpful framework that allows farmers and researchers to monitor real-time data accurately [65]. For instance, in [66], a dense WSN was used for greenhouse climate control. Simulations on monitoring and controlling greenhouse climate using WSNs were conducted in [67]. Building on this, in [68], the authors studied the application of WSNs for controlling greenhouse parameters in precision agriculture. In 2015, a study conducted in a Greek greenhouse demonstrated the capabilities of WSNs in achieving spatially distributed climate control. In this regard, [69] proposed a hierarchical WSN system for greenhouse monitoring that combines Internet technology and intelligent object communication models.

Although WSNs have been widely used in greenhouse applications, they still have several limitations. High deployment and maintenance costs, power supply issues, and vulnerability to harsh greenhouse conditions can limit the effectiveness of dense sensor networks. This has increased interest in virtual sensing [70].

Virtual sensing provides a cost-effective alternative by estimating conditions at unmeasured locations using data from existing sensors. This approach offers a cost-effective alternative in scenarios where direct physical measurements are difficult or impractical [71]. Virtual sensors rely on software algorithms that process data from existing hardware sensors to estimate real-time values at specific points of interest [72].

Virtual sensors have several applications in various engineering fields. In gas sensing systems, they were employed to improve performance [73]. In automotive systems, they play an integral role in vehicle control [74], while in mobile robot applications, they enhance adaptability and functionality [75]. Aircraft engines also incorporate virtual sensors to detect faults in their complex nonlinear systems, ensuring safety and reliability [76]. In greenhouse monitoring, although still limited, virtual sensing has been explored as a means to overcome the practical constraints of physical sensor networks [77].

In general, virtual sensing approaches can be divided into two main categories of model-based and data-driven methods [78]. Model-based virtual sensors exploit system knowledge described through equations [79], while data-driven virtual sensors rely on historical measurements, which serve as input for statistical or machine learning methods [80, 81].

A typical example of a model-based approach is the use of CFD methods to model fluid flows in greenhouses [82, 83, 84]. Studies in this area have focused on CFD simulations for analyzing experimental data [85], forecasting temperature distribution [86], creating dynamic models for greenhouses [87], and investigating thermal balance in greenhouse air conditioning systems [88]. Recent advancements have combined CFD with machine learning to enhance prediction accuracy, enabling the development of virtual

sensor systems for real-time greenhouse monitoring [89, 90]. However, this approach remains challenging due to its complexity, high computational requirements, and sensitivity to disturbances in greenhouse environments [84].

Unlike the model-based approach, data-driven virtual sensors estimate physical conditions using available data, statistical methods, and state-of-the-art machine learning techniques. These approaches offer flexibility and scalability, particularly for capturing the dynamic and spatially distributed nature of greenhouse microclimates. For instance, the relationship between temporary and permanent sensors, considering both time and spatial context, is captured by linear regression in [91] and by a trained RNN in [18]. The method presented in [92] uses RL to manage sensor states. In this method, the agent learns from LSTM-based predictions to determine the best times to activate or deactivate sensors, aiming to save energy while maintaining accuracy.

3.3 Problem Formalization and Proposed Solution

This chapter focuses on predicting the microclimatic conditions at specific points within a greenhouse. The task is approached by learning from context-rich historical time-series data that capture both environmental dynamics and spatial-temporal variability. The problem is formalized by introducing the key concepts of POI, temporal domain, context, contextual measurement, and time series, which together define the data structure used for microclimate prediction.

Definition 3.1 (Points of Interest (POIs)) *In this concept, a POI is a specific location within the greenhouse where microclimate information is relevant to agricultural purposes. POIs serve as the spatial targets for which virtual sensors are expected to provide predictions. Temporary sensors are installed at these locations only during the data collection phase. After this phase, the trained model generates virtual estimates to replace the temporary sensors.*

Definition 3.2 (Temporal Domain) *A temporal domain $\mathbb{T}$ is a tuple $(T, \leq)$ where T is a set of not-empty temporal instants and $\leq$ is an ordering relation on T [93].*

In other words, a temporal domain $\mathbb{T}$ is a set of totally ordered time instants. Given that, granularity is defined as follows.

Definition 3.3 (Temporal granularity) *Given a temporal domain $\mathbb{T}$, the temporal granularity is a mapping G from the integers $\mathbb{Z}$ (set of indexes) to a subset of $\mathbb{T}$ such that:*

1. *If $i < j$ and $G(i)$ and $G(j)$ are not empty, then the elements of $G(i)$ are less than all the elements of $G(j)$.*
2. *If $i < k < j$ and $G(i)$ and $G(j)$ are not empty, then also $G(k)$ is not empty.*

In other words, the granularity defines a partition of the temporal domain into indivisible units, called *granules*.

Time granularity is the resolution of a statement's temporal qualification. In this study, we considered only discrete temporal domains, with a minimum granularity of 1 minute. Given that, a temporal duration or temporal interval will be measured as the number of minutes between two granules. For the purposes of this study, we consider only non-negative temporal intervals. Accordingly, we define a temporal interval as an element of the set $\mathbb{N}$, where $\mathbb{N}$ denotes the set of natural numbers including zero, following the standard algebraic definition [94].

Each measurement is associated with a multidimensional context describing the environmental, temporal, and spatial conditions under which it was acquired.

Definition 3.4 (Context) Let $m \in \mathbb{R}$ be a measurement taken at time $t \in T$. A context is a tuple

$$\gamma = (c_1, \dots, c_n) \quad (3.1)$$

where each c_i is the value of a contextual dimension d_i . The set of all possible contexts is denoted by $\mathcal{C}$.

In the greenhouse scenario, the context includes:

- 26 external climate variables (weather station),
- 3 internal climate variables (permanent sensor),
- 7 temporal descriptors (year, month, week, day, etc.),
- 1 spatial descriptor (distance from the permanent sensor).

Definition 3.5 (Contextual Measurement) A contextual measurement is a tuple

$$p = (m, t, c_1, \dots, c_n) \quad (3.2)$$

where m is the measured value, t is the acquisition time, and $(c_1, \dots, c_n)$ is the associated context.

Definition 3.6 (Time Series) Given a set of contextual measurements $M = \{p_1, \dots, p_k\}$, each taken at time $t_i \in T$ and context $\gamma_i \in \mathcal{C}$, a time series is the ordered sequence

$$S = [p_1, \dots, p_k] \quad (3.3)$$

such that $t_1 < t_2 < \dots < t_k$.

Each element of the sequence corresponds to a time-step processed by the RNN.

Definition 3.7 (Feature Vector) For each contextual measurement $p = (m, t, c_1, \dots, c_n)$, the RNN receives the feature vector:

$$x = [c_1, \dots, c_n] \quad (3.4)$$

while the target value to be predicted is m .

Thus, the learning task consists of approximating the mapping:

$$f : (c_1, \dots, c_n) \mapsto m \quad (3.5)$$

across sequential time steps.

3.3.1 Model Architecture

Having established the formal structure of the prediction task and the contextual information that defines the input space, we now shift our focus to the models designed to capture these spatio-temporal dependencies. To this end, we evaluate two proposed architectures for microclimate temperature mapping, a hybrid CNN-LSTM model and a Time-LLM model, and compare them against a baseline LSTM. The following sections describe each architecture in detail.

3.3.1.1 Model Architecture I: Baseline LSTM Model

Early work on virtual sensing in greenhouse used RNNs to capture the spatiotemporal dependencies of microclimate variables [18]. The key idea is to replace temporary sensors with virtual sensors whose outputs are predicted by a context-aware RNN. This RNN is trained on data from a small set of permanent sensors and a limited number of temporary sensors placed at POIs.

As mentioned in Section 3.1, traditional greenhouse climate models often assume uniform climate conditions, a simplification that reduces model complexity but fails to capture the heterogeneity observed in practice. This limitation can lead to poor agronomic decisions, as localized microclimate variations may trigger plant diseases or uneven growth. To address this, virtual sensors powered by RNNs were introduced to provide localized climate predictions without the need for dense sensor deployment.

The approach relies on a small number of permanent sensors strategically placed in the greenhouse, complemented by temporary sensors installed at plant locations for limited periods. Data collected from these temporary sensors, together with permanent sensor readings and external weather station data, are used to train the RNN. Once trained, the network can substitute the temporary sensors with reliable predictions, effectively creating virtual sensors that continue to provide localized climate information.

A key innovation is the introduction of contextual measurements, where each observation is enriched with multiple contextual dimensions, including external weather conditions, internal greenhouse measurements, temporal descriptors, and spatial distances to permanent sensors.

By embedding these contextual dimensions, the model learns not only temporal dependencies but also spatial and environmental correlations that drive microclimate variability.

The baseline architecture is a LSTM, later extended to BiLSTM for bidirectional sequence learning. The model processes time series of contextual measurements, where each time step includes the feature vector described above. As displayed in Figure 3.1, the architecture consists of:

- Two stacked LSTM layers to capture long-term sequential dependencies in climate data.
- Dropout layers to mitigate overfitting, which is crucial when training with limited temporary sensor data.
- A dense output layer with linear activation to predict the target climate variable.

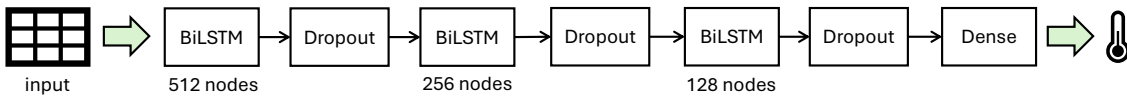


Figure 3.1: Architecture of the baseline BiLSTM model for virtual sensing.

This design enables robust handling of missing data, outliers, and long-term dependencies, making it well-suited for greenhouse environments where sensor failures or connectivity issues are common.

3.3.1.2 Model Architecture II: Hybrid CNN-LSTM for Spatio-Temporal Temperature Mapping

As described in Section 3.3.1.1, the idea of microclimate mapping and virtual sensing is implemented by training an RNN on measurements gathered from temporary sensors installed at several POIs [18], with contextual features added to the time series to represent both spatial and temporal variability. This method provides an effective balance between accuracy and cost, as the temporary sensors can be removed or redeployed after data collection. Expanding on that framework, in our earlier publication [95], we strengthened the integration of contextual information during model training. In particular, while the use

of RNNs allows us to correctly model the long-term dependencies within the data, namely the temporal relation between successive measurements (Section 2.2), we also require a tool that better exploits the local patterns among the contextual features. For instance, even if the temperature at a given moment clearly depends on the temperature measured ten minutes before, it is also influenced by other contextual factors at that moment, such as the current temperature in another nearby location, humidity levels, and other environmental factors. For this reason, we propose combining a CNN (Section 2.2.3) with the RNN model.

Building on these considerations, we explore hybrid CNN–LSTM architectures, which represent a practical approach to processing spatiotemporal data. Indeed, as previously explained, while an LSTM model can capture the temporal evolution of these patterns, a CNN model can also extract spatial relations between features in the data. Combining both types of models into a hybrid architecture enables learning both spatial and temporal features from the input data, thereby improving understanding of the underlying mechanisms governing the data [96].

There are several ways to combine these two kinds of networks:

- (a) LSTM followed by CNN
- (b) CNN followed by LSTM
- (c) CNN–LSTM parallel architecture

To better understand their composite architecture, Figure 2.4 shows the architecture of the pure LSTM model used in [3] and adopted as a term of comparison in the experiments.

In case (a), the LSTM output is used as input for the CNN. In this approach, the long-term dependencies captured by the LSTM are passed to the CNN model, which models their spatial relationships. Figure 3.2 illustrates this hybrid architecture.

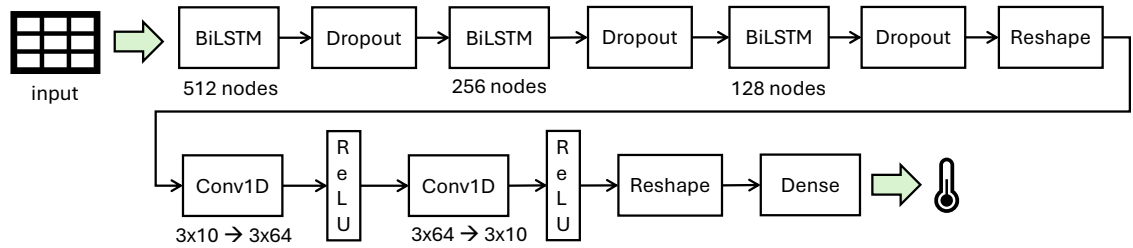


Figure 3.2: Architecture of an LSTM followed by CNN.

In case (b), the CNN output is used as input for the LSTM. In this approach, the CNN extracts features from the time series data and then feeds these features into an LSTM for further analysis. This allows the CNN to capture local patterns in the data, while the LSTM can learn the temporal dependencies among them. Figure 3.3 shows this architecture.

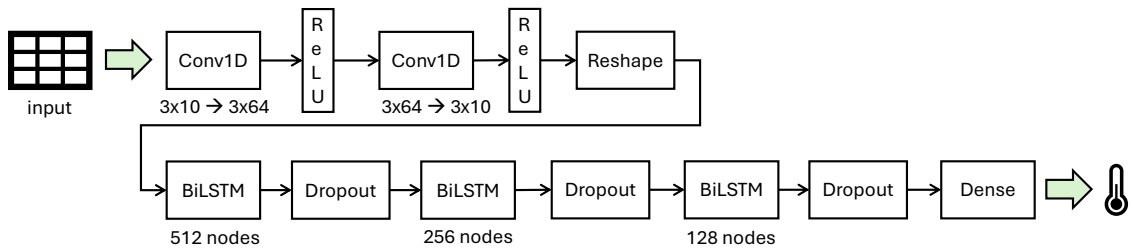


Figure 3.3: Architecture of a CNN followed by LSTM.

Finally, in case (c), a parallel architecture is used, where the LSTM and the CNN operate on the input data independently, and their outputs are concatenated and passed to a fully connected layer. This hybrid approach allows one to exploit the benefits of both models, resulting in improved performance on tasks involving sequential data compared to using only one of the two models. Figure 3.4 displays an example of a parallel composition of a CNN and an LSTM model.

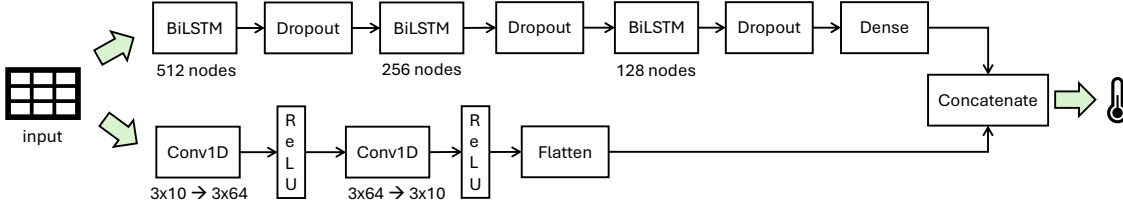


Figure 3.4: Architecture with an LSTM and a CNN in parallel.

These hybrid architectures represent a specialized approach to microclimate mapping, optimized for a balance between spatial-temporal accuracy and computational efficiency. While these architectures can be deployed on edge devices with limited resources, their development requires an initial offline training phase that relies on large, diverse datasets to adequately capture the variability of greenhouse conditions. In practice, gathering such extensive datasets in agricultural environments is challenging due to sparse sensing infrastructure and the high cost of long-term data collection. This challenge prompts exploration of an alternative methodology that utilizes pre-trained foundational models. Instead of training from scratch, these models can be adapted through a reprogramming framework to leverage the extensive sequential knowledge gained from vast amounts of data. The next section discusses whether LLMs, which have been trained on vast amounts of textual data, can be adjusted to improve predictive accuracy in these limited situations.

3.3.1.3 Model Architecture III: Time-LLM, Transformer-Based time series Forecasting

Recent progress in transformers (Section 2.2.4) and LLMs suggests an alternative pathway in virtual sensing: leveraging pretrained sequence models for time-series forecasting. In [19], we investigated Time-LLM [38], a reprogramming framework that enables pretrained LLMs such as GPT-2 to perform multivariate time series forecasting in greenhouse settings without retraining the underlying model.

The proposed approach leverages the Time-LLM framework to transform multivariate time series data into textual prototypes compatible with pretrained LLMs. The key innovation lies in reprogramming numerical time series into textual prototypes that LLMs can interpret, without retraining the LLM itself. This enables efficient forecasting with minimal data and strong generalization.

As illustrated in Figure 3.5, Time-LLM consists of five sequential stages:

1. **Input Embedding:** The multivariate time series $X \in \mathbb{R}^{n \times t}$, where n is the number of features and t is the number of time steps, is decomposed into n univariate series. Each series is normalized to mitigate distribution shifts. The normalized series is segmented into patches of length l and embedded via a linear layer.
2. **Patch Reprogramming:** Embedded patches are transformed into textual prototypes using a cross-attention mechanism and word embeddings. These prototypes describe temporal patterns (e.g., “short rise then steady decline”) and align with the LLM’s vocabulary.

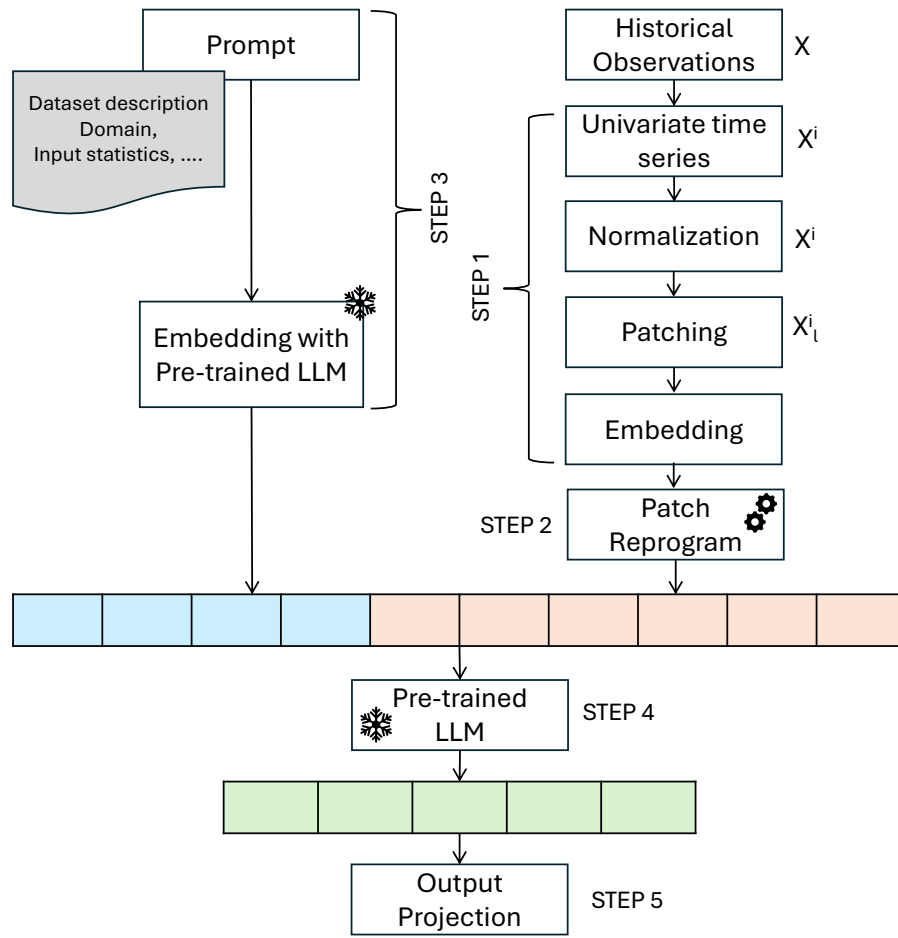


Figure 3.5: Time-LLM pipeline: from raw time series to LLM-based prediction.

- Prompt-as-Prefix:** A domain-specific prompt is prepended to the reprogrammed patches to guide the LLM's inference. This prompt includes dataset description, domain context, and statistical metadata (Figure 3.6).
- LLM Inference:** The combined prompt and reprogrammed input are passed to a frozen GPT-2 model to generate predictions.
- Output Projection:** The LLM output is flattened and converted back into numerical time series format, discarding the prefix.

Prefix Prompt Example:

This dataset contains environmental and temperature data collected from multiple sensors in a smart greenhouse setting every 15 minutes. Each row represents a timestamped observation including: temperature, distance to central station, barometric pressure, central station temperature, humidity, heating/cooling degree days, and sensor ID. The objective is to forecast future temperature values using historical multivariate time series data.

Figure 3.6: Example of domain-specific prefix prompt used in Time-LLM.

This modular design allows Time-LLM to operate without retraining the LLM, reducing computational cost and enabling rapid deployment across different sensor configurations and domains.

3.4 Application Setup

The experiments were conducted in two adjacent greenhouse tunnels located near Verona (45°20′25.9″N, 11°05′12.3″E and 45°20′47.13″N, 11°01′46.4″E) in north-eastern Italy, a region characterized by warm, humid summers and mild to moderately cold winters. Each tunnel covers 275.6 m² (50 m × 5.3 m), is oriented northeast–southwest, and is covered with transparent plastic film. As illustrated in Figure 3.7, the structures represent a typical multispan greenhouse module for this geographical area. Before crop transplanting, the soil was tilled and mulched with a semitransparent green polyethylene film. During the monitoring period, both tunnels followed seasonal crop rotations: the first hosted eggplants, lettuce, and cucumbers, while the second cultivated tomatoes, lettuce, and eggplants.



Figure 3.7: Application setup in north-eastern Italy, showing the two adjacent greenhouse tunnels used for data collection.

3.4.1 Sensing Infrastructure and Data Collection

To monitor the greenhouse microclimate, a sensing network combining fixed and temporary devices was deployed (Figure 3.8). Each tunnel contained a permanent microclimate station by Evja s.r.l.¹ installed at the center (labeled “cs” in the figure), continuously measuring air temperature, relative humidity, and solar radiation. Immediately outside each tunnel, a Davis Vantage Pro2² weather station provided 26 meteorological variables, including temperature, humidity, barometric pressure, wind speed, rainfall, solar irradiance, dew point, wet-bulb temperature, and heat index.

¹<https://www.evja.eu/solutions>

²<https://davisinstruments.com/pages/vantage-pro2>

Inside each greenhouse, approximately seven temporary wireless sensors were positioned at different distances (7.5–16.5 m) from the central station to capture spatial variability. These POI sensors primarily recorded local air temperature (and humidity, when available), providing the target measurements for microclimate mapping. All sensors were synchronized to log data every 15 minutes. This configuration, one permanent station complemented by a limited number of POI sensors, reflects a practical and cost-effective implementation of the virtual-sensing paradigm.

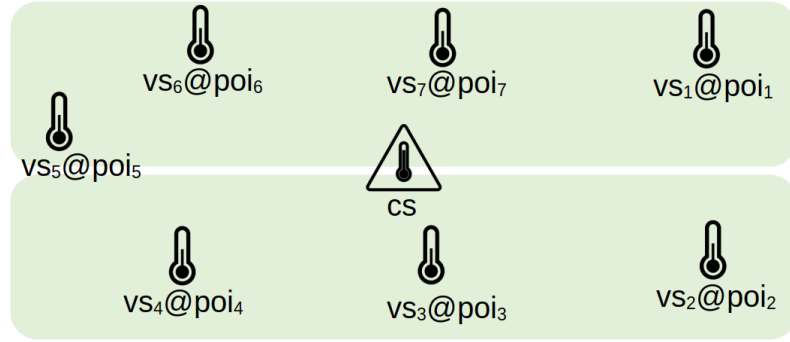


Figure 3.8: Sensor placement within the greenhouse.

Data collection spanned from late August 2021 to early July 2022, producing an almost year-long multivariate time series under real operational conditions. The dataset captures both typical seasonal dynamics and abrupt disturbances caused by severe weather events. For instance, Figure 3.9 illustrates episodes of strong winds (exceeding 28 km/h) and intense rainfall (up to 37 mm/h). These events are particularly relevant, as wind speed and rainfall intensity are among the variables used for inference in the proposed framework, contributing to a diverse and realistic dataset for evaluating model robustness.

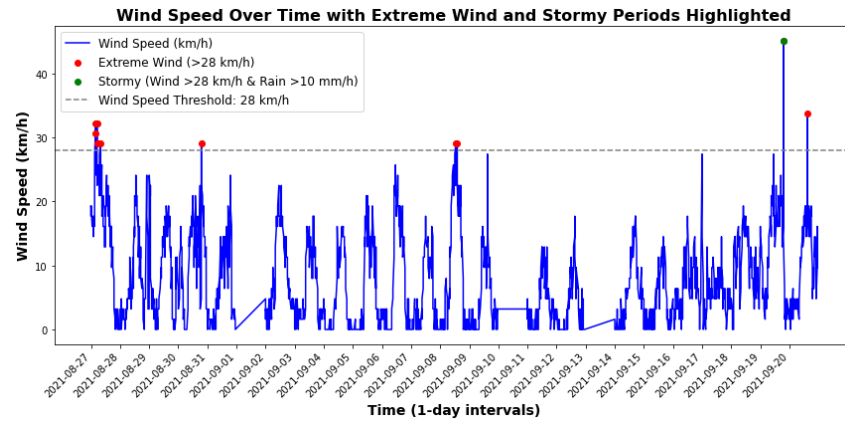


Figure 3.9: Examples of extreme weather condition variations observed within a one-month period of the experimental dataset.

At each 15-minute interval, the system recorded a comprehensive set of variables that combined external meteorological data, internal climate measurements, and contextual metadata as described in Definition 3.4. This integrated spatio-temporal feature set serves as a basis for predicting microclimate conditions at unmeasured locations, enabling the creation of virtual sensors for greenhouse monitoring.

3.4.2 Evaluation Metrics

To assess the predictive performance of proposed methods on the Verona greenhouse dataset, standard error metrics introduced in Section 2.4 were employed. These metrics quantify different aspects of model

accuracy and robustness across the seven POIs inside the greenhouse.

The metrics considered were MAE, MAPE, and MSE, in which y_i is the true value at time step i , $\hat{y}_i$, the corresponding predicted value, and N the total number of samples.

Using all three metrics ensures that performance differences across methods are assessed from complementary perspectives, covering both absolute and relative error behavior.

3.5 Experimental Results

For a fair comparison, all methods described in the previous sections were evaluated on the Verona greenhouse dataset introduced in Section 3.4.1, using the standard error metrics defined in Section 3.4.2 across seven POIs.

The original study on integrating LSTM and CNN ([17]) was focused on examining how the integration of convolutional layers impacts recurrent models used for virtual sensing. Therefore, we only report the baseline results from that research. The extensive experimental analyses conducted for the Time-LLM framework, such as studies on sequence length, ablation of contextual features, and generalization tests, were not replicated for the combined LSTM, CNN models. In this context, these hybrid models serve as a representative reference point, rather than being a fully re-evaluated architecture.

3.5.1 Experiments Result of Hybrid LSTM and CNN Method

In this experiment, the method based on the BiLSTM model, discussed in Section 3.3.1.1, was used as the baseline in the experimental evaluation. The performance of other hybrid approaches is compared against this baseline. To assess the effect of combining a CNN with an LSTM architecture, we evaluated three different hybrid models that differ in the ordering and integration of the CNN and LSTM components outlined in Section 3.3.1.2. Table 3.1 summarizes the quantitative results of this experiment.

Table 3.1: Comparison of temperature-prediction errors for virtual sensing using the baseline LSTM and the proposed CNN-LSTM hybrid models.

POI	LSTM followed by CNN			CNN followed by LSTM			Parallel LSTM and CNN			Baseline LSTM		
	MAPE%	MAE	MSE	MAPE%	MAE	MSE	MAPE%	MAE	MSE	MAPE%	MAE	MSE
1	7.70	1.35	3.37	13.15	2.56	11.49	10.45	1.79	5.54	11.79	2.36	9.95
2	7.22	1.29	3.30	13.85	2.87	14.28	8.88	1.59	4.26	10.43	2.14	8.46
3	8.86	1.57	4.54	16.08	3.30	18.76	9.03	1.64	4.70	14.65	3.00	16.10
4	7.23	1.34	3.77	15.14	3.20	18.91	8.36	1.55	4.70	11.44	2.43	11.86
5	8.04	1.47	4.09	16.38	3.34	19.19	9.66	1.77	5.33	15.33	3.17	17.42
6	7.02	1.34	3.95	14.37	3.09	17.61	10.23	1.93	6.90	11.76	2.52	12.43
7	6.57	1.23	3.07	12.49	2.68	14.19	7.85	1.46	3.98	10.56	2.28	11.16
All	5.20	0.97	2.15	5.36	0.99	2.16	5.46	0.99	1.82	5.97	1.09	2.39

The results of Table 3.1 show that the combined use of LSTM and CNN consistently outperforms the baseline. At the individual POI level, the LSTM-CNN hybrid model achieves the lowest errors in most cases. This indicates that applying convolutional layers after the recurrent component helps refine spatial-temporal features extracted from the data in this application. In contrast, the CNN-LSTM configuration performs noticeably worse across all POIs relative to the best-performing model. This suggests that extracting spatial patterns before modeling temporal dependencies is less effective for greenhouse microclimate data, where temporal continuity is dominant.

The parallel CNN-LSTM architecture demonstrates intermediate performance; while it consistently outperforms the baseline LSTM model, it does not reach the accuracy levels of the LSTM-CNN model. This suggests that, although parallel feature extraction captures complementary spatial and temporal cues, the sequential refinement of the LSTM-first pipeline is more effective for this application.

The baseline LSTM model exhibits the highest errors at nearly all POIs, which confirms that incorporating convolutional layers enhances the model's ability to capture local spatial variations in the greenhouse environment.

When aggregating results across all POIs, the advantages of the hybrid architectures become more evident. The LSTM-CNN model achieves the lowest overall errors, outperforming both the parallel hybrid and the baseline LSTM model. Although the CNN-LSTM model shows weaker performance per POI, it produces competitive overall metrics through averaging, yet it still does not exceed the best-performing configuration.

Overall, the findings demonstrate that enriching the baseline recurrent model with convolutional processing significantly improves temperature prediction accuracy and that the ordering of these components plays an important role in capturing the underlying spatiotemporal structure of greenhouse microclimate dynamics.

An example of the predicted versus actual temperature values in some random dates per four models during the test phase is shown in Figure 3.10, illustrating the qualitative differences in prediction accuracy.

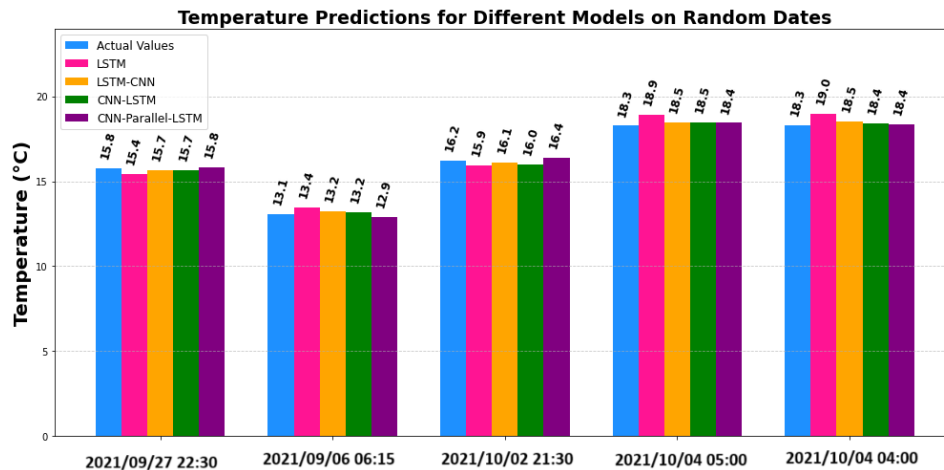


Figure 3.10: Evaluation of virtual-sensing performance for the baseline LSTM model and the proposed CNN-LSTM hybrid architecture, showing predicted versus actual temperature values.

3.5.2 Experiments result of Time-LLM framework

Given historical data presented in Section 3.4.1, several experiments have been performed to evaluate the performance of the Time-LLM model in forecasting the future temperature in each POI. In all the experiments, we used the official Time-LLM source code³ and, as a pre-trained LLM model, the GPT2-small-0.124b⁴. This is the smallest version of GPT-2, with 124M parameters, and it can also be run on a machine with quite limited resources. In our case, we perform the experiments on a notebook equipped with a NVIDIA GeForce RTX 2060 mobile graphics card with 6 GB of VRAM.

³<https://github.com/KimMeen/TimeLLM>

⁴<https://huggingface.co/openai-community/gpt2>

In all experiments, we used the following prompt prefix. This prefix provides the Time-LLM model with essential context about the dataset structure, the meaning of each feature, and the objective of the prediction task. In particular, it describes the multivariate time-series inputs, the spatial information associated with each POI, and the target variable to be forecast.

This dataset contains environmental and temperature data collected from multiple sensors in a smart greenhouse setting every 15 minutes. Each row represents a timestamped observation including:

- temperature: Temperature measured by the sensor (target variable).
- dist_to_central_station: Distance of the sensor from the central weather station (in meters).
- barometer_hpa: Atmospheric pressure in hectopascals.
- temp_centre: Temperature measured at the central station.
- hightemp_centre: Maximum temperature at the central station during the period.
- lowtemp_centre: Minimum temperature at the central station during the period.
- hum: Relative humidity percentage.
- heating_degree_days/cooling_degree_days: Degree-day indicators for thermal comfort analysis.
- sensor_id: Identifier of the sensor that generated the measurement.

The objective is to forecast future values of outside temperature using historical multivariate time-series data from all sensors.

3.5.2.1 Experiment 1: Effect of Sequence and Prediction Length

The first experiment examines the effects of sequence and prediction length on prediction accuracy. With “sequence length” we identify the number of time steps composing each time series in the input, while the “prediction length” denotes the length of the required predictions. For this experiment, we consider only the data about the first sensor poi_1 and the following alternative configurations:

1. Sequence and prediction length = 96 (i.e., 24 hours)
2. Sequence and prediction length = 6 (i.e., 1.5 hours)
3. Sequence length = 6 (i.e., 1.5 hours) and prediction length = 1 (i.e., next time step)

The initial learning rate is set to 0.001, and we set an early stopping condition of 10 epochs without improvements. To assess the effectiveness of Time-LLM, we include DLinear (Section 2.2.4) as a comparative baseline and compare them on both univariate and multivariate time series: pure time series, where at each time step only the temperature is observed, and contextual time series, where, for each time step, several other features are collected alongside the temperature. The obtained results are reported in Table 3.2 in terms of MAE, namely the sum of absolute errors (i.e., predicted versus actual value) divided by the sample size. As shown, Time-LLM always performs better than the corresponding DLinear configuration, with the multivariate alternative as the best option, confirming the importance of contextual information in microclimate time series forecasting, as already demonstrated in [18, 17].

With respect to the role of the sequence and prediction length, we can observe that the accuracy of Time-LLM increases as such lengths decrease, while it appears that DLinear is best suited for long time series, even if its performances are always worse than those of the competitor. Finally, we observe that Time-LLM performs very well when the required prediction is for the immediately following single time step, rather than a complete sequence.

Figure 3.11 compares the trend between the real temperatures and the predictions made by Time-LLM for POI 1 in the second configuration, namely with sequence length equal to 6 and prediction length of 1.

Table 3.2: Error comparison between the DLinear and Time-LLM models in the presence of univariate and multivariate time series and different sequence and prediction lengths.

Model	time series	Seq. Len.	Pred. Len.	MAE
DLinear	Univariate	96	96	0.55876
DLinear	Multivariate	96	96	0.63043
Time-LLM	Univariate	96	96	0.52088
Time-LLM	Multivariate	96	96	0.47872
DLinear	Univariate	6	6	0.38478
DLinear	Multivariate	6	6	0.27984
Time-LLM	Univariate	6	6	0.13739
Time-LLM	Multivariate	6	6	0.15660
DLinear	Univariate	6	1	0.48458
DLinear	Multivariate	6	1	0.37506
Time-LLM	Univariate	6	1	0.06614
Time-LLM	Multivariate	6	1	0.03960

As you can see, the prediction closely follows the real trends, with only a few instances in which it smooths the estimated temperature values.

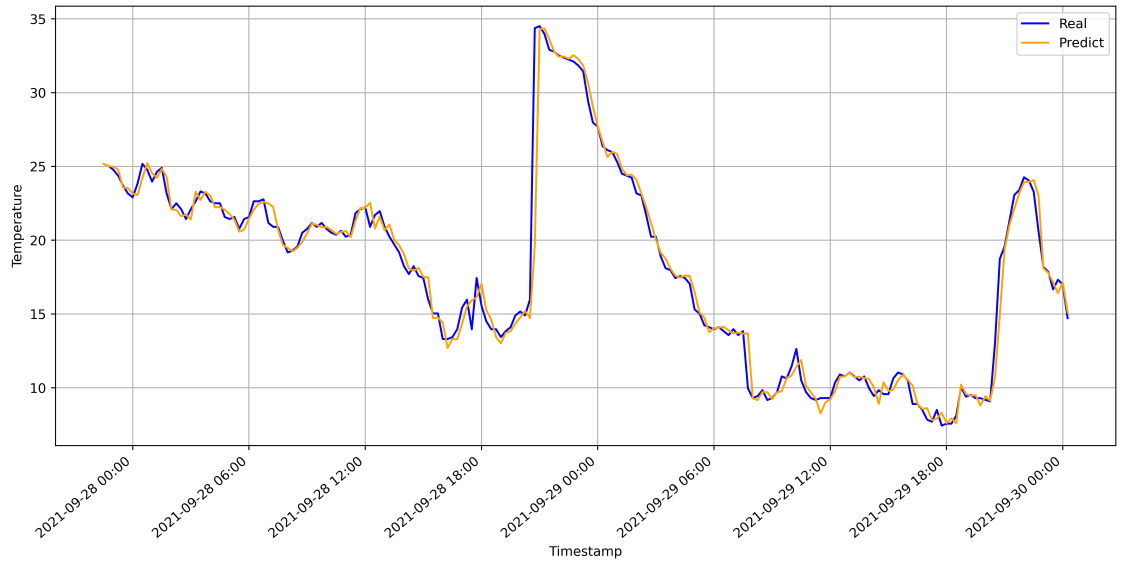


Figure 3.11: Real temperatures and Time-LLM predictions for POI 1 with sequence and prediction length equal to 6 (i.e., 1.5 hours).

3.5.2.2 Experiment 2: Role of Contextual Features

In this experiment, we evaluate the role of contextual features in predicting accuracy (Definition 3.4). Even if the different accuracies obtained with univariate and multivariate methods confirm the usefulness of these additional features, we would like to investigate whether all of them are necessary or whether some introduce noise into the input information. Therefore, given the best configuration obtained before, namely the Time-LLM with sequence length equal to 6, prediction length equal to 1, and multivariate time series, we compare the previous result with the ones obtained by considering only the following contextual features: the temperature and humidity measured by the central sensor *cs*, the heating and cooling degrees days. The obtained results are reported in Table 3.3 for each POI. We observe that using a multivariate time series

with a selected set of contextual features greatly improves accuracy, with an MAE decreasing from 0.15660 to 0.03960 for POI 1. Moreover, the table compares the obtained results with the best results reported in Section 3.5.1, using a bi-directional LSTM followed by a CNN, as well as the baseline bi-directional LSTM alone. As can be noticed, Time-LLM not only outperforms the other two baseline techniques, but it is also able to provide quite stable results for all POIs with MAPE values always between 1.0% and 1.5%, and MAE values always between 0.030 and 0.040.

Table 3.3: Error comparison for all virtual POIs between Time-LLM, the LSTM + CNN model, and the baseline Bi-directional LSTM. The best result for each metric is highlighted in bold.

POI	Time-LLM Multi			LSTM - CNN			LSTM		
	MAE	MAPE (%)	MSE	MAE	MAPE (%)	MSE	MAE	MAPE (%)	MSE
1	0.03960	1.33	0.25	1.35	7.70	3.37	2.36	11.79	9.95
2	0.03911	1.30	0.24	1.29	7.22	3.30	2.14	10.43	8.46
3	0.03894	1.43	0.28	1.57	8.86	4.54	3.00	14.65	16.10
4	0.03890	1.46	0.29	1.34	7.23	3.77	2.43	11.44	11.86
5	0.03029	1.11	0.24	1.47	8.04	4.09	3.17	15.33	17.42
6	0.03202	1.12	0.26	1.34	7.02	3.95	2.52	11.76	12.43
7	0.03106	1.08	0.23	1.23	6.57	3.07	2.28	10.56	11.16

3.5.2.3 Experiment 3: Generalization Capabilities

The last experiment we performed is related to the generalization capabilities of Time-LLM model, namely the ability: (1) to build a unique model for all the POIs together, instead of training a different reprogramming model for each POI, and (2) to use the unique model to provide predictions for POIs it has never seen before. For the first investigation, we built a training and test set including historical time series from all the POIs with an additional feature representing the POI identifier, while for the second investigation, we built a training set containing historical time series enriched with the identifier from all the POIs except the seventh, while the test set will include only requests for the seventh. The obtained results are reported in Table 3.4. We can notice that building a unique model decreases the accuracy of the obtained results, with an MAE that becomes about half a degree, while the MAPE percentage doubles with respect to the values reported in column “Time-LLM Multi” of Table 3.3. However, the increase in error is negligible, confirming the LLM’s strong generalization capabilities.

Table 3.4: Performance of the global Time-LLM model trained on all virtual POIs.

Dataset	MAE	MAPE (%)
All	0.49	2.28
All (no 7)	0.50	2.30

Figure 3.12 compares the results obtained for predictions about POI 7 in the two cases: the one in which a unique global model has been trained also with such POI, and the case in which the unique global model has been trained with all POIs except the seventh. As can be seen, the trend is very similar, and in this case, some peaks are either added or missed by the predictions relative to the solution for which each POI’s model is specifically trained.

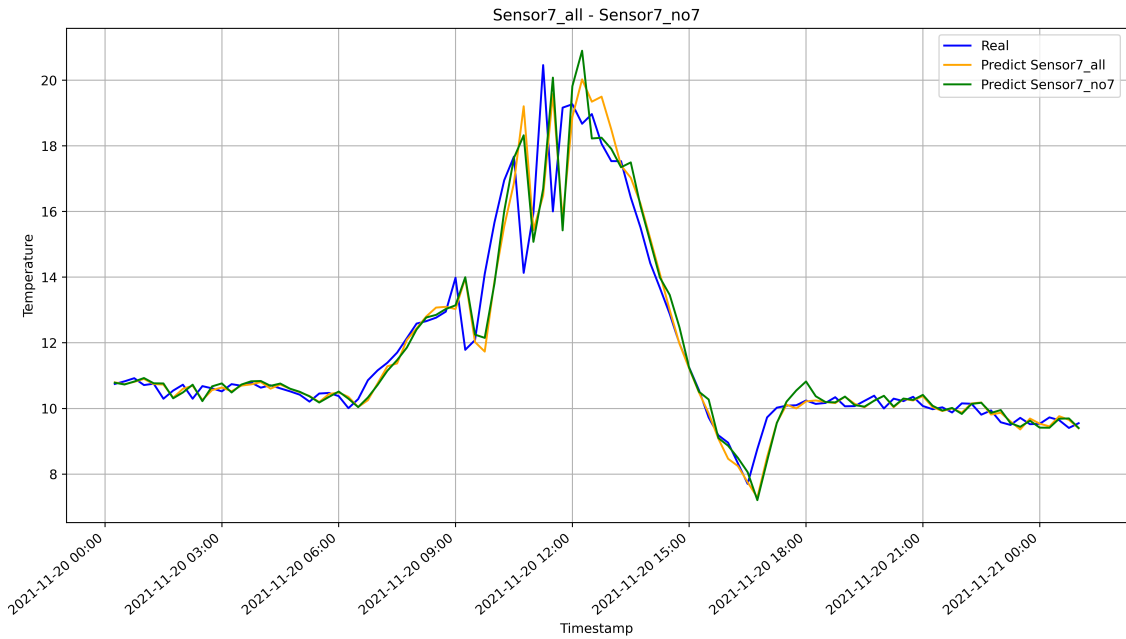


Figure 3.12: Comparison of observed temperatures and predictions from a global Time-LLM model trained with and without POI 7.

3.6 Discussion and Comparative Analysis

This chapter examined various virtual sensing architectures, including baseline recurrent models, hybrid CNN-RNN designs, and a pretrained transformer for modeling greenhouse microclimates.

The experimental results demonstrated significant differences in performance, complexity, and suitability for deployment. Rather than viewing these models as direct competitors that replace one another, they should be regarded as part of a multi-tiered solution framework that addresses various agricultural infrastructure needs. The discussion below synthesizes these findings and interprets their implications for real-world agricultural sensing systems.

3.6.1 Analysis of Predictive Accuracy

The prediction accuracy metrics (Table 3.1 and Table 3.3) show a large difference between the models.

- The Baseline and Hybrid Paradigm:** The BiLSTM baseline provides moderate accuracy but shows high variability across locations, with MAPE values ranging from 10–15%. In contrast, all CNN+RNN hybrid models outperform the baseline, with the LSTM-CNN architecture achieving the best results. This hybrid consistently reduces MAPE to around 6–8% across most POIs and yields the lowest overall error ($\approx 5.20\%$), demonstrating that applying temporal modeling first, followed by spatial refinement, is the most effective strategy in this setting.
- Time-LLM transformer-based model:** Time-LLM achieved the best performance by a large margin. Its transformer-based design produced extremely low errors, with MAPE around 1.0–1.5% at all POIs and MAE of only 0.03–0.04°C, an order of magnitude better than the RNN and hybrid CNN+RNN models. Its accuracy was highly consistent across locations, indicating that the transformer captured both temporal patterns and sensor relationships. Even when predicting at unseen POIs, performance remained strong (global MAPE $\approx 2.3\%$). Overall, Time-LLM achieved the highest accuracy across

all conditions, demonstrating that a well-adapted pretrained transformer can model greenhouse microclimates far more precisely than specialized RNN-based architectures.

Although the experiments were conducted in two adjacent tunnels, the proposed models were intentionally designed to generalize beyond this specific physical layout. The use of context-rich feature vectors, including external weather variables, internal climate measurements, temporal descriptors, and distance-based spatial encoding, allows the models to learn relationships that are not restricted to a particular greenhouse geometry or sensor arrangement. The dataset also spans a full monitoring period during which the tunnels hosted different crops, including eggplants, lettuce, cucumbers, and tomatoes, introducing natural variation in canopy structure, shading, and transpiration behavior. While the precise timing of each crop cycle is not explicitly considered in this study, the presence of these biologically diverse conditions exposes the models to multiple microclimate regimes. This combination of spatial encoding and biological variability strengthens the robustness of the learned representations. Nevertheless, full cross-site generalization cannot be guaranteed without external validation. Such cross-greenhouse evaluation and transfer-learning strategies could form a valuable direction for future work.

3.6.2 Computational Efficiency and Deployment Trade-offs

The models were evaluated not only for prediction accuracy but also for computational efficiency, including time granularity, training speed, memory footprint, and suitability for inference, which are critical factors in real-world agricultural applications.

3.6.2.1 Effect of Time Granularity

The choice of a 15-minute sampling interval is crucial for the quality of the predictions. Finer intervals (e.g., 1–5 minutes) can capture rapid fluctuations in microclimate but introduce significant noise and redundancy. Additionally, they lead to longer sequence lengths, which can reduce model stability and increase computational costs. On the other hand, coarser intervals (30–60 minutes) may oversmooth the data and fail to capture short-term temperature dynamics. Therefore, the selected 15-minute interval strikes a balance between preserving meaningful microclimate variations and maintaining manageable sequence lengths for LSTM-based models and Time-LLM, without overwhelming them with noise.

3.6.2.2 Memory Footprint

All trained LSTM-based and hybrid LSTM+CNN models were compact, with file sizes ranging from 15.7 MB to 16.6 MB, making them suitable for deployment on standard embedded systems. In contrast, the Time-LLM model, built on GPT-2 Small with 124 million parameters, requires approximately 500 MB of storage, even with a lightweight reprogramming module applied to greenhouse sensor data. As a result, Time-LLM is better suited for GPU-equipped edge devices or cloud deployments than for low-power embedded platforms.

3.6.2.3 Training Time

Training times varied by architecture:

- **Baseline LSTM:** fastest at 164 seconds.
- **CNN-LSTM hybrid:** slowest at 373 seconds.
- **Other hybrid designs:** intermediate, with LSTM-CNN taking 225 seconds and parallel LSTM+CNN 199 seconds.

All training experiments were conducted on a Windows 11 laptop with an Intel Core i7-11800H CPU (8 cores, 16 threads), 16 GB RAM, and an NVIDIA RTX 3050 GPU.

Despite differences, all LSTM-based models completed training in under 7 minutes, demonstrating practicality for iterative model development and updates.

3.6.2.4 Trade-Offs and Deployment Recommendations

For deployment, all LSTM-based models are lightweight enough to enable low-latency inference in standard greenhouse monitoring systems. Hybrid models require longer training times but provide improved accuracy, while the LSTM remains a solid option for resource-constrained environments due to its minimal training and inference requirements. Its 16 MB size allows it to reside directly on an edge gateway or a sophisticated wireless sensor node, enabling real-time adjustments without requiring a stable internet connection.

Time-LLM, despite offering superior prediction accuracy, has a significantly higher computational and memory footprint. Its inference pipeline and model size make it less suitable for typical low-power greenhouse hardware. It is best positioned as a Digital Twin component, with data pushed to a central server or cloud instance to enable high-fidelity microclimate mapping for long-term agronomic research.

3.7 Conclusion

This chapter discussed a range of virtual sensing architectures, including baseline recurrent models, hybrid CNN-LSTM designs, and pretrained Time-LLM, to assess their suitability for modeling greenhouse microclimates. The experiment analysis indicates that these models should not be considered direct competitors; instead, they function as complementary components within a multi-tier sensing framework designed to address various agricultural infrastructure constraints.

In terms of accuracy, the hybrid LSTM-CNN architecture consistently outperformed the baseline LSTM, reducing MAPE from 10-15% to approximately 6-8% across most POIs. This demonstrates that combining temporal modeling with spatial refinement is an effective strategy for predicting microclimates. However, the Time-LLM transformer achieved the highest accuracy by a considerable margin, delivering MAPE values of 1.0-1.5% and maintaining strong generalization even at unseen pois. These findings indicate that pretrained transformers can effectively capture complex temporal-spatial relationships with remarkable precision.

The computational analysis provides a clearer understanding of the deployment landscape. All LSTM-based and hybrid models are lightweight and quick to train, making them suitable for use in embedded greenhouse hardware and low-power edge devices. In contrast, Time-LLM requires around 500 MB of storage, positioning it as a high-fidelity DT component rather than a model for on-node inference.

Overall, this chapter establishes a coherent hierarchy of virtual sensing solutions: compact hybrid models for embedded, real-time monitoring and transformer-based architectures for cloud-supported DTs that require dense, high-precision environmental mapping. These insights lay the groundwork for the next chapter, which will focus on integrating mobile rovers within the concept of greenhouse virtual sensing.

Chapter 4

Modeling of Autonomous Rover for Greenhouse Temperature Monitoring

Chapter contents

4.1	Introduction	45
4.2	Related Work	45
4.3	Problem Formalization and Proposed Solution	47
4.3.1	Common Definitions Used Across All Methods	47
4.3.2	Methodology I: Fixed-Trajectory Baseline	48
4.3.3	Methodology II: DQL-Based Trajectory Optimization	49
4.3.4	Methodology III: PPO-Based Energy-Aware Optimization	51
4.4	Experimental Results	57
4.4.1	DQL Stop-Pattern Optimization Results	57
4.4.2	Experiment Results: PPO Energy-Aware Trajectory Optimization	58
4.4.3	Predictive Performance Comparison	60
4.5	Discussion and Comparative Analysis	61
4.5.1	Information Density and Prediction Accuracy	62
4.5.2	Model Complexity and Efficiency	62
4.6	Conclusion	63

4.1 Introduction

As established in Section 3.1, virtual sensors offer a cost-effective solution for greenhouse monitoring by utilizing historical data to predict microclimate variables. However, these models are inherently limited by the static nature of their training data. Typically, they are calibrated or trained using data from fixed or temporary sensor deployments, resulting in a rigid spatial sensing framework. This constraint limits their ability to adapt to the high spatiotemporal variability often present in greenhouse environments.

To bridge this gap, we propose integrating autonomous agricultural robots, hereafter referred to as rovers, equipped with mobile sensing units. These agents create a dynamic data acquisition layer that performs on-demand measurements at specific POIs. Throughout this chapter, “POI” refers to a Point of Interest in the general sense. In formal definitions and mathematical models, we use the lowercase notation poi_i to denote a specific element within the set of all available points $\mathcal{P}$, where i serves as the unique identifier for each location.

By transitioning from static to mobile sensing, the system can:

- **Refine Spatial Models:** Offer real-time ground-truth data to enhance virtual sensor predictions in areas with high uncertainty.
- **Target Environmental Gradients:** Concentrate sampling efforts in regions experiencing rapid microclimatic shifts or unusual behavior.
- **Reduce Infrastructure Reliance:** Ensure high-quality monitoring without the high costs associated with dense, permanently installed sensor networks.

This new capability introduces the challenge of trajectory planning. The robot must operate within strict constraints, including limited energy capacity, restricted travel time, and mechanical wear. Traditional heuristic-based patrolling methods, such as fixed-route cycles, are often suboptimal in greenhouse environments, where conditions are non-uniform.

To address these challenges, this chapter introduces an optimization framework based on DRL. Unlike traditional solvers, DRL enables the robot to learn sophisticated navigation policies that balance trade-offs among time constraints, energy consumption, and information gain. Our proposed models incorporate contextual factors, including greenhouse structural layouts, historical sensor variance, and real-time feedback, to ensure the robot prioritizes locations where data collection will have the greatest impact.

We utilize the experimental setup previously described in Section 3.4 to ensure a consistent comparative analysis. It is important to clarify that the contributions of this chapter do not lie in robotic control or obstacle avoidance. Instead, this work focuses on optimizing the sensing process. Then, the robotic approaches are evaluated based on their sensing performance, computational feasibility, and adaptability. In addition, this chapter continues the data-driven environmental modeling effort by optimizing mobile sensing. No mechanistic climate model is used, reflecting the practical constraints of greenhouse physics.

4.2 Related Work

Before discussing mobile sensing and trajectory optimization within greenhouses, it is essential to define the focus of this section. The concept of virtual sensing, its motivations in greenhouse environments, and the primary categories of virtual sensor models, ranging from model-based approaches (e.g., CFD) to data-driven methods (e.g., RNN/LSTM architectures), have been reviewed in the “Related Works” section of the previous chapter (Section 3.2). That section also addressed the limitations of dense WSNs, highlighted the need for cost-effective alternatives, and examined the role of contextual information in microclimate estimation.

In this section, we focus on two complementary research directions in virtual sensing through mobile platforms: the use of rovers as mobile sensing units within greenhouses, and path planning and active sensing strategies, particularly those based on RL, which determine how these rovers should navigate to maximize sensing utility.

The growing use of autonomous and semi-autonomous agricultural robots has created opportunities for using them not only as mechanical operators but also as mobile sensing platforms within greenhouses and other controlled growing environments [97]. Initial research on greenhouse robotics primarily focused on automating tasks, such as harvesting, spraying, and monitoring crops [98, 99, 100, 101], in which robots navigate through crop rows and interact with plants or infrastructure. As microclimate variations became increasingly evident, the use of mobile robots for environmental data collection emerged as a practical solution. Instead of relying on dense WSNs, a robot equipped with a limited number of sensors can periodically navigate through the greenhouse and measure climatic variables at multiple locations. This mobile sensing approach reduces WSNs hardware and maintenance costs while providing flexible spatial coverage applicable for different crops, greenhouse layouts, and growth stages.

Environmental sensors installed on these platforms can be utilized to create spatial maps of environmental variables such as temperature and humidity. These measurements can often be combined with external weather data. By using virtual sensing models, such as regression, Gaussian processes, or LSTM-based predictors, these sparse and trajectory-dependent measurements can be interpolated or inferred at locations that have not been visited. This process effectively transforms the robot into a moving “anchor” for a virtual sensor network [102, 103, 104].

A key insight from this research is that the informational value of a robot’s trajectory is influenced not only by the area’s coverage but also by when and how often specific locations are visited. While fixed, cyclic trajectories (such as visiting all rows at fixed intervals) are easy to implement, they may not be ideal in situations where climatic dynamics are highly variable in both space and time. This limitation highlights the need for adaptive path planning and active sensing strategies, in which the robot’s movements are specifically optimized to enhance the quality of the reconstructed climate field or improve the performance of subsequent decision-making tasks.

Deciding about visiting locations, their order, and frequency is closely related to informative path planning and active sensing. Classical approaches formulate this as a combinatorial optimization problem, similar to variants of the “Traveling Salesman Problem” [105]. The main objective of these methods is to maximize coverage and information gain while adhering to time or distance constraints [104, 106]. These methods largely depend on hand-crafted heuristics or mixed-integer programming and generally assume that the performance criteria are static and time-invariant. These limitations reduce their effectiveness in dynamic greenhouse environments, where measurement values can vary with context, such as time of day, weather conditions, crop stage, and the current state of the prediction model.

RL, and DRL [39], provides a natural framework to address these limitations. In RL-based active sensing, the robot is modeled as an agent that interacts with the environment, observing a state (e.g., battery level, recent climate history), choosing an action (e.g., which path to follow), and receiving a reward that encodes measurement utility, energy consumption, or prediction accuracy. Over time, the agent learns a policy that maximizes the expected cumulative reward, effectively discovering sensing strategies that adapt to the environment’s dynamic spatiotemporal structure [107].

Value-based DRL methods such as DQL [45] have been applied to path planning and data collection in wireless sensor networks and environmental monitoring, where the agent learns the long-term value of visiting specific regions under different conditions [108, 109]. In these DRL works, the reward is typically defined in terms of coverage or reduction in estimation error, and the learned policy often follows static or

greedy baselines. However, DQL are limited to discrete action spaces and can become unstable in more complex, constrained environments [49].

These drawbacks encouraged a move toward policy-gradient and actor-critic methods. These approaches are better suited for handling more complex decision-making problems, including continuous action spaces and situations involving multiple competing objectives. In this category, PPO [50] has been widely adopted for a range of complex robotic path-planning and decision-making problems, including scenarios that involve long-horizon planning and coordination among multiple agents [110, 111, 112].

In summary, compared to static path planners, RL-based active sensing can adapt to changing climatic conditions and operational constraints. Additionally, it can be fine-tuned as new data becomes available. This flexibility makes it particularly well-suited for virtual sensing frameworks in greenhouses, where the robot's trajectory is not an end in itself but a means to improve the quality and reliability of data-driven climate estimation. Building on these principles, we present our studies on optimizing path planning for agricultural robots in greenhouse environments in the following sections.

4.3 Problem Formalization and Proposed Solution

The objective of this section is to formalize the decision-making problem that enables an autonomous rover to collect informative microclimate measurements while operating inside the greenhouse. To explore this objective, three sensing strategies are evaluated: a deterministic baseline with a fixed trajectory, a DQL policy, and a PPO policy. Although these methods vary in how they direct the rover toward high-value sampling locations, they share several fundamental concepts, including the representation of the greenhouse environment, the definition of POIs, and the structure of the sensing task, which are discussed in Section 3.3. Additionally, each strategy introduces specific assumptions, constraints, and decision rules that require distinct formal treatment. The following sections describe the components specific to each method, while concepts and definitions common to all methods are referenced directly from the previous chapter to avoid redundancy.

4.3.1 Common Definitions Used Across All Methods

The sensing strategies discussed in this chapter share a set of foundational concepts that define the structure of the sensing task. To maintain clarity and avoid redundancy, we summarize all common definitions below, with references to their formal statements provided in the previous chapter:

- **POI** In the methods presented in this chapter, these locations correspond to the POIs at which the autonomous rover is allowed to stop and acquire microclimate measurements.

To formalize the problem, we assume that:

- There is a set of POIs to be periodically sampled for a climatic variable of interest;
- The rover is equipped with a device capable of sampling that variable at specified intervals;
- The rover periodically follows a trajectory that potentially visits all POIs.
- **Temporal Domain:** This refers to the ordered set of discrete time instants at which measurements are acquired, introduced in Definition 3.2.
- **Temporal Granularity:** This is the mapping that groups temporal instants into indivisible units, as described in Definition 3.3.
- **Context:** This refers to the multidimensional descriptor associated with each measurement, encompassing environmental, temporal, and spatial attributes, defined in Definition 3.4.

- **Contextual Measurement:** This is the combination of a measurement, its acquisition time, and its contextual descriptors, given in Definition 3.5.
- **Time Series:** This denotes the ordered sequence of contextual measurements used as input for recurrent models, formalized in Definition 3.6.
- **Feature Vector:** This is the input representation provided to the learning model at each time step, defined in Definition 3.7.

These definitions form the shared conceptual foundation for all three decision-making strategies. Therefore, the following sections will focus exclusively on the elements specific to each method, referring back to the definitions above only when necessary.

4.3.2 Methodology I: Fixed-Trajectory Baseline

The baseline methodology, building on the framework established in [3], uses a mobile rover that follows a deterministic, cyclical trajectory within the greenhouse. As illustrated in Figure 4.1, the rover traverses a predefined path that intersects a set of POIs. In this configuration, the spatial sensing structure is static; every $poi_i \in \mathcal{P}$ is located directly along the rover's navigation route to ensure consistent accessibility.

At each POI, the rover records measurements at regular time intervals. These time-stamped, spatially tagged observations form a spatiotemporal dataset that captures environmental dynamics along the rover's route. Because the rover follows a cyclical path, each POI is revisited periodically rather than continuously.

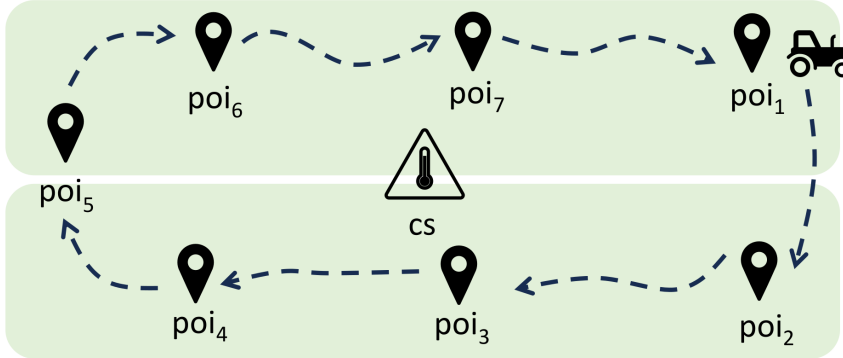


Figure 4.1: Greenhouse layout highlighting the fixed navigation path and the spatial distribution of sampling POIs.

To reconstruct a continuous microclimate profile during periods when the rover is not physically present at a POI, an LSTM network was used. By analyzing historical rover trajectories and recorded climate variables, the model learns both the temporal patterns and the spatial relationships among POIs and their surrounding sampling points.

Each LSTM input sequence consists of sensor measurements collected by the rover at multiple spatial locations over time, along with contextual features described in Section 4.3.1. The model maps a time series input vector $\mathbf{x}_t$ to the predicted microclimatic state $\hat{y}_t$ at a target poi_i . The input vector $\mathbf{x}_t$ is defined as:

$$\mathbf{x}_t = [\mathbf{s}_t, d_t, \mathbf{e}_t, \mathbf{g}_t, \mathbf{m}_t]. \quad (4.1)$$

Here:

- $\mathbf{s}_t$: measurements collected by the rover at its current spatial location,
- d_t : distance between the rover's position and the target POI,
- $\mathbf{e}_t$: external climate features from the fixed weather station,

- $\mathbf{g}_t$: measurements from the central greenhouse sensor,
- $\mathbf{m}_t$: temporal metadata (e.g., hour, day, month).

During the training phase, temporary ground-truth sensors are deployed at selected POIs to provide reference measurements. After training, the model is used to infer microclimatic conditions at any POI, even when no direct observation is available at that time. This approach enables virtual sensing with fewer physical sensors by leveraging the rover's motion to emulate a distributed sensing network.

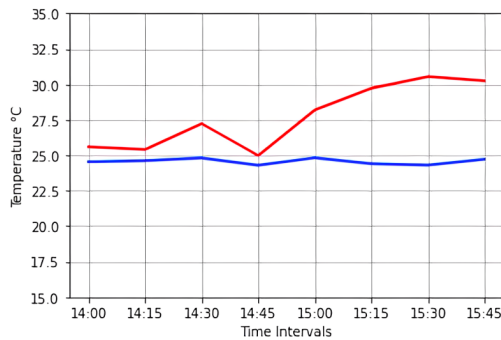
This method serves as a baseline for evaluating more advanced strategies, such as adaptively planned trajectories and context-aware sensing policies, by establishing performance under deterministic and resource-constrained operating conditions.

4.3.3 Methodology II: DQL-Based Trajectory Optimization

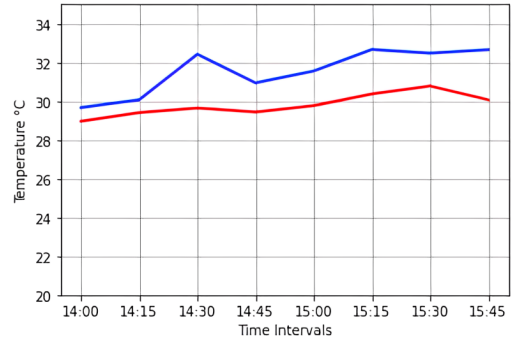
As discussed in Section 4.3.2, the baseline framework involves a rover that follows a fixed route to collect climate data, enabling virtual sensing without the need for temporary sensors.

While the fixed-trajectory baseline provides a consistent data stream, it is inherently inefficient in dynamic environments. Greenhouse microclimates exhibit non-uniform spatiotemporal variance; consequently, the “Information Value” of a measurement at any given poi_i is not constant.

As demonstrated in Figure 4.2, the thermal dynamics of specific locations fluctuate based on external conditions. On October 1st (Figure 4.2(a)), poi_3 exhibits high variance ($\approx 4^\circ\text{C}$), while poi_7 remains static. Conversely, on August 8th (Figure 4.2(b)), poi_7 becomes the high-variance node. A fixed trajectory forces the rover to spend equal time at both locations, wasting energy and time on redundant data collection. By identifying and skipping “low-information” POIs, the rover can increase the sampling frequency of “high-information” areas.



(a) October 1st, 2021



(b) August 8th, 2021

Figure 4.2: Temperature measurements at poi_3 (blue line) and poi_7 (red line) on two different days in a greenhouse near Verona, Italy.

Building on this observation, we propose a context-aware enhancement to fixed-trajectory strategies. We formulate the task as an Informative path planning problem in which the rover must select an optimal stop pattern σ from the global set of POIs:

$$\mathcal{P} = \{p_i\}_{i=1}^n. \quad (4.2)$$

The core idea is to prioritize POIs where the difference between newly measured values and previously observed or predicted conditions is substantial, while omitting POIs whose expected change is negligible.

For example, with reference to Figure 4.3, the proposed methodology determines that poi_2 and poi_5 are less informative and can be skipped. As a result, the rover reduces its overall travel time and revisits more informative POIs (i.e., $poi_1, poi_3, poi_4, poi_6$, and poi_7) more frequently.

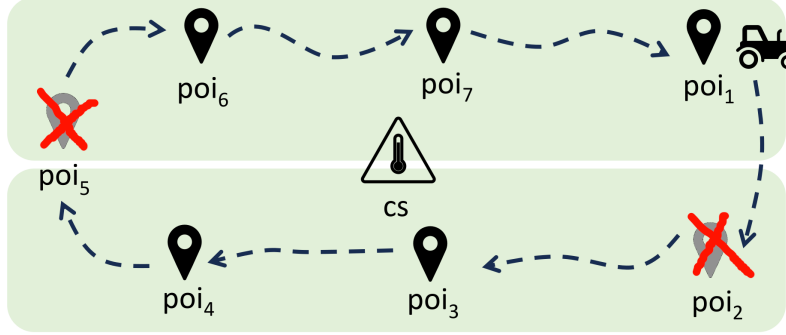


Figure 4.3: Greenhouse configuration where the proposed methodology identifies poi_2 and poi_5 as optional POIs to skip during the rover's trajectory.

The methodology leverages DQL, as discussed in Section 2.3.3, to estimate future rewards for stopping at each POI using past measurements.

We define the measurement trajectory as follows.

Definition 4.1 (Measurement trajectory) Given a set of POIs $\mathcal{P} = \{p_i\}_{i=1}^n$ inside the greenhouse, we define a measurement trajectory as the ordered sequence $\gamma = \langle p_1, \dots, p_n \rangle$ composed of the POIs in $\mathcal{P}$ in which the rover stops and measures the climatic variable of interest.

For the example in Figure 4.1, the measurement trajectory γ is $\langle poi_1, poi_2, \dots, poi_7 \rangle$. During the execution of this trajectory, two temporal components can be identified: the time required for the rover to move from a given POI to the next one, and the time it spends in a POI for its measurement. In particular, the following notation will be used:

- t_i : time spent at POI p_i for measurement;
- t_{ij} : travel time from POI p_i to p_j ;
- $\tau_\gamma = \sum_{i=1}^n t_i + \sum_{i=1}^{n-1} t_{ij}$: the total duration of a full trajectory.

The goal is to identify which POIs in γ can be skipped to save time and prioritize more informative locations. A POI where the rover stops for measurement is a “measurement stop”, and the optimal sequence of stops forms the “stop pattern”.

Definition 4.2 (Stop pattern) Given a measurement trajectory γ , we define a stop pattern σ as an ordered sequence of POIs extracted from γ where the agricultural rover must stop during its trajectory to collect the desired measurements.

The following condition holds between the duration of a measurement trajectory γ , and a stop pattern σ : $\tau_\sigma \leq \tau_\gamma$ where τ_σ is defined as:

$$\tau_\sigma = \sum_{p \in \sigma} t_i + \sum_{i=1}^{n-1} t_{ij} \quad (4.3)$$

The choice of POIs to include in a stop pattern is context-dependent (Definition 3.4).

Considering the definition of context (Definition 3.4), for this study, three contextual dimensions are crucial: (i) measurement time t , (ii) greenhouse climatic conditions from the central station cs in Figure 4.3, and (iii) the relative position of the POI with respect to the rover.

Within DRL terminology, the agent is the rover, the state combines its current position and context, and the reward is defined as follows.

Definition 4.3 (Reward) Given a POI $p \in \mathcal{P}$ representing the current position of the rover at time t , the reward associated with the stopping into a subsequent POI $p' \in \mathcal{P}$ is determined by the difference between the temperature measured or estimated in p' during the previous trajectory and the expected temperature in p' at time $t + t_{pp'}$, where $t_{pp'}$ is the time required to reach p' from p .

This reward guides the selection of the next measurement stop and determines the optimal stop pattern for the current context.

As illustrated in Figure 4.4, the DQL model takes past virtual sensor data and the current context as input and produces the optimal stop pattern for the next trajectory. These measurements are subsequently used to update the model.

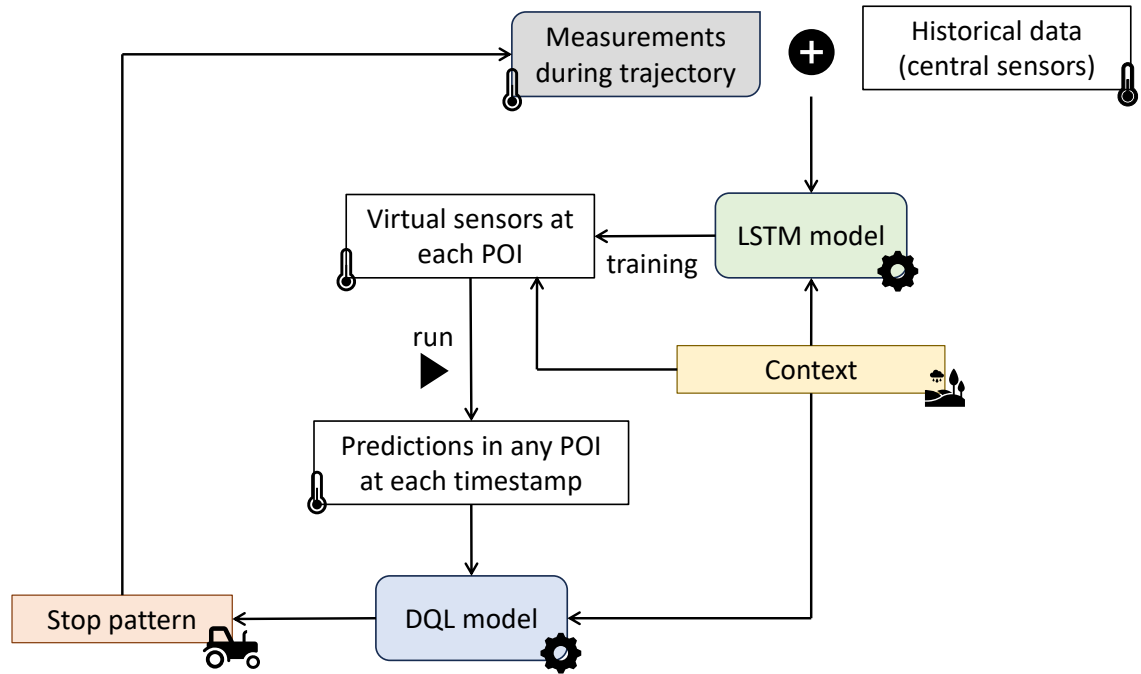


Figure 4.4: Diagram of the proposed DRL approach, based on DQL for context-aware trajectory planning.

The DQL model comprises two main components: the Q-Learning neural network (see Section 2.3.3.3) and the reward function. The former is used by the DQL to learn the value of an action in a particular state, and includes the following three layers: an input layer with 24 units and ReLU activation function (see Section 2.1.1), a hidden layer with 24 units and ReLU activation function, and an output layer with linear activation. The model is compiled with MSE loss and the Adam optimizer (see Section 2.1.4). Conversely, the reward function is defined as in Definition 4.3, and it considers the temperature difference in the same POI at the time of the previous passage and at the current time. These measurements can then be fed into the LSTM prediction model described in Section 4.3.2 to estimate conditions at unmeasured POIs or at times when the rover is elsewhere.

4.3.4 Methodology III: PPO-Based Energy-Aware Optimization

In the context of virtual sensing and robotic data collection, in [4], we take a step forward in defining the optimal stopping pattern by carefully combining the amount and significance of the collected data

with the *energy aspects* of the agricultural robot. In particular, we introduced a more complex model of the environment that considers the energy consumption associated with each path and the fact that some POIs represent not only data sources but also charging stations. Moreover, alternative paths may exist between one POI and another, with varying energy costs and benefits for monitoring and recharging opportunities. More precisely, the best stopping pattern is identified as a combination of a DRL approach and a PPO algorithm, described in Section 2.3.4, which are used together to balance the number and quality of measurements with the robot's autonomy. The reason for this selection is that, in this study, we are dealing with a complex trajectory optimization problem for a rechargeable robot operating in a greenhouse environment. DRL provides a data-driven approach to learning optimal decision-making policies in dynamic, uncertain environments. Among available DRL methods, we adopt the PPO algorithm, which offers a favorable balance between sample efficiency, exploration, and training stability. PPO's clipped surrogate objective prevents destabilizing policy updates, making it especially suited to robotic applications involving continuous learning. Using the PPO framework, the robot learns to adaptively schedule its movements, decide when to recharge, and select trajectories that maximize cumulative measurement reward while satisfying energy constraints.

Figure 4.5 illustrates an example of the mentioned enriched environment where some paths have been added with respect to Figure 4.1, arrows have been removed to denote the possibility to travel in both directions, and charging stations have been positioned at some POIs. For example, from poi_5 it is

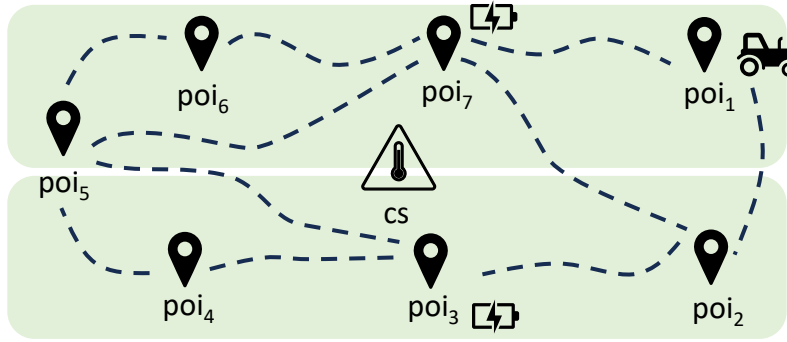


Figure 4.5: Example of enriched model environment where some POIs have been identified as charging stations and some direct paths have been identified.

possible to directly reach the charging stations in poi_7 and poi_3 , while from the charging station in poi_7 , it is possible to directly reach poi_2 without visiting poi_1 , or going back to poi_6 .

4.3.4.1 Formal Problem Definition and the PPO-Based Methodology

The environment under investigation is characterized by the following underlying hypothesis, which will be formalized as a whole in Definition 4.4.

- An agricultural robot periodically describes a trajectory during its farming operations that potentially touches all elements of a set $\mathcal{P}$ of predefined and well-identified POIs. The visit order is not predefined, and there are no precedence constraints. Moreover, some POIs in $\mathcal{P}$ could be skipped during a specific trajectory.
- Each POI $p \in \mathcal{P}$ could be reached directly or indirectly by passing through other POIs, so there could be more than one path connecting the same pair of POIs.
- There is no fixed direction in visiting POIs, namely, the robot can go forward and backward among the POIs.

- The robot is equipped with a device that can sample climatic variables during the execution of its tasks. The measuring activity is done during farming operations. Therefore, we assume the presence of a function: t , which returns for each POI p the time required to complete the farming operations, including the measuring task. Since the assumed temporal granularity defines a resolution of one minute, the duration $t(p)$ for each $p \in \mathcal{P}$ becomes the number of minutes needed to complete the task.
- The movement from one POI $p_1 \in \mathcal{P}$ to another POI $p_2 \in \mathcal{P}$ requires some time, which is represented by a function m that returns for each pair of POIs the amount of time needed to perform such a movement. Again, since we consider a temporal granularity of 1 minute, m returns the number of minutes required to complete such a movement.
- The robot has a limited amount of energy charge, which is also consumed by moving among POIs, so it needs to be recharged periodically. To simplify, we assume that the available and consumed energy are expressed as integers representing the battery life in minutes.
- Some POIs contain a charging station in which the robot battery can be charged. The charging activity takes some time to complete, and it does not overlap with farming/measuring operations. Without loss of generality, we assume that the charging time depends only on the charging station's current capabilities, but the model is general enough to accommodate a more complex function that also depends on the charging conditions, such as the current battery level. In other words, in practical scenarios, charging time depends on the battery level at the moment of recharge, and the robot may take less time to recharge if the battery is not fully depleted. Future work could focus on incorporating a dynamic charging-time model based on the robot's actual battery level to enable more accurate energy management. Integrating variable charging times into the trajectory-planning algorithm will improve scheduling efficiency and yield more realistic, effective operational strategies. In any case, the time required to complete a charging operation is represented by the number of minutes needed to complete it.
- The farming/measuring operations are orthogonal to charging operations, and a robot can stop in a POI to either measure, charge, or do both.

An environment characterized by these hypotheses can be formally represented as a graph in which the nodes represent POIs and the edges represent the paths connecting them.

Definition 4.4 (Environment Graph) *The environment characterizing the problem at hand can be described as a bi-directed labeled graph $G = (V, E, \mathcal{L}, \mathcal{E})$ where:*

- V is the set of nodes representing the POIs.
- $E \subseteq \{(u, v) : u, v \in V, u \neq v\}$ is the set of bi-directional edges representing the paths connecting the nodes.
- $\mathcal{L} : V \rightarrow \mathcal{P} \times \mathbb{N}^+ \times \mathbb{N}$ is a labeling function assigning to each node $u \in V$ a label $\langle p, t_p, c_p \rangle$ defined as follows:
 - $p \in \mathcal{P}$ is the identifier of the POI.
 - $t : \mathcal{P} \rightarrow \mathbb{N}^+$ is the function that represents the time required to complete the measuring task together with the farming task inside p . The notation t_p is used in place of $t(p)$ for not cluttering the notation,

- $c : \mathcal{P} \rightarrow \mathbb{N}$ indicates whether a given POI p contains a charging station or not, i.e., $c_p = 0$ means that p does not contain a charging station, while $c_p > 0$ represents the time required for the charging operation.
- $\mathcal{E} : E \rightarrow \mathbb{N}^+ \times \mathbb{N}^+$ is a function that assigns to each edge $e \in E$ a label $\langle m_{u,v}, e_{u,v} \rangle$ as follows:
 - $m : \mathcal{E} \rightarrow \mathbb{N}^+$ is the function that represents the time required by the robot to move between two adjacent POIs. In the following, for not cluttering the notation, we will use $m_{u,v}$ to denote the time spent to move from POI p_u to POI p_v . Since the edges are bi-directional we assume that $m_{u,v} = m_{v,u} \forall (u,v) \in \mathcal{E}$
 - $e : \mathcal{E} \rightarrow \mathbb{N}^+$ is the function that represents the energy required by the robot to move between two adjacent POIs, i.e., $e_{u,v}$ is the energy to be consumed to move from POI p_u to POI p_v . Since the edges are bi-directional we assume that $e_{u,v} = e_{v,u} \forall (u,v) \in \mathcal{E}$

As an instance, Figure 4.6 reports the graph notation for the environment introduced in Figure 4.5. In particular, three charging POIs can be identified (i.e., poi_3 , poi_5 and poi_7). The time required to reach each POI from another one can be different (i.e., from 15 to 45 minutes), while the time to be spent in each POI for measuring is 15 minutes. In this study, we assume that the energy consumed to traverse an edge is proportional to the travel time, with a unitary coefficient, so that its magnitude equals the travel time. However, more complex formulations can be adopted without altering the notation.

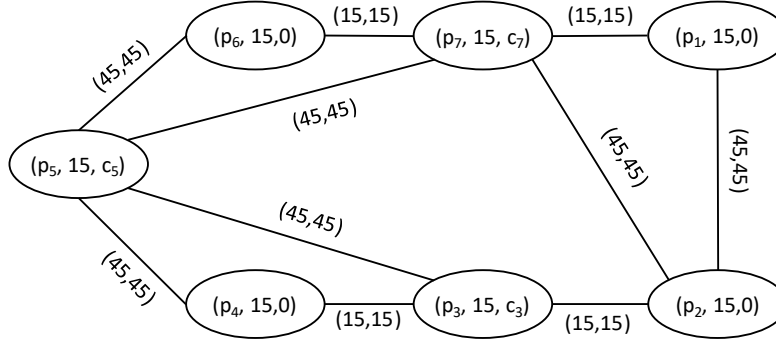


Figure 4.6: Example of environment represented through the labeled graph introduced in Def. Definition 4.4, a generic value c_i denotes a capacity value greater than 0.

Given the definition of environment graph, we can derive the definitions of *robot trajectory*, *measurement trajectory*, and *charging trajectory*.

Definition 4.5 (Robot trajectory) Given an environment graph $G = (V, E, \mathcal{L}, \mathcal{E})$ built starting from a set of POIs $\mathcal{P} = \{p_i\}_{i=1}^n$ inside a greenhouse, we define a robot trajectory as the ordered sequence $\gamma = \langle p_1, \dots, p_n \rangle$ of POIs in which the robot stops for doing a measurement of the climatic variables of interest (together with its farming tasks) or for charging.

Definition 4.6 (Measurement trajectory) Given a robot trajectory $\gamma = \langle p_1, \dots, p_n \rangle$, a measurement trajectory is a sub-sequence $\mu = \langle p_i, \dots, p_k \rangle \subseteq \gamma$ composed only of the POIs in γ where the robot stops to perform only a measurement.

Definition 4.7 (Charging trajectory) Given a robot trajectory $\gamma = \langle p_1, \dots, p_n \rangle$, a charging trajectory is a sub-sequence $\chi = \langle p_j, \dots, p_h \rangle \subseteq \gamma$ composed only of the POIs in γ where the robot stops for charging.

Given a robot trajectory γ together with its related measurement μ and charging χ sub-sequences, the time τ_γ required to complete the entire trajectory γ can be computed as:

$$\tau_\gamma = \sum_{i \in \mu} t_i + \sum_{j \in \chi} c_j + \sum_{i,j \in \gamma} m_{ij} \quad (4.4)$$

where $i \in \mu$ or $i \in \chi$ identifies the index of a POI inside the measurement or charging trajectory, respectively, while $i, j \in \gamma$ identifies a pair of consecutive POIs in γ .

With reference to the example in Figure 4.1, a possible robot trajectory could be represented by the sequence $\gamma = \langle p_1, p_2, p_7, p_5, p_6 \rangle$, where the measuring trajectory μ might correspond to γ , since we may decide to take a measurement in each touched POI, while the charging trajectory χ might be composed by p_7 only, since the robot might need only one charging operation to complete the whole trajectory.

Table 4.1: Summarization of used symbols.

Symbol	Meaning
$\mathbb{T}$	temporal domain
POI	General term for Point of Interest
poi_i	The i -th specific Point of Interest used in trajectories and models
$\mathcal{P}$	The set of all POIs: $\{p_1, \dots, p_n\}$
V	set of nodes representing POIs
E	set of edges between POIs
$\mathcal{L}$	node labeling
t_p	time for measuring in $p \in \mathcal{P}$
c_p	time for charging in $p \in \mathcal{P}$
$\mathcal{E}$	edge labeling
m_{ij}	time for moving from $i \in \mathcal{P}$ to $j \in \mathcal{P}$
e_{ij}	energy consumed for moving from $i \in \mathcal{P}$ to $j \in \mathcal{P}$
γ	robot trajectory
μ	measuring trajectory
ξ	charging trajectory
τ_γ	time for completing γ
x	contextual measure
$\delta = \langle d_1, \dots, d_n \rangle$	contextual dimensions
$\rho = \langle w_1, \dots, w_n \rangle$	context
$r(p, q)$	reward for going from p to q
ς	charging level of the robot
ℓ	position of the robot
$\sigma(p, q)$	checks if $p = q$

Given a set of POIs $\mathcal{P}$ that the robot should periodically sample, the aim of this paper is to identify the optimal trajectory while considering the battery level of the robot.

As already discussed, the choice of the sequence of POIs to include in the robot trajectory γ is not fixed, and it can change depending on the particular conditions (i.e., context) in which the measurements are performed. For this reason, we need to define the notion of measurement context as follows.

Considering the definition of context (Definition 3.4), for this problem, these main types of contextual dimensions are of interest: (i) the time instant i in which the measurement is performed, (ii) the general climatic conditions inside the greenhouse, represented by the values collected by the central climatic station cs in Figure 4.5, (iii) the relative position of a POI with respect to the current position of the robot, which is

important in terms of travel time and required charge to get there, (iv) the battery level of the robot and its position relative to the nearest charging station which is important to choose next POI to visit.

With reference to the DRL terminology, the agent is represented by the agricultural robot, and the notion of state is represented by its current node occupied inside the environment graph, together with the specific current context. For our purposes, the robot can be formalized as follows:

Definition 4.8 (Robot agent) *The robot agent is a tuple $\langle \varsigma, \ell \rangle$ where $\varsigma : \mathbb{T} \rightarrow \mathbb{N}$ is a function returning for each temporal instant $i \in \mathbb{T}$ the charging level of the robot battery, while $\ell : \mathbb{T} \rightarrow V$ is a function returning for each temporal instant $i \in \mathbb{T}$ the POI $p \in \mathcal{P}$ currently occupied by the robot.*

The final ingredient we need to formalize is the notion of reward used during the learning process to provide prompt feedback and encourage the agent to take the most suitable action.

Definition 4.9 (Reward) *Given a POI $p \in \mathcal{P}$ representing the current position $\ell(i)$ of the robot at time $i \in \mathbb{T}$, the reward associated with stopping in a subsequent POI $q \in \mathcal{P}$ is defined as follows:*

$$r(p, q) = \Delta_q(i + m_{pq}) - \sigma(p, q) - e_{pq} - \Gamma_{pq}(t) \quad (4.5)$$

This reward function is designed with two main priorities:

1. Promote safety and avoid critical conditions

- **Avoiding Stagnation:** A significant penalty is applied if the agent stays in the same place. This is captured by $\sigma(p, q)$ in Equation (4.5), which checks if p and q represent the same POI or not.
- **Battery Management:** Severe penalties are given for critical battery levels (i.e., it is too low to reach a charger, or it is empty). Conversely, a positive reward is provided for successfully reaching a charging station, especially when the battery is low. A penalty is also applied when the battery drops below a warning threshold, prompting the agent to seek charging early. These considerations about the battery level are captured by the term $\Gamma_{pq}(t)$ in Equation (4.5), which is based on the value returned by the function $\varsigma(t)$ in Definition 4.8.

2. Performance Optimization

- **Temperature Considerations:** The reward for visiting a new location is based on the difference between the temperature that was previously measured or estimated there and the temperature that is expected at that location when the agent arrives. A larger temperature difference indicates greater importance in visiting that location, leading to a higher reward for selecting it. This is captured by the term $\Delta_q(i + m_{pq})$ in Equation (4.5).
- **Time and Resource Efficiency:** The agent is penalized for the time taken for actions and the resources consumed, encouraging faster and more efficient operations. This is captured by the term e_{pq} in Equation (4.5), which measures the energy required to move from p to q .

This reward structure ensures the agent prioritizes safety while simultaneously learns to perform its tasks efficiently and effectively. It will be used in each state to identify the next best long-term action, namely the POI to choose as the next stop in the trajectory. Based on this, the best trajectory is the optimal sequence of stops for the current context. Notice that, since there may be multiple paths connecting the same pair of POIs, we use Dijkstra's algorithm to find the shortest path between all pairs of POIs.

The result of the proposed approach is the sequence of POIs in which the robot should stop to perform measurements, recharge, or both. The PPO agent was implemented in Python using the PyTorch deep

learning framework, following the reference implementation provided by the CleanRL library¹. The complete source code of this work is publicly available in the GitHub repository².

4.4 Experimental Results

In this section, we evaluate and compare three trajectory-planning approaches for greenhouse climatic sensing:

- Baseline fixed-trajectory approach presented in [3], where the agricultural robot always visits all POIs in a fixed order (see Section 4.3.2).
- DQL stop-pattern optimization proposed in [5] (see Section 4.3.3).
- PPO energy-aware trajectory planning proposed in [4] (see Section 4.3.4).

For a fair comparison, all methods presented were evaluated on the Verona greenhouse dataset (see Section 3.4) using standard error metrics discussed in Section 3.4.2).

4.4.1 DQL Stop-Pattern Optimization Results

In the baseline method, the robot traverses all POIs in a fixed sequence. Referring to the configuration shown in Figure 4.7, it can be observed that a complete measurement trajectory traverses the POIs in the following order: poi_1 , poi_2 , poi_3 , poi_4 , poi_5 , poi_6 , and poi_7 , taking 15 minutes for each movement and staying 15 minutes in each POI for collecting the data. Given 6 movements and 7 stops (15 minutes each), the total duration is 195 minutes (3 hours and 15 minutes).

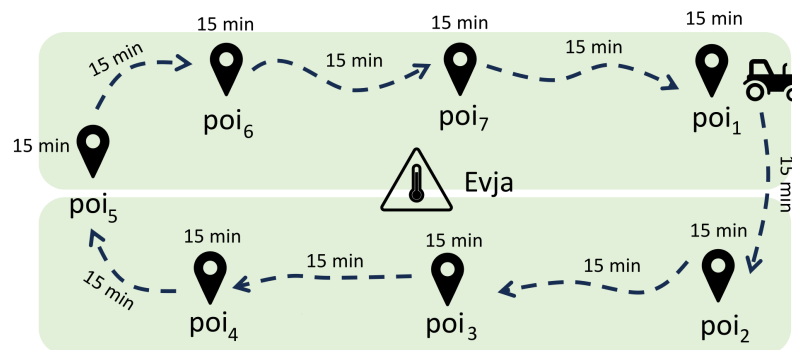


Figure 4.7: Setup of the real-world case scenario considered for the experiments.

Unlike the baseline approach, we propose a DQL-based model that selects a subset of POIs to visit based on expected temperature variation and contextual features. Table 4.2 demonstrates an example on August 27th, where 3 POIs (poi_3, poi_5, poi_6) are skipped due to low temperature variation. These skips save 45 minutes, enabling more frequent revisits to high-variation POIs such as poi_1 .

Table 4.3 summarizes the aggregated time metrics comparing DQL with the baseline. The results indicate that the proposed approach reduces the average trajectory duration by approximately 35% over the baseline in our experimental setting.

¹<https://github.com/vwxyzjn/cleanrl>

²<https://github.com/AshrafSharifi/RechargeableRobot-Trajectory-PPO>

Table 4.2: Example of stop pattern suggested by the proposed approach.

Pol	Status	Temperature variation	Visit time	Measured Temperature
1	Visited	3.55	2021/8/27 11:45	18.15
2	Visited	3.36	2021/8/27 12:15	20.08
3	Skipped	0.03	-	21.97
4	Visited	5.90	2021/8/27 13:00	22.07
5	Skipped	1.05	-	20.57
6	Skipped	1.65	-	23.14
7	Visited	7.32	2021/8/27 14:00	26.08

Table 4.3: Comparison between the proposed approach and the baseline [3].

Metric	Proposed Method	Baseline
Minimum trajectory duration	1:45:00	03:15:00
Maximum trajectory duration	2:30:00	03:15:00
Average trajectory duration	2:06:00	03:15:00

4.4.2 Experiment Results: PPO Energy-Aware Trajectory Optimization

Figure 4.8 depicts the environment graph with 7 POIs used in this experiment. In order to avoid cluttering the graph, we adopt the following convention: each edge (p, q) reports only the time required to reach POI q from POI p (m_{pq}), the required energy is expressed in minutes of battery consumption and is assumed to be numerically equal to the travel time. A small battery icon identifies the nodes containing a charging station, namely p_3 , p_5 , and p_7 in Figure 4.8; both the measurement time (t_p) and the charging time (c_p) are fixed at 15 minutes for each POI p .

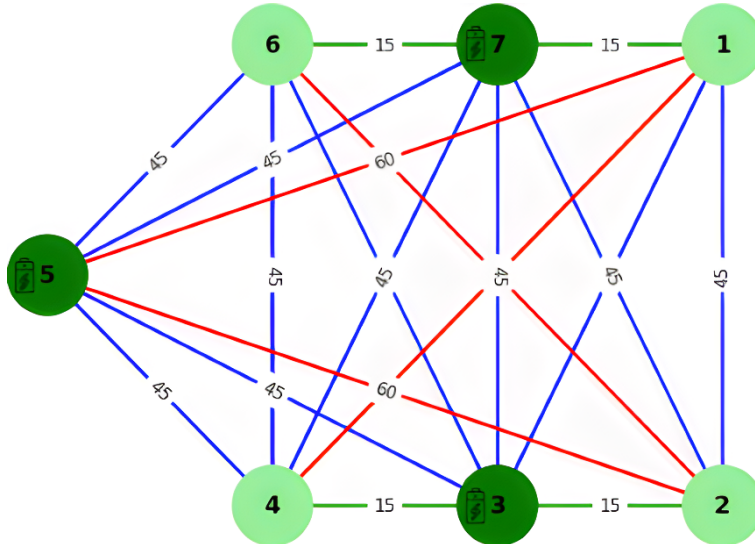


Figure 4.8: Greenhouse environment considered for experiments.

Given this configuration and the historical context in which the measurement trajectories are to be performed, we compute the best trajectory using the proposed PPO approach described in Section 4.3.4. Figure 4.9 illustrates an example of the algorithm's behavior, where for each step, the start and end POI are identified by a green box indicating the timestamp of the starting time, the remaining battery level, and

the reward associated with this movement action. The robot starts in p_5 with a full battery level of 300; in this situation, the algorithm identifies the best action to perform the movement towards p_4 (Step 1), where the robot arrives after 45 minutes, stays for 15 minutes, consuming a total of 60 units of energy and obtaining a reward of 19.28. Similarly, in Step 2, the robot has enough battery to reach each POI without any problem, and the algorithm identifies the movement towards p_1 as the best action. In Step 3, p_6 is the best POI to visit, but since there is no direct path from p_1 to p_6 , the robot passes through p_7 without performing any measurements or charging operations. Indeed, the end time reported in p_6 is 45 minutes after the departure from p_1 : 30 minutes for traveling and 15 minutes in p_6 for measuring and other farming tasks. Conversely, in Step 4, the algorithm decides that the robot needs to be recharged (its battery level is below a predefined safety threshold of 140 units, empirically selected to prevent mid-trajectory depletion). Therefore, the next POI to visit should be identified among p_3 , p_5 , and p_7 , because they all contain a charging station. In particular, the algorithm chooses p_3 because its reward is greater than that of the other two: even though it is farther away than p_7 , it is more interesting for the estimated temperature difference component. It is worth noting that, in this case, the reward associated with the action is much greater than the other, as the robot can stop at the same POI to perform two operations: measuring and charging. In other words, in this trajectory, p_3 will become part of both the measuring and the charging sub-trajectories.

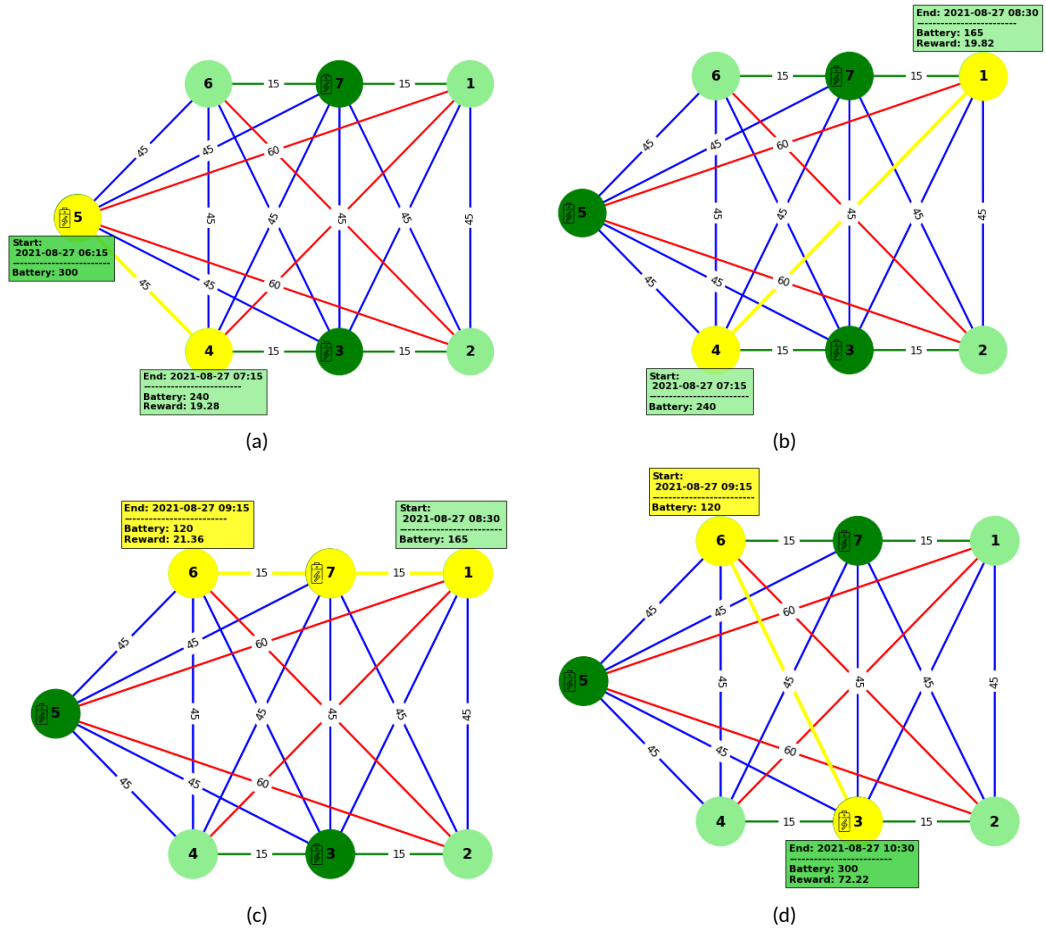


Figure 4.9: A sample of the suggested trajectory produced by the proposed method: (a) Step 1, (b) Step 2, (c) Step 3, and (d) Step 4.

The proposed approach can be applied to various environments with different configurations, thus leading to different trajectories. Figure 4.10 compares the results obtained from the same starting situation

(i.e., initial POI, timestamp, and battery level) but in a slightly different environment regarding the available connections between POIs. In the original environment, the suggested trajectory was p_1, p_6 (through p_7), and then p_3 . Conversely, removing the edge from p_7 to p_6 determines that the best POI to visit after p_1 becomes p_3 , followed by p_6 . In other words, the order of visits between p_6 and p_3 has been reversed, since p_3 is now on the shortest path to p_6 , and the robot can exploit this. This experiment demonstrates the adaptability of the PPO-based planner to dynamic changes in the environment topology.

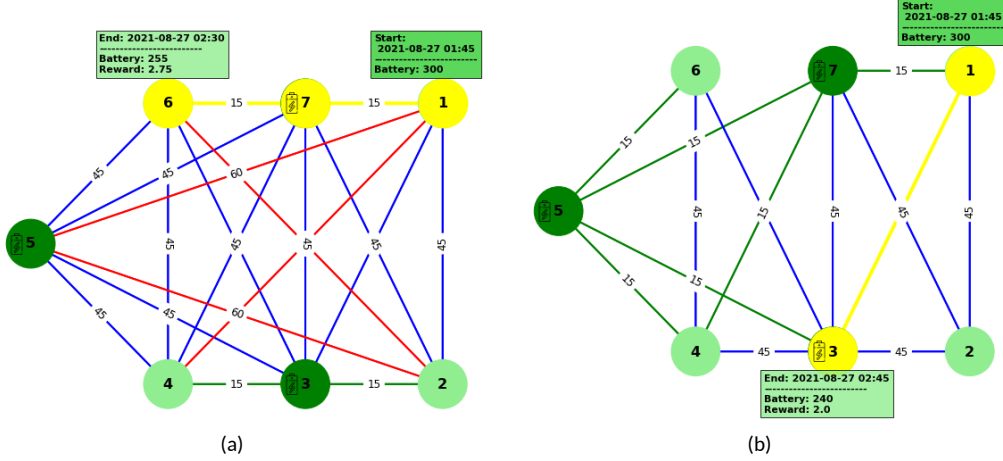


Figure 4.10: Application of the approach in: (a) Original environment, and (b) Changed environment configuration.

As another experiment, we evaluate the robot's battery level during the suggested trajectories. First, we conducted tests to examine the impact of reducing the initial battery capacity on the number of times the robot had to stop for charging. The results showed that when the robot's full charge was set to 300 units, it made 257 charging-station stops across all the considered 1,656 trajectories. However, when the full charge was reduced to 150 units, the number of stops increased significantly to 406. Secondly, considering an initial battery level of 300 and the trajectories produced by the method in [5] under identical environmental and initial conditions, we examined how many times the robot is located at POIs equipped with a charging station during a critical moment (i.e., its battery level is less than the threshold) and so can solve this situation without stopping out of charge in the middle of a trajectory. Our experiment shows that, with the previous approach, the robot needed to be charged in approximately 168 cases, whereas in 83 of these cases it was not located at POIs containing a charging station.

4.4.3 Predictive Performance Comparison

The results presented in the previous sections demonstrate that the proposed approaches reliably identify stop patterns across a variety of scenarios. Furthermore, the trajectories generated using these patterns achieve a higher cumulative reward than trajectories generated by always visiting all POIs. Although these results are encouraging, the primary objective of our methodology is to improve the predictive performance of the deep learning model, with particular emphasis on estimating the temperatures of different POIs within the greenhouse.

To assess the effectiveness of the proposed approaches with respect to this objective, we use the stop patterns they generate to collect a new dataset of measurements, which we then use to train the LSTM model. The resulting estimation accuracy is compared against a baseline in which the LSTM is trained on measurements collected through complete trajectories, as proposed in [3].

Table 4.4 reports the average estimation errors for the examined approaches, both per POI and aggregated over all POIs. The column *Visits* indicates how many times each POI is visited during the

training phase, highlighting that the proposed approach may visit POIs a varying number of times depending on the criteria it must satisfy.

We further compare the prediction accuracy of the deep learning model from [18] for temperature estimation when trained on three different types of historical data. All methods were evaluated under identical environmental configurations, hyperparameter settings, and training protocols to ensure a fair comparison.

- Measurements collected using the trajectories identified by the PPO method discussed in Section 4.3.4
- Measurements collected using DQL approach, which does not account for charging requirements and only allows skipping certain POIs in direct trajectories, discussed in Section 4.3.3
- Measurements collected by always executing complete trajectories, as suggested in Section 4.3.2

For each POI, we report the prediction error of the trained DL model, in terms of MAPE, MAE, and MSE, along with the number of times the POI is visited in the corresponding trajectories.

Table 4.4: Comparison of estimation errors obtained by the LSTM model when trained using trajectories generated by: (i) the PPO-based approach [4], (ii) the DQL-based approach [5], and (iii) the baseline complete trajectories [3].

POI	PPO-based Trajectories				DQL-based Trajectories				Baseline (Complete Trajectories)			
	# Visits	MAPE (%)	MAE	MSE	# Visits	MAPE (%)	MAE	MSE	# Visits	MAPE (%)	MAE	MSE
1	419	5.1	0.93	2.2	343	7.7	1.3	4.4	560	8.8	1.4	4.1
2	395	5.5	1.0	2.5	241	5.7	1.0	2.6	560	6.1	1.2	2.6
3	396	5.6	1.0	2.6	323	7.2	1.3	4.6	520	8.6	1.5	4.7
4	356	5.9	1.2	3.2	53	7.3	1.3	4.6	560	8.3	1.5	4.8
5	383	5.2	1.0	2.6	62	6.8	1.3	3.6	520	7.5	1.4	3.8
6	384	5.8	1.2	3.7	87	8.2	1.6	6.0	560	8.3	1.6	5.3
7	405	5.1	1.0	2.5	161	6.8	1.3	4.7	560	6.9	1.3	3.7
All	2738	5.4	1.1	2.7	1270	7.1	1.3	4.3	3840	7.8	1.4	4.1

The results in Table 4.4 show that the PPO-based approach provides the most accurate temperature predictions, achieving the lowest or comparable error values across most POIs. This improvement is largely due to its ability to generate a richer, more informative training set by revisiting POIs more frequently and flexibly than other methods.

The DQL-based method outperforms the baseline across several POIs, especially when it allocates more visits, but its overall accuracy remains lower than that of the PPO approach. Compared to the complete-trajectory baseline, the DQL strategy benefits from skipping certain POIs and shortening cycles, allowing additional visits at selected locations.

This result indicates that reward-optimized trajectories also tend to improve the spatio-temporal coverage of the collected data, thereby directly benefiting the learning process of the LSTM model.

4.5 Discussion and Comparative Analysis

The progression from a fixed-trajectory baseline to DRL frameworks represents an evolution from deterministic sensing to autonomous, information-driven exploration. This section synthesizes the performance of the three methodologies: the Fixed Baseline, DQL, and PPO, by analyzing the critical trade-offs between predictive fidelity and operational viability.

4.5.1 Information Density and Prediction Accuracy

The main objective of implementing mobile optimization is to improve the Signal-to-Noise Ratio of the training data provided to the LSTM estimation model. Our results indicate that the quality of the measurements significantly influences model performance more than just the number of observations we collect.

- **The Redundancy Problem:** The baseline approach (MAPE: 7.8%) collected the largest dataset (3,840 visits). However, by visiting every POI regardless of environmental variance, the data became saturated with redundant readings that provided little important information to the LSTM, limiting its learning potential.
- **Utility-Driven Sampling (DQL):** By selectively skipping low-variance POIs, the DQL approach reduced total visits by roughly 67% (1,270 visits) while actually improving accuracy to 7.1% MAPE. This confirms that focusing on high-utility transitions, where temperature gradients are high, allows the LSTM to model environmental dynamics more effectively.
- **The Optimal Balance (PPO):** PPO achieved the highest accuracy (5.4% MAPE). PPO's more balanced reward formulation enabled a higher sampling rate (2,738 visits) while still prioritizing POIs based on information gain and available energy. This performance difference is also influenced by the structure of the navigation environment: because the underlying graph is bidirectional, the agent can move both forward and backward between POIs, providing greater flexibility in adjusting its trajectory. PPO exploits this flexibility more effectively than DQL, allowing it to revisit informative locations when necessary rather than committing to overly aggressive skipping behavior.

While predictive accuracy is the academic goal, operational survivability is the practical requirement for agricultural robotics. The shift from DQL to PPO necessitates an increase in model complexity to handle the “real-world” constraints of the greenhouse.

As noted in Section 4.1, detailed robot control and real-time obstacle avoidance are outside the scope of this chapter. However, the proposed graph-based DRL methods give some indication of how the framework would react to unexpected situations. If a path becomes blocked or the rover experiences a partial failure, this can be represented as a change to the environment graph G , such as the removal of an edge or node. As shown in the topological-adaptation experiment (Section 4.4.2), the PPO planner can adjust its decisions when the graph structure changes, selecting new feasible routes in response. Similarly, long-term battery degradation, which reduces effective capacity or increases discharge rates, is naturally handled by the PPO multi-objective reward formulation: because the agent continuously monitors battery state and is penalized for unsafe operation, a degraded battery would lead to earlier recharging decisions and more conservative routing. While these mechanisms do not replace dedicated obstacle-avoidance or fault-tolerant navigation modules, they suggest that the sensing policies can adapt when upstream systems signal blocked paths, reduced mobility, or declining battery health.

4.5.2 Model Complexity and Efficiency

Beyond predictive accuracy, the methods differ substantially in their computational and operational complexity. The DQL model improves time efficiency by skipping low-value POIs, reducing the average trajectory duration to 2 hours and 6 minutes compared to the baseline's fixed 3 hours and 15 minutes. However, its optimization focuses primarily on maximizing measurement utility (temperature difference), without considering broader operational constraints.

In contrast, PPO operates within a more expressive environment model that includes bidirectional edges and multiple alternative paths between POIs. This richer graph structure, combined with explicit

shortest-path computation using Dijkstra's algorithm, provides the agent with greater flexibility to adjust its trajectory. As a result, PPO is able to maximize measurement collection efficiency while respecting the robot's energy and autonomy constraints. The transition from DQL to PPO therefore introduces a significant increase in model and environment complexity.

- **DQL Complexity:**
 1. **Simpler architecture:** DQL relies on a Q-learning neural network trained on historical data to estimate Q-values for each state-action pair. This makes it relatively easy to implement and validate.
 2. **Single-objective reward:** The reward function is defined solely by the temperature difference at the next POI and the travel time required to reach it. No additional operational constraints are modeled.
 3. **Lack of energy awareness:** A major limitation is the absence of battery management. In 83 of 168 critical situations, the DQL agent produced trajectories that left the robot needing to charge when it was not at a POI equipped with a charging station, creating a risk of operational failure.
- **PPO Complexity:**
 1. **Richer environment model:** PPO extends the environment to incorporate robot autonomy, bidirectional navigation, and explicit energy management, enabling more realistic decision-making.
 2. **Multi-objective reward function:** The reward formulation balances safety and performance. It penalizes unsafe battery levels, rewards reaching charging stations when necessary, and simultaneously encourages maximizing temperature difference while minimizing time and energy costs.
 3. **Actor-Critic architecture:** PPO employs an Actor-Critic framework that provides a stable balance between exploration, sample efficiency, and training robustness. The clipping mechanism in the PPO objective prevents destabilizing policy updates, which is particularly valuable in complex robotic environments. The actor and critic networks (with two and three hidden layers, respectively) are trained concurrently.

While DQL improves efficiency over the fixed baseline through importance-based planning, its simplicity comes at the cost of operational reliability. PPO achieves superior predictive accuracy and robust trajectory planning by adopting a significantly more complex framework that integrates energy constraints, bidirectional navigation, and multi-objective optimization. This added complexity enables PPO to avoid failure cases encountered by DQL and to schedule trajectories more effectively.

4.6 Conclusion

Chapter 4 introduces the dynamic data acquisition layer of the greenhouse DT by incorporating autonomous agricultural rovers as mobile sensing agents. This chapter illustrates how transitioning from static sensor grids to mobile, intelligence-driven sampling significantly enhances microclimate mapping while lowering infrastructure costs.

The key findings and contributions are outlined below.

- **Optimized Data Quality:** The results indicate that the informational value of measurements plays a crucial role in model accuracy. By focusing on locations with high thermal variance, the system

minimizes redundant readings and effectively captures the environmental gradients most relevant to subsequent modeling.

- **Performance of DRL Strategies:** The DQL policy enhanced sampling efficiency by selectively skipping low-variance points. This approach resulted in a 35% reduction in trajectory duration and a decrease in temperature prediction error from 7.8% to 7.1% in terms of MAPE.

PPO proved to be the most effective strategy, achieving the highest predictive accuracy (5.4% MAPE). It successfully balanced multiple objectives, such as information gain, energy consumption, and proximity to charging stations.

- **Operational Reliability:** Incorporating energy-aware constraints into the PPO framework addressed a key limitation of earlier models, ensuring that the rover remained within safe battery thresholds and avoided mid-trajectory failures.
- **Enhanced Virtual Sensing:** When the optimized trajectories were integrated into the virtual sensing models developed in Chapter 3, the resulting environmental state estimates improved, confirming the value of dynamic, information-driven data acquisition.

This chapter presents a methodology for converting a standard robotic platform into an intelligent sensing agent. By facilitating adaptive, efficient, and reliable data collection, the system enhances the environmental foundation of the DT and lays the groundwork for the biological modeling layers discussed in the following chapters.

Chapter 5

Modeling the Growth of Rainbow Trout for the Digital Twin of an Aquaculture Facility

Chapter contents

5.1	Introduction	66
5.1.1	Case Study: Precision Fish Farming (PFF) and Rainbow Trout.....	69
5.2	Related Work	70
5.2.1	Mechanistic Approaches	70
5.2.2	Data-Driven Methods	71
5.2.3	Hybrid Data-driven, Model-driven Approaches.....	71
5.3	Problem Formalization and Proposed Method	72
5.3.1	Mechanistic Component: The Bioenergetic Model.....	72
5.3.2	Feature Engineering and Data Pre-processing	73
5.3.3	Machine Learning Architectures.....	74
5.3.4	Advanced Learning Paradigms	74
5.4	Application Setup and Data Description	76
5.4.1	Farm Infrastructure and Management	76
5.4.2	Sensor Deployment and Environmental Monitoring	76
5.4.3	Biomass and Operational Data	77
5.5	Experimental Results.....	77
5.5.1	Experiment Setup	77
5.5.2	Preprocessing of Weight Measurements.....	78
5.5.3	Logarithmic Transformation and Target Definition	79
5.5.4	Implementation Details and Hyperparameters	80
5.5.5	Model Architecture Comparison	81
5.5.6	Evaluation of the Hybrid Framework and Baseline Comparisons	81
5.5.7	Evaluation of Incremental Learning and Data Efficiency.....	85
5.5.8	Performance and Training Data Requirements of the Proposed Hybrid Approach .	86
5.5.9	Interpretability and Feature Importance Analysis	87
5.5.10	Framework Generalization and Operational Scalability	88
5.6	Conclusion.....	88

5.1 Introduction

Modeling the behavior of complex systems is a central challenge across many application domains, where accurate models are required to support prediction and operational optimization. Such systems are typically governed by multiple interacting factors that evolve in nonlinear and time-varying ways. In addition, individual units within a system may exhibit variability, and their behavior can change over time.

In this chapter, a biological application is considered as a representative and challenging case study to illustrate these general modeling principles.

Building on the previous chapters, this work extends the DT paradigm beyond environmental monitoring to the modeling of biological systems. Earlier chapters focused on virtual sensing in greenhouse environments, where autonomous robotic sampling and NN networks were used to estimate spatially distributed climate variables. These systems, while complex, were primarily governed by physical processes that were relatively homogeneous across space and time. In contrast, the present chapter shifts the modeling target from the environment to the organisms inhabiting it. Biological systems introduce additional layers of complexity: individuals of the same species may follow different growth trajectories, their physiological states evolve dynamically, and their interaction with the environment is inherently bidirectional. As a result, the collected data must inform not only a model of the environment but also a model of the organisms themselves. This transition motivates the adoption of hybrid mechanistic-ML approaches that capture both biological variability and environmental influences. Here, the thesis transitions from purely data-driven environmental modeling to the full hybrid DT framework. The mechanistic bioenergetic model provides a biologically grounded structure that complements the learned temporal patterns.

These challenges motivate the use of modeling frameworks that can operate continuously and adapt to changing conditions. In this context, DTs provide a unifying paradigm that links computational models with real-world systems. DTs are increasingly adopted across scientific, engineering, and operational domains to simulate physical objects in their environments. A DT is more than a mathematical model. It is a dynamic, real-time virtual replica continuously updated with data from its physical counterpart [113]. Significantly, the virtual model evolves by changing state and updating its structure as new data arrives. This dynamic nature enables DTs to model biological processes effectively, provided suitable model adjustment methods are available. To achieve this, DTs rely on robust models that typically fall into two categories. Mechanistic models use domain-specific scientific knowledge and are often described by differential or variational equations to explain system behavior. In contrast, data-driven models identify relationships between variables by learning patterns directly from observed data, without explicitly considering underlying processes. While mechanistic models offer explainability and theoretical generalization, they require extensive parameter calibration for each biological object [114, 115, 116] and may not capture unknown dynamics [117]. On the other hand, purely data-driven machine learning approaches, like the one illustrated in Figure 5.1, have achieved impressive results in tasks such as pattern recognition and prediction, especially when physical modeling is difficult or infeasible. However, these methods typically require large, high-quality datasets and may produce biologically inconsistent results [118, 119].

Therefore, neither mechanistic nor purely data-driven models alone can adequately represent the dynamics and variability observed in complex systems. Thus, there is a clear need for hybrid modeling approaches that integrate mechanistic model-driven knowledge with data-driven ML techniques. The aim is to adapt equations from the mechanistic approach to the specific characteristics of a given system, as encoded in the collected data. The combination of these two approaches can also be applied incrementally, allowing the hybrid model to evolve and correct itself as new data becomes available.

Recent advances in Scientific Machine Learning (SciML) [120] support this direction by combining established scientific models with ML design to create predictive systems that are both accurate and

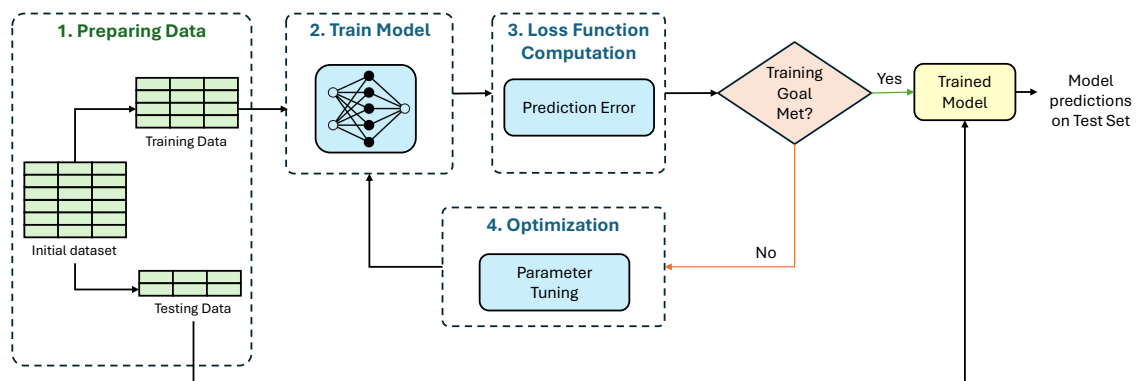


Figure 5.1: Purely data-driven machine learning workflow.

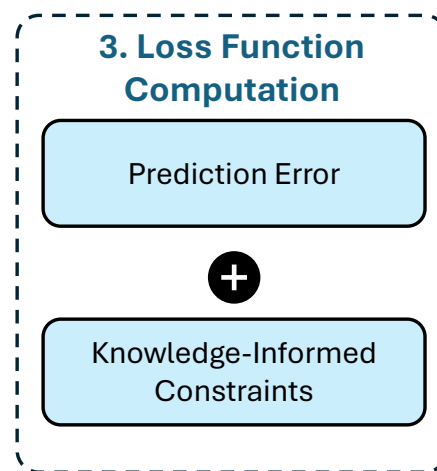


Figure 5.2: Loss function of knowledge-informed machine learning

biologically consistent [121]. A common implementation within SciML is the Knowledge-Informed Machine Learning (KIML) model, which embeds prior scientific knowledge into the ML architecture or training process. This knowledge, from fields such as physics, chemistry, biology, or engineering, can take the form of governing equations, conservation laws, or empirical relationships. In these frameworks, the training objective often extends beyond the conventional accuracy-driven objective used in purely data-driven methods, shown in Figure 5.1, where the model is optimized only to minimize the error between predictions and observed data. More specifically, a KIML framework is obtained by replacing the purely data-driven loss function shown in Figure 5.1 with a knowledge-based component, as depicted in Figure 5.2. In this way, the loss function of KIML includes additional terms to enforce compliance with established scientific or mechanistic constraints, such as equations, balance laws, or boundary conditions, alongside standard data-fitting components. This integrates data-driven and knowledge driven objective into the training process, enabling the model to produce predictions that are consistent with empirical data and reflective of known system behavior.

However, such architectures often require close integration between the mechanistic model and the learning algorithm, which significantly increases implementation and computational complexity. As highlighted in [122], these frameworks often experience optimization instability and acute hyperparameter sensitivity due to competing objectives within the loss function. Moreover, this strict coupling limits adaptability in real-world applications, making it challenging to modify the system to evolving operational demands or varied biological datasets.

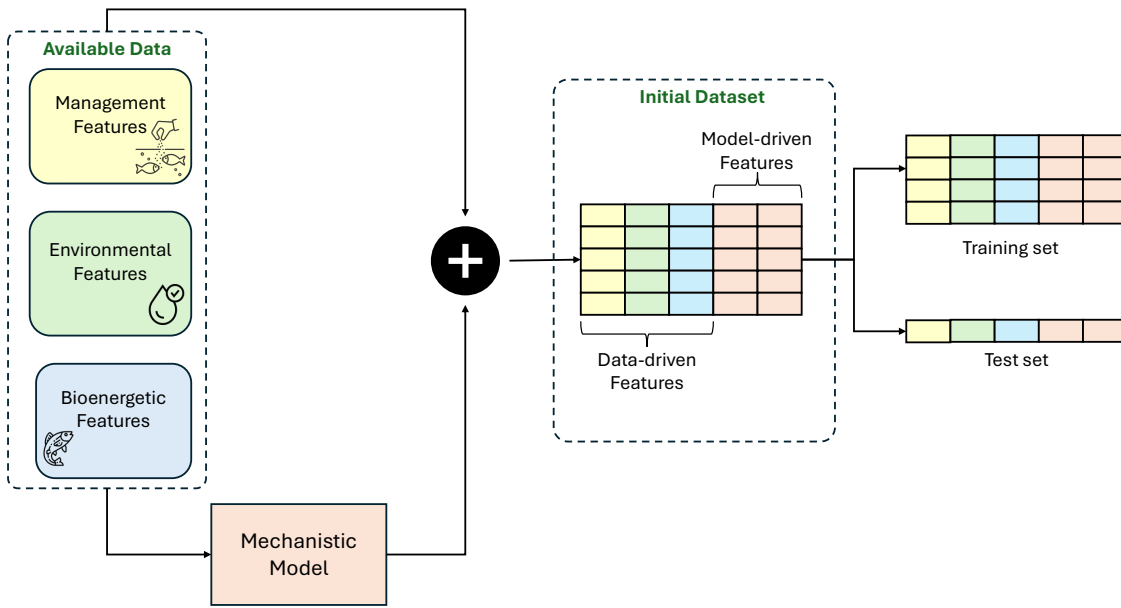


Figure 5.3: Enrichment of the input dataset implemented by the proposed hybrid approach

In this chapter, we propose an alternative hybrid framework, based on feature space enrichment. Unlike frameworks that impose constraints via the loss function, our method leverages the mechanistic model to directly enhance the input space. By transforming raw time series data into a more informative feature set, we provide the neural network with a richer context for learning without constraining the optimization landscape. This “lightweight” integration preserves the robustness of standard deep learning optimizers while ensuring biological consistency. Consequently, our framework is uniquely suited for commercial deployment, where computational stability and ease of retraining are critical requirements. This is achieved by replacing the standard “Data Preparation” block of a traditional data-driven framework (Figure 5.1) with our proposed enrichment module shown in Figure 5.3.

In machine learning, adding new features to the training set is generally not equivalent to introducing new constraints during optimization. Indeed, whereas additional constraints typically reduce the optimizer’s degrees of freedom, additional features usually increase the model’s flexibility by providing more information to explain the target, potentially enriching the hypothesis space the model can explore. However, when the added features are derived from known physical or biological laws, as in our proposed hybrid approach, they can substantially improve the learning process by making good solutions easier to identify and poor ones less likely to emerge. Consider, for example, a case in which, instead of providing only raw temporal information, one supplies $\sin(\omega t)$ and $\cos(\omega t)$, knowing that the underlying process is periodic. In this situation, rather than imposing a hard periodicity constraint, as is often done in SciML, one can bias the model toward learning periodic patterns by using a more informative input representation. More generally, incorporating features that encode prior knowledge can help the model learn the appropriate mapping more effectively. By contrast, constraint-based approaches such as those used in SciML explicitly restrict the set of admissible outputs. In the context of developing a DT of a biological system, the mechanistic model captures general principles that describe average behavior but must still be refined with data to account for the specific characteristics of each individual. In this setting, leveraging mechanistic knowledge during feature construction may be more effective than imposing it as a constraint during optimization. We demonstrate the utility of this framework through a case study in Precision Fish Farming (PFF). Aquaculture is one of the fastest-growing food production sectors worldwide [123]. Accurate predictions of fish weight

distribution and total biomass are very important for optimizing feed supply and planning biomass transfer among tanks and harvest. However, estimating fish size distribution from direct sampling is time-consuming and stressful for fish and may yield biased results due to the small sample size relative to the stocked population [124]. By combining bioenergetic mechanistic models with ML, our hybrid approach provides a scalable, non-invasive method for simulating dynamic growth under fluctuating environmental conditions.

To navigate the implementation and validation of this hybrid strategy, the remainder of this chapter is organized in a progressive framework. After introducing a case study of this work in Section 5.1.1, Section 5.2 reviews the state-of-the-art in aquaculture modeling. In this context, Section 5.3 formalizes the mathematical integration of the bioenergetic model equations with deep learning, detailing our specific feature-space enrichment pipeline and the composite bidirectional LSTM-CNN network architecture. Section 5.4 then grounds the study in a real-world pilot facility in Preore (Trento), detailing the continuous telemetry streams from automated multi-parameter water sondes and the non-invasive infrared Biomass Daily system used to generate our dataset. Section 5.5 provides a rigorous evaluation of the framework through an expanding-window incremental learning setup, benchmarks the hybrid model against purely mechanistic and data-driven baselines, and evaluates the internal reasoning of the predictive network using SHapley Additive exPlanations (SHAP). Finally, Section 5.6 concludes the chapter by summarizing the core insights into biological regularization, data efficiency, and the operational implications of this hybrid method.

5.1.1 Case Study: Precision Fish Farming (PFF) and Rainbow Trout

Having established the hybrid modeling framework, we now focus on a practical application in PFF, an emerging paradigm in modern aquaculture. PFF aims to exploit advanced sensing, data analytics, modeling, and control technologies to support the monitoring, optimization, and automation of fish farming operations [2]. Rather than relying on traditional experience-based management, PFF promotes knowledge-driven practices based on objective, data-informed decision-making.

The global population is expected to reach 9 billion by 2050, placing increasing pressure on sustainable food production [125]. Fisheries and aquaculture represent rapidly growing food production sectors, witnessing an average annual growth rate of 6.7% over the past three decades [123]. They provide high-quality proteins and essential nutrients, which are critical for global food security, nutrition, and economic development. Within this context, PFF addresses challenges such as feed scarcity, environmental impacts, and animal welfare, while leveraging technological innovation and global collaboration.

As illustrated in Figure 5.4, a key concept in PFF is the **Observe-Interpret-Decide-Act (ODIA)** cycle [2]:

1. **Observe:** Continuous monitoring of fish and environmental conditions;
2. **Interpret:** Extraction of meaningful insights from collected data;
3. **Decide:** Informed decision-making based on interpreted information;
4. **Act:** Implementation of decisions to optimize fish health, growth, and welfare.

Within PFF, precise prediction of fish weight is crucial, as it enables farmers to optimize feeding strategies, reduce waste, support stock management, and improve productivity and sustainability. Rainbow trout (*Oncorhynchus mykiss*), a cornerstone species in European aquaculture, provides an ideal testbed for this challenge. Originally translocated from the United States to Europe in the late 19th century [126], rainbow trout has become one of the most widely farmed freshwater species due to its adaptability, robust growth, and high market demand. In Italy alone, it accounts for approximately 25.5% of total aquaculture value and nearly 69.9% of the farmed fish volume, highlighting its economic importance. Its widespread availability

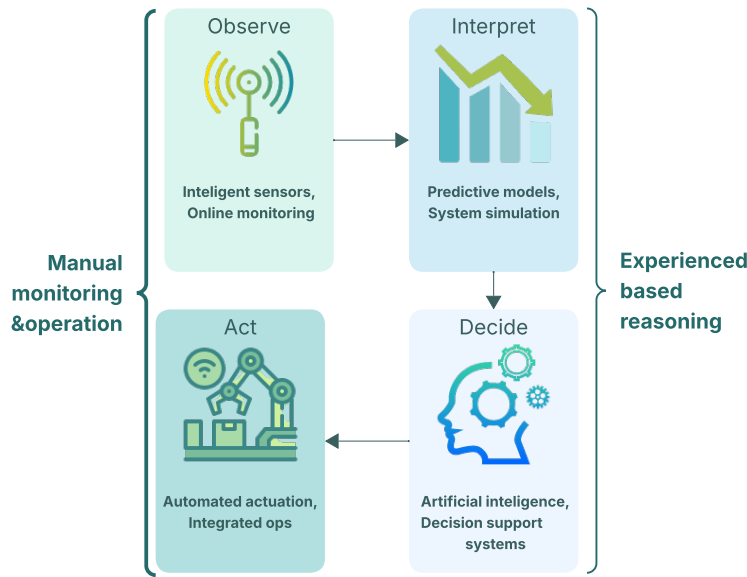


Figure 5.4: A cyclical representation of PFF inspired by[2].

in multiple product forms, fresh, smoked, and frozen, supports a diverse consumer market, and it is a significant export commodity within the European Union [127].

Traditional methods for estimating fish weight rely on manual measurements, which are labor-intensive and time-consuming and can stress the fish [124]. Mechanistic growth models capture valuable biological and environmental relationships, but they require extensive calibration, are sensitive to varying conditions, and may fail to capture all complex dynamics [9, 128]. Purely data-driven approaches, such as ML, can learn complex patterns from available data but typically require large, high-quality datasets that may not always be available.

These challenges make rainbow trout weight prediction an ideal scenario for hybrid mechanistic–ML modeling, which combines mechanistic insights with data-driven flexibility. By integrating domain knowledge with observational learning, hybrid models can improve prediction accuracy, adapt to environmental variability, and support sustainable aquaculture decision-making under real-world conditions.

5.2 Related Work

This section reviews mechanistic, data-driven, and hybrid modeling approaches, with emphasis on PFF, as our case study concerns an aquaculture DT. While these methods have been widely studied in other fields, the discussion here centers on aquaculture applications, where practical constraints motivate the development of improved hybrid strategies.

5.2.1 Mechanistic Approaches

As mentioned in Section 5.1, the main objective of PFF is to help fish farmers make informed decisions. These decisions improve animal welfare and productivity by increasing the accuracy, precision, and consistency of farming operations [2]. PFF achieves this by using predictive models that incorporate environmental factors and fish physiological responses, collected through modern sensors.

A key contribution of PFF is the development of dynamic models that predict biological variables such as fish weight. These models integrate observable factors, such as biomass and feed inputs, with external variables, such as temperature and Dissolved Oxygen (DO) [128, 129, 130]. Nutrient-based growth models relate development to nutrient intake [131]. The Dynamic Energy Budget (DEB) theory [132] provides a physiological framework for allocating energy to maintenance, growth, and reproduction. DEB-based bioenergetic models are widely used to simulate growth under changing environmental conditions. Similarly, energy-protein (EP) flux models quantify growth based on metabolic energy use and protein deposition [131]. Classical approaches, such as the von Bertalanffy Growth Function (VBGF) [133], are also used to describe long-term growth patterns. Although mechanistic models offer strong biological interpretability, they often require complex formulations and many parameters that are difficult to identify in commercial aquaculture. Environmental variability, including differences in equipment, water systems, and feeding practices, can cause significant deviations between model predictions and actual growth outcomes. These challenges limit the practical use of purely mechanistic frameworks and highlight the need for adaptive or data-driven alternatives [131].

5.2.2 Data-Driven Methods

Recent advances in Artificial Intelligence (AI) and ML have significantly expanded the capabilities of PFF systems. Examples include predictive growth modeling, feeding optimization, water quality analytics, disease detection, and genetic evaluation [134, 135]. Applications range from DL for DO forecasting [136] to ML-based water quality prediction [137, 138] and species classification using Random Forest and k-NN [139]. Vision-based DL models aid biomass estimation, while Internet of Things (IoT) frameworks use NNs to provide real-time water-quality predictions [140]. Temporal convolutional networks perform well in long-horizon predictions [141]. For risk management, Gaussian models classify environments by mortality risk and water quality [142]. The BigFish project applies ML and large-scale data for oxygen prediction, sea lice modeling, and operational decision support [116]. ML-driven feeding optimization systems using sensor and video data have also been developed [143]. Building on these advances, Li et al. [131] present a data-driven growth prediction method that extracts feeding rules from historical records and combines them with a regression-based weight model. However, despite the notable progress outlined above, data-driven methods have several limitations. They depend on high-quality data, which is often limited on real farms. Sensor and vision-based systems can be expensive and sensitive to noise, and models trained at specific sites may not generalize well. In addition, these methods often lack biological interpretability and can produce predictions that are inconsistent with known physiology, especially when feeding records are treated as black-box inputs.

5.2.3 Hybrid Data-driven, Model-driven Approaches

Hybrid approaches aim to combine the interpretability of mechanistic models with the adaptability of data-driven methods. Burke et al. [144] present a hybrid framework that integrates a physics-based fish behavior model with DL to monitor Atlantic salmon health. In this system, mechanistic simulations describe fish movement and interactions, while NNs estimate behavioral parameters using echo-sounder and video data, enabling early detection of health anomalies.

These methods align with SciML, which integrates physical or biological models into ML architectures to enhance data efficiency and scientific consistency [145]. However, these architectures often suffer from instability during optimization and high implementation complexity due to competing objectives in the

loss function [122]. This rigid coupling limits flexibility, making it difficult to adapt to diverse datasets or real-world operational requirements.

This chapter introduces a lightweight hybrid framework that uses mechanistic models to enrich the input feature space, rather than embedding them within the learning architecture. By combining physiologically informed model outputs with time series data from farm monitoring systems, the method improves weight prediction accuracy while maintaining interpretability and operational simplicity. This approach supports robust performance with limited data and provides a practical solution for deploying hybrid intelligence in commercial aquaculture.

5.3 Problem Formalization and Proposed Method

As mentioned in Section 5.1, in our proposed DT framework, PFF serves as a case study for demonstrating how modeling challenges can be addressed through a hybrid data-driven and model-driven approach. This approach integrates bioenergetic features derived from a mechanistic model with standard environmental time series data, enabling a more accurate and dynamic representation of the aquaculture system. This hybrid framework uses the mechanistic model in a preliminary step to enrich the input data. This preprocessing step provides the NN with a more physically meaningful and data-rich input space, leveraging scientific understanding to help the model generalize effectively and maintain compatibility with real biological processes, thereby enabling it to predict rainbow trout fish weight over time.

5.3.1 Mechanistic Component: The Bioenergetic Model

The mechanistic model of rainbow trout used in this study is a bioenergetic model that describes the energy budget underlying fish growth. As detailed by Bolzonella et al. [130], it characterizes biological processes through the following concepts:

- Energy intake A from feed depends on daily ration, digestibility, and the energy content of carbohydrates, lipids, and proteins.
- Energy losses due to fasting catabolism (C), which relates to respiration rate.
- Energy allocation to somatic tissue growth (ε_T), representing the energy content of fish tissue.

These parameters quantify the energy available for growth and its allocation within the organism, directly influencing weight gain over time in fish farming systems. More precisely, as indicated in [130], individual weights can be simulated using the bioenergetic model expressed as:

$$\frac{dw}{dt} = \frac{A - C}{\varepsilon_T} \quad (5.1)$$

where w represents the fish weight, A is the rate of assimilated energy (net anabolism), C is the fasting catabolism, which quantifies metabolic energy losses of a fasting individual, and ε_T is the energy content of somatic tissue.

In discrete form, the dynamic relationship can be expressed as:

$$w_{t+\Delta_t} = w_t + \Delta_t \frac{A_t - C_t}{\varepsilon_{T,t}} \quad (5.2)$$

This discrete formulation clarifies how the growth at next time step (i.e., $t + \Delta_t$) depends on the balance between energy intake and metabolic expenditure, together with the most estimate of fish weight at the most recent time step t . The quantities A_t , C_t , and $\varepsilon_{T,t}$ were computed following the bioenergetic

relationships reported in [130], using environmental measurements (e.g., water temperature, feed ration, and dissolved oxygen). These dynamic variables were then used as input features for the ML model predicting fish weight. This enabled more accurate forecasting of growth trajectories, thereby supporting better-informed management decisions for optimizing aquaculture operations.

5.3.2 Feature Engineering and Data Pre-processing

In this work, we propose a hybrid data-driven and model-driven approach that integrates bioenergetic features from a mechanistic model with standard environmental data. To demonstrate this, we address the problem of predicting the weight of rainbow trout over time. Table 5.1 lists all features used, grouped into three main categories: *management features*, feed ration is treated as a management (control) variable, representing human intervention in the system rather than an environmental or physiological factor; *environmental features*, describing the surrounding water conditions; and *bioenergetic features*, computed through the mechanistic model from the collected data. By incorporating all three feature sets, the hybrid approach aims to enhance predictive performance, particularly in cases with limited sample sizes, compared to purely data-driven models that use only the first two feature sets.

Table 5.1: Feature categories with descriptions and units

Management Features		
<i>FeedRation</i>	Daily amount of feed provided	g
Environmental Features		
<i>Cond</i>	Electrical conductivity	$\mu\text{S}/\text{cm}$
<i>NH₄⁺</i>	Ammonium ion	mg/L
<i>NO₃-N</i>	Nitrate nitrogen	mg/L
<i>DO</i>	Dissolved oxygen	mg/L
<i>pH</i>	Acidity/alkalinity	-
<i>Sal</i>	Salinity	PSU
<i>Temp</i>	Water temperature	$^{\circ}\text{C}$
Bioenergetic Features		
<i>A</i>	Net anabolism	kJ/day
<i>C</i>	Fasting catabolism	kJ/day
<i>ε_T</i>	Energy content of somatic tissue	kJ/g

With reference to the environmental features in Table 5.1, note that salinity is typically not considered in freshwater environments, as levels are kept very low and stable. Indeed, in freshwater PFF systems, salinity control involves monitoring and maintaining salt concentrations compatible with rainbow trout welfare and growth, and in our use case, the measured values range from 0.03 to 0.12 PSU. However, even in nominally freshwater conditions, salinity can be included as a control variable because abnormal fluctuations may reflect changes in water quality and affect trout physiology [146], thereby influencing growth and final body weight, even if to a lesser extent than other parameters such as temperature, dissolved oxygen, feed, and pH. As will emerge from the experimental section, its role is better modulated by our hybrid data-driven, model-driven approach than by the classical data-driven approach, where its importance is assumed equal to that of the other features.

In general, the proposed hybrid approach leverages scientific understanding to enrich the feature

set, enabling the model to generalize more effectively and produce predictions that better align with real biological processes. In other words, by making some complex yet known relations explicit, the model can leverage additional data to refine its general knowledge, thereby requiring fewer input samples to increase overall accuracy.

In the following subsections, we describe the ML architectures and the learning paradigms examined in this paper. We first outline the architectures used to model the system dynamics, and then discuss the learning paradigms that complement these architectures, such as incremental learning and explainability, which enhance the adaptability and interpretability of the overall framework in dynamic operational settings.

5.3.3 Machine Learning Architectures

As outlined in Section 2.2, LSTM models are effective at capturing long-term dependencies in sequential data, making them suitable for modeling fish growth over time. However, they are less effective at capturing localized patterns within individual observations. In this context, temporal features display clear sequential dependencies, while environmental and bioenergetic features frequently exhibit local short-term patterns, such as fluctuations in temperature, oxygen levels, or energy assimilation rates, which are more effectively captured by convolutional operations. At the same time, CNN models are proficient at extracting localized representations within the feature space, thereby complementing the LSTM's capacity to model temporal dynamics. In Section 3.3.1.2 we already described several hybrid architectures combining CNN and LSTM networks, including sequential CNN-LSTM and LSTM-CNN designs as well as parallel CNN-LSTM structures, for the estimation of temperature and environmental conditions inside a greenhouse. In this chapter, we evaluate all these configurations in Section 5.5.5 within the experimental setup described in this study. Based on their empirical performance, the LSTM-CNN architecture was selected, as positioning the CNN layers after the LSTM block consistently provided superior results for this application. This configuration allows the LSTM to first resolve global temporal growth trends, after which the CNN layers refine the prediction by modeling localized, nonlinear interactions between environmental stressors and the bioenergetic state. Accordingly, the framework is enhanced by replacing the machine learning component in Figure 5.1 with the architecture shown in Figure 5.5.

As shown in Figure 5.5, all input features, including management, environmental, and bioenergetic variables derived from the mechanistic approach, are initially processed by a stack of three bidirectional LSTM layers with 512, 256, and 128 units, respectively, interleaved with dropout layers to mitigate overfitting. These LSTM layers are designed to capture long-term dependencies and sequential patterns, such as temporal trends in environmental conditions and the progression of fish growth. The LSTM stack output is reshaped and passed to a CNN module comprising two 1D convolutional layers (64 and 10 filters) with ReLU activations (Section 2.1.1), followed by a dense layer for final prediction. The CNN module is particularly effective at identifying local and nonlinear patterns in the combined feature representations, likely emphasizing bioenergetic interactions and short-term correlations among environmental variables. This sequential architecture enables the model to leverage both temporal dependencies and local feature relationships.

5.3.4 Advanced Learning Paradigms

The proposed hybrid data-driven and model-driven framework is naturally suited to support advanced learning paradigms that are essential for dynamic PFF operations. In particular, the presented learning pipeline presented in Figure 5.5 can be enriched with two complementary components that enhance its adaptability and practical utility: “incremental learning” and “explainability”. Together, these extensions

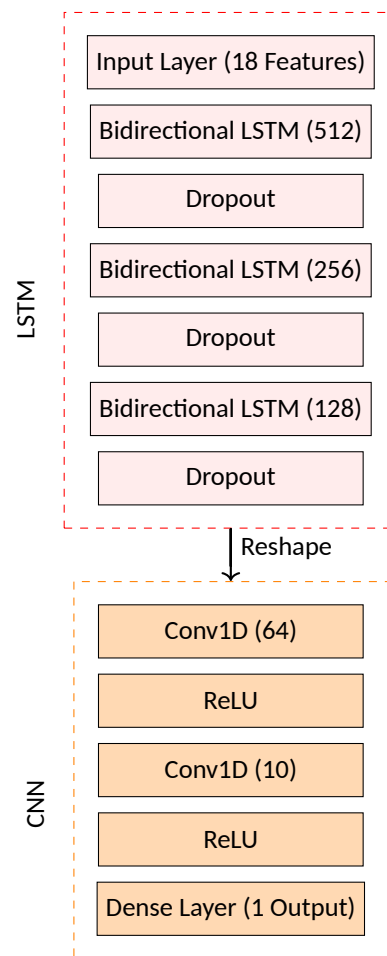


Figure 5.5: LSTM followed by CNN Model Architecture

strengthen the robustness of the hybrid model in real-world aquaculture settings, where environmental conditions evolve continuously, and operational decisions must be justified with transparent evidence.

Incremental learning is an ML paradigm in which a model continuously learns from new data while retaining knowledge from previous data [147]. Traditional batch learning trains models on fixed datasets. In contrast, incremental learning enables models to adapt as new information arrives. This adaptability suits real-world applications where data come sequentially or change over time, such as predictive maintenance, financial forecasting, sensor networks, and DTs.

Explainability provides a complementary layer of interpretability by revealing how different mechanistic and data-driven features contribute to the model's predictions. SHAP [148] is a model-agnostic framework for interpreting ML predictions by assigning each feature an importance value based on its contribution to the output. This method, grounded in cooperative game theory, uses the Shapley value to fairly quantify each feature's impact on the model's predictions and to provide interpretability. In this study, SHAP was employed to evaluate the impact of management, environmental, and bioenergetic features on rainbow trout weight predictions in the proposed hybrid data-driven, model-driven framework. This approach complements standard accuracy metrics and provides insight into the model's internal decision-making process. Visualizations, such as the summary plot, improve the interpretation of results and support transparency and trust in AI-driven aquaculture management.

5.4 Application Setup and Data Description

The data used in this study were collected from a rainbow trout (*Oncorhynchus mykiss*) farm in the municipality of Preore, province of Trento (Northern Italy) ¹. The farm is dedicated to rainbow trout grow-out and represents a well-regulated, real-world case study for evaluating the proposed hybrid approach.

5.4.1 Farm Infrastructure and Management

The facility is composed of seven concrete raceways, each approximately 200 m long, 8 m wide, and 0.8 m deep (Figure 5.6). Water for the entire farm is withdrawn from the Sarca River and, after passing through the raceways, is returned to the same river. The farm is operated by a family-run enterprise associated with the ASTRO Consortium (Associazione Troticoltori Trentini).

ASTRO has established standardized protocols for aquaculture operations, including feeding programs, water-quality guidelines, and quality requirements for the final product. Compliance with these protocols is certified by the Protected Geographical Indication (PGI) label, ensuring traceability and high-quality production standards.

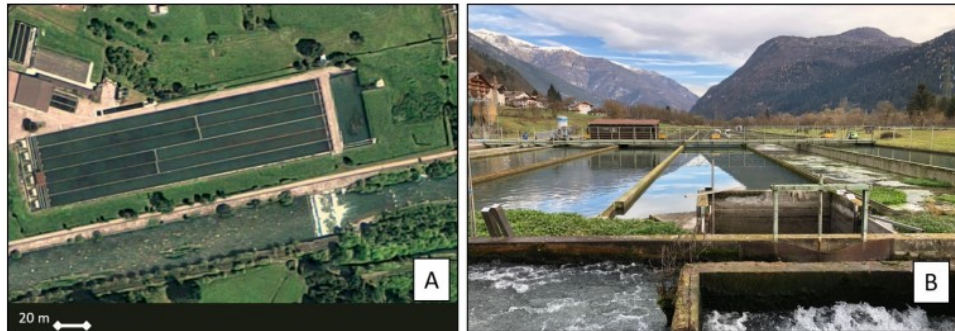


Figure 5.6: Pilot trout farm in Preore (Trento), instrumented during the GAIN project for environmental and biomass monitoring.

5.4.2 Sensor Deployment and Environmental Monitoring

The farm is equipped with sensors for monitoring key water parameters. Specifically, two identical multi-parameter sondes were installed at both the inlet and outlet of a selected raceway, enabling high-frequency measurement of:

- Water Temperature
- Dissolved Oxygen (DO)
- pH
- Electrical Conductivity
- Turbidity
- Ammonium, Nitrate, and Nitrite Concentrations
- Oxidation–Reduction Potential (ORP)
- Chlorophyll-a, Phycocyanin, and Other Relevant Water-Quality Indicators

¹<https://cordis.europa.eu/project/id/773330/results/it>

The complete list of monitored variables and sensor specifications is provided in the GAIN Deliverable D1.1 [149]. These high-frequency measurements capture the dynamic environmental conditions that influence trout metabolism and growth, thereby providing essential input for hybrid mechanistic, data-driven modeling.

5.4.3 Biomass and Operational Data

Fish biomass was monitored using automated systems, such as Biomass Daily (BD) [149], enabling non-invasive measurement of size distributions and growth dynamics over time. As illustrated in Figure 5.7, the BD system uses infrared light and sensors to measure fish dimensions non-invasively whenever a fish swims through a sensing frame. From these measurements, the length and height of each fish are estimated, and then individual weights are calculated using regression models [149]. This provides daily estimates of size distribution and total biomass without manual weighing.

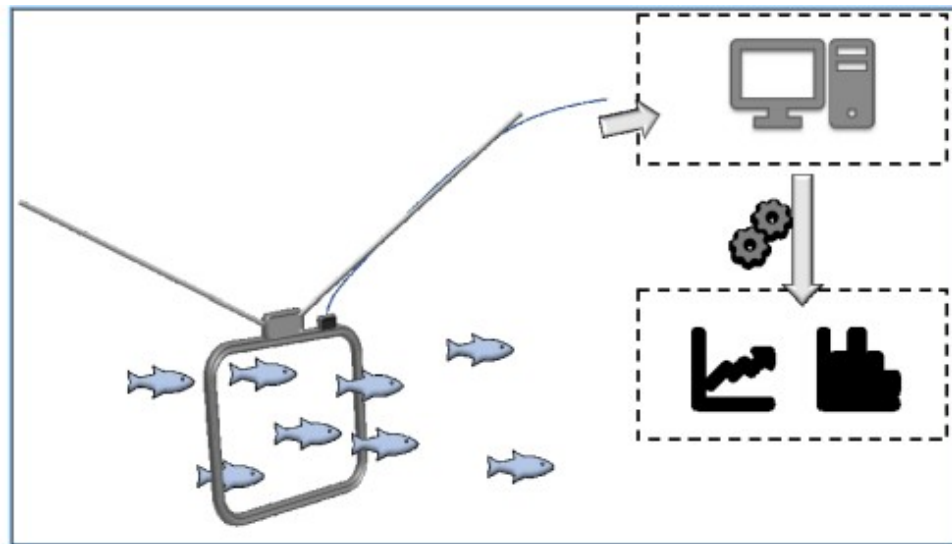


Figure 5.7: Biomass Daily System functioning.

Key metadata for this study were recorded in accordance with ASTRO protocols, including feeding logs, real-time environmental monitoring, and continuous biomass measurements. These data make the farm an ideal pilot site for testing hybrid, mechanistic, data-driven approaches to predict rainbow trout weight.

5.5 Experimental Results

This section presents experiments conducted on the dataset presented in Section 5.4 to compare the accuracy of the proposed hybrid data-driven and model-driven method with that of a pure data-driven and mechanistic approach. Some experiments have also been conducted to assess the incremental learning capabilities of the proposed method, and a further comparison has been conducted with respect to the data augmentation technique.

5.5.1 Experiment Setup

The solution introduced in Section 5.3 has been validated with the data collected at a rainbow trout farm in Preore, Northern Italy (Section 5.4). As mentioned, the BD device, an infrared, non-invasive technology, is used to monitor rainbow trout weight distribution over time. Monitoring was conducted from August

to September 2019 and from July to November 2020. During these periods, all relevant water quality parameters and daily feed amounts were recorded, as previously described. The BD device, together with multi-parametric probes, was used to track changes in fish weight over time.

The dataset comprises three distinct monitoring campaigns. Table 5.2 summarizes the temporal boundaries and the sampling frequency for each campaign. To support reproducibility, the source code and datasets are publicly available on GitHub².

Table 5.2: Metadata and sampling parameters for the three monitoring campaigns.

Campaign ID	Time-Frame (Duration)	Sampling Frequency (obs/day)
1	2019-07-03 – 2019-07-20	24
2	2019-11-15 – 2019-12-17	24
3	2020-07-08 – 2020-07-30	72

Because the data were collected during multiple campaigns with varying sampling frequency, each campaign was treated as a distinct time series. Performance was assessed using the MSE, MAE, and MAPE presented in Section 3.4.2.

Given that the second campaign provides larger sample sizes (33 days), the majority of the experimental analysis focuses on this campaign's data to ensure statistical reliability. Campaign 1 is not included in the comparison reported in this section because its time series is too short to provide a reliable training set; the model would encounter too few temporal patterns to learn meaningful growth dynamics, resulting in unstable or uninformative outcomes. The specific campaign utilized for each experiment is explicitly mentioned in the following experiments.

5.5.2 Preprocessing of Weight Measurements

The estimates produced through the BD device are inherently fluctuating, since they depend on the fraction of the population that falls within the frame and the size of that subsample. For instance, Figure 5.8 shows the hourly fish weight estimates collected by the BD device on Campaign 2. This sampling-induced variability introduces noise into predictions, which serve as ground-truth values during the training phase of ML models, making it essential to apply appropriate processing techniques to uncover underlying growth patterns over time. Indeed, to transform raw, high-frequency biomass measurements into a stable training target for the predictive model, a multi-stage preprocessing pipeline was implemented. This pipeline addresses the inherent fluctuation of automated aquaculture monitoring and consists of two main components that prepare the raw measurements for training: (i) Data Cleaning and Daily Aggregation, and (ii) Temporal Smoothing of Daily Weight, as detailed below.

5.5.2.1 Data Cleaning and Daily Aggregation

Despite the frequency of BD estimates, we aggregate the data into daily intervals because fish growth is not physiologically meaningful at the hourly scale. Given that, the weight of an average individual inside the population is obtained as follows. First, outliers are removed by using the 1st and 99th percentiles of the daily data distribution. This step helps eliminate spurious values caused by mis-estimation or unusual fish postures. Second, from the remaining data, we then extract the median daily weight as the representative value. The median is preferred over the mean due to its greater resilience to the skewed, heavy-tailed noise profiles typically found in underwater sensing systems.

²<https://github.com/AshrafSharifi/fishweightPrediction-hybridModel>

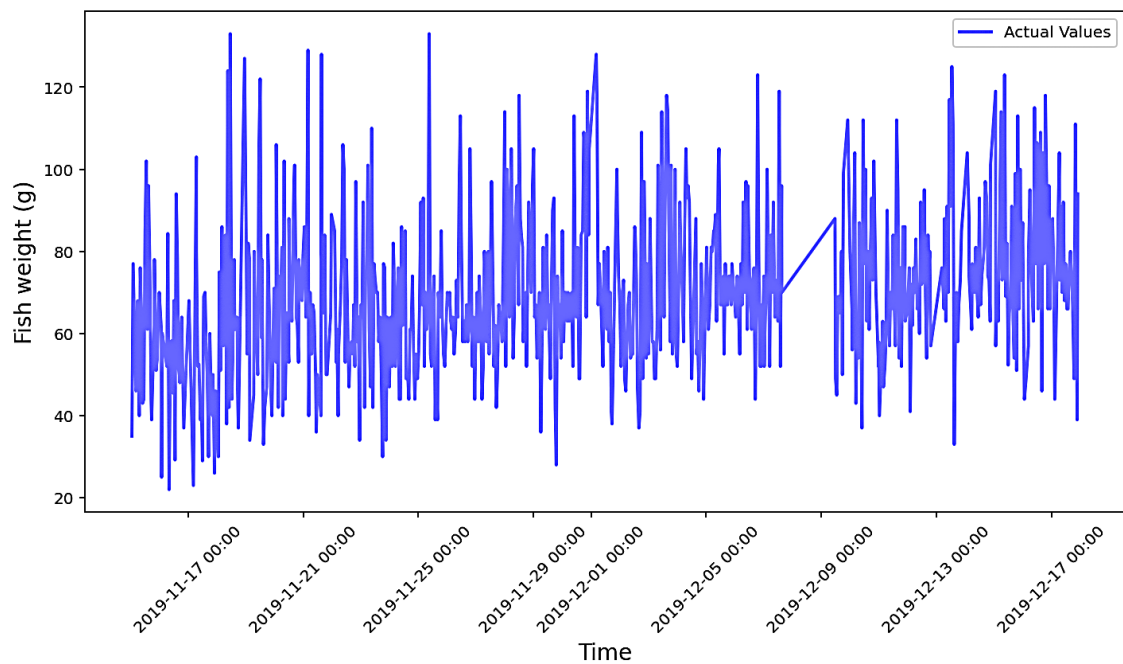


Figure 5.8: Hourly weight measurements for Campaign 2 acquired by the BD system.

5.5.2.2 Temporal Smoothing of Daily Weight

Even after daily aggregation, the weight signal still showed short-term fluctuations that do not reflect actual biological growth. To obtain a physiologically plausible trajectory, we smooth the daily median weights using a “Savitzky–Golay” filter, with a window length of 7 days and a polynomial order of 2. The Savitzky–Golay filter, first introduced by Savitzky and Golay in 1964 [150], is a digital filter applied to data points to smooth the data without distorting underlying trends. This is achieved by fitting a low-degree polynomial to successive subsets of adjacent data points using linear least squares. This method is preferred in biological modeling with respect to spline smoothing because it preserves higher-order moments of the data, ensuring that subtle changes in growth acceleration are not lost during smoothing, while the latter produces a more rigid global curve that may over-smooth localized variations, potentially masking key biological responses or shifts in growth rates [151]. Moreover, unlike a simple rolling average, the Savitzky–Golay filter maintains the overall trend and curvature of the growth curve while effectively removing high-frequency noise.

The application of this technique to the data in Figure 5.8 produces the trend displayed in Figure 5.9. As you can notice, we obtain a smooth daily weight trend that more accurately approximates the population’s true growth trajectory.

5.5.3 Logarithmic Transformation and Target Definition

To make the weight data more manageable for the model, we log-transform the daily smoothed weight values. This transformation reduces the impact of very large values and stabilizes growth patterns over time. Given that, we compute the daily log-growth as the difference in the log-weight between two consecutive days, providing a clear measure of how much the fish has grown from one day to the next.

In this way, the model can focus on relative changes in weight rather than absolute fish size, improving learning and minimizing sensitivity to noise.

Since the values are already log-transformed, this difference naturally reflects proportional growth, making it more consistent and easier for the model to predict than raw weight changes. It isolates daily

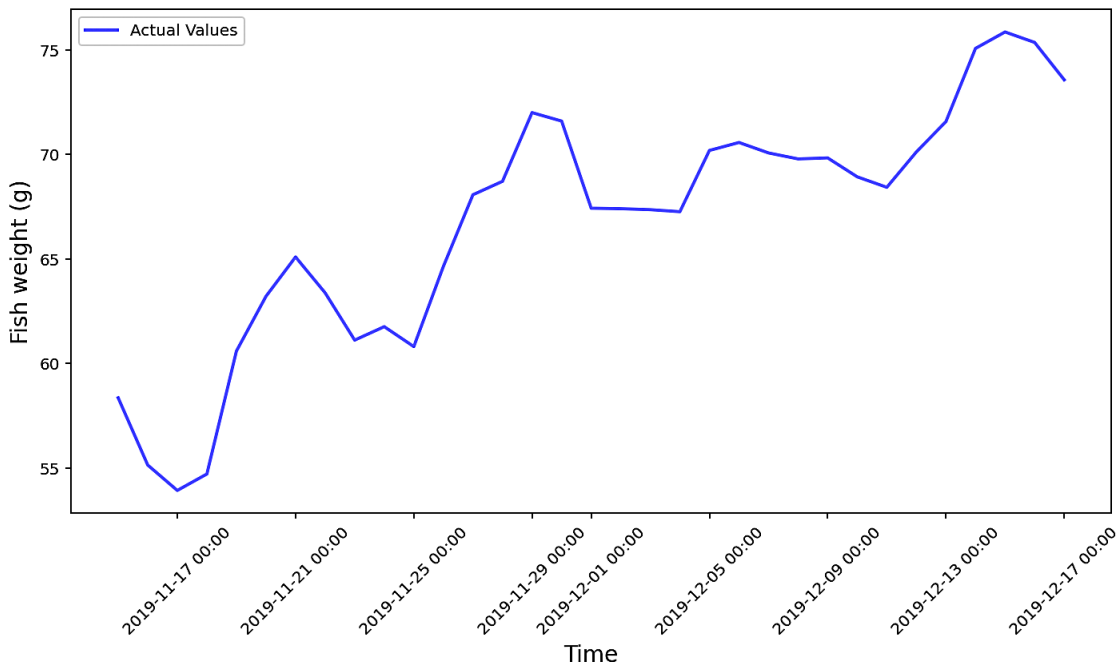


Figure 5.9: Smoothed daily weight estimates for Campaign 2 after noise reduction and aggregation.

pure values, allowing the model to directly link growth to factors such as temperature or oxygen levels. This transformation could stabilize the benefits of the approach for both small and large harvest-ready fish, as it emphasizes growth rate rather than absolute weight, which is represented by smaller values in the former cases and larger values in the latter. Indeed, without this transformation, prediction errors made in the latter are considered worse than those obtained in the former.

The preprocessing steps described above allow us to obtain, from the BD estimates, a clean, biologically meaningful target ground truth signal for the model considered. Indeed, the target, daily log-growth, has a smooth, scale-independent truth value and is directly related to environmental factors.

5.5.4 Implementation Details and Hyperparameters

As described, the proposed predictive model uses a many-to-one configuration, where a fixed-length sequence of 72 hours, representing three days of environmental, biological, and feeding history, is mapped to a single daily growth target. This 72-hour lookback window was selected to capture short-term metabolic responses while maintaining a consistent input dimensionality. To optimize the network, an Adam optimizer (see Section 2.1.4) was employed with a learning rate of 0.001 and a batch size of 16. To ensure the model generalizes well to unseen data, a dropout rate of 0.2 was applied to the hidden layers as a regularization measure, and training used early stopping to prevent overfitting to high-frequency noise.

Because aquaculture data is inherently chronological, traditional random cross-validation was replaced by the “TimeSeriesSplit” strategy. More precisely, to evaluate the model’s performance, a five-fold progressive TimeSeriesSplit strategy was implemented. In this context, a fold refers to one chronological split of the time series into a training segment followed by a validation segment. Each fold, therefore, represents an independent forecasting scenario in which the model is trained on past observations and evaluated on future, unseen data. This setup preserves temporal causality and prevents information leakage, ensuring that performance metrics reflect realistic deployment conditions in which predictions are always made using only historical information. In the first fold, the model trains on the earliest portion of the dataset

and is evaluated on the immediately following segment; in each subsequent fold, the training window expands forward in time to include the previous validation set, while a new block of data is reserved for testing. This folding procedure ensures that the model is consistently tested on unseen data that follows the training period chronologically. The final performance metrics reported for the model represent the average evaluation across all five folds, providing a robust and stable estimate of its forecasting ability. This rigorous separation of training and test data ensures that the model's performance accurately reflects its ability to generalize to future conditions in a DT forecasting system.

5.5.5 Model Architecture Comparison

To identify the most suitable model architecture, we evaluated several recurrent, convolutional configurations on Campaign 2. The tested architectures include LSTM, CNN-LSTM, LSTM-CNN, and Parallel LSTM,CNN, each combining temporal and spatial feature extraction in different orders, with configuration details provided in Section 3.3.1.2. All models were trained using the full set of input features listed in Table 5.1, and the same experimental setup was used to ensure a fair comparison. The goal of this experiment is to determine which configuration provides the most stable and accurate predictions and, therefore, should be adopted in subsequent experiments. Table 5.3 reports the result of this experiment.

Table 5.3: Comparison of different recurrent-convolutional architectures for fish weight prediction on Campaign 2, using the full feature set. Results are reported as mean $\pm$ standard deviation.

Model	MSE (mean $\pm$ std)	MAE (mean $\pm$ std)	MAPE (% , mean $\pm$ std)
LSTM	14.96 $\pm$ 13.31	3.04 $\pm$ 1.67	4.56 $\pm$ 2.77
CNN-LSTM	9.91 $\pm$ 5.03	2.59 $\pm$ 0.59	3.79 $\pm$ 0.73
LSTM-CNN	5.82 $\pm$ 3.09	1.96 $\pm$ 0.62	2.93 $\pm$ 1.00
Parallel LSTM,CNN	67.75 $\pm$ 62.81	5.99 $\pm$ 3.58	9.03 $\pm$ 5.82

The results show clear performance differences across the tested architectures. The LSTM-CNN model achieves the lowest errors on all metrics, with an MSE of 5.82, MAE of 1.96, and MAPE of 2.93%, outperforming all other variants. This indicates that extracting spatial features after temporal modeling provides a more informative representation of the fish weight dynamics. Indeed, this configuration allows the LSTM to first resolve global temporal growth trends, after which the CNN layers refine the prediction by modeling localized, nonlinear interactions between environmental stressors and the bioenergetic state. In contrast, the Parallel LSTM, CNN model performs substantially worse, with high variance and significantly larger errors, suggesting that the additional CNN parameterization in this configuration introduces instability rather than improving predictive power.

Overall, the LSTM-CNN architecture demonstrates the best balance of accuracy and robustness, making it the most suitable choice for both the pure data-driven baseline and the proposed hybrid approach in the subsequent experiments. Based on these results, all subsequent experiments in this section will utilize the LSTM-CNN architecture for both the pure data-driven baseline and the proposed hybrid model. By fixing the architecture in this manner, we ensure that the following comparisons focus on the impact of integrating the model-driven features inside the training data, rather than differences in network design.

5.5.6 Evaluation of the Hybrid Framework and Baseline Comparisons

To evaluate the forecasting performance of the proposed method, we employed two complementary experimental setups. The first setup assesses the model's generalization ability through controlled cross-

validation, while the second evaluates its performance in predicting the fish's complete weight trajectory. Both experiments utilize the two feature configurations described in Section 5.3.2:

- The full feature set, as listed in Table 5.1, which includes both mechanistic and data-driven variables and therefore corresponds to the “hybrid modeling” configuration.
- A reduced feature set including only management and environmental features from Table 5.1, representing the “purely data-driven” baseline model.

Additionally, the model's predictions are compared against a purely mechanistic model-driven approach based on the bioenergetic framework proposed in [130].

5.5.6.1 Cross-validated Comparison of Feature Sets

The first experiment evaluates the two feature configurations, full and reduced, which lead to the pure-data-driven and the proposed hybrid alternatives, respectively, using a time series cross-validation scheme. For each feature configuration, the dataset is divided into chronological folds, ensuring that validation data always occurs after the training data. This prevents leakage and provides a realistic estimate of forecasting performance.

For each fold, the model is trained on the earlier portion of the data and validated on the subsequent segment. Metrics from all folds are then averaged to summarize performance. Averaging is meaningful in this context because each fold represents an independent forecasting task with its own training and validation split, allowing the model's generalization behavior to be assessed across multiple temporal conditions.

Table 5.4: Comparison of performance metrics for the pure data-driven baseline and the proposed hybrid data-driven, model-driven approach across Campaigns 2 and 3.

Method	MSE	MAE	MAPE(%)
Campaign 2			
Pure data-driven	32.79	3.76	5.80
Proposed hybrid approach	5.82	1.96	2.93
Campaign 3			
Pure data-driven	371.96	13.63	7.23
Proposed hybrid approach	182.58	11.77	5.16

The results in Table 5.4 show that the proposed hybrid approach consistently outperforms the pure data-driven baseline in both Campaign 2 and Campaign 3. In Campaign 2, the hybrid model achieves the largest gains, with clear improvements in all metrics. This strong performance is partly due to more stable growth patterns and a longer observation period (33 days, as indicated in Table 5.2), which provides richer temporal information for model fitting.

Campaign 3 is more challenging, leading to higher errors for both methods. This is mainly due to higher biomass variability and a shorter duration of only 22 days, which limits the temporal context for learning growth trajectories. Despite this, the hybrid model still performs better, halving the MSE and reducing the MAPE from 7.23% to 5.16%. This shows that the mechanistic knowledge helps stabilize predictions even with high variability and limited data.

Overall, the hybrid approach is more robust, performing best when data are plentiful (Campaign 2) and still maintaining clear advantages when variance is high and the training period is shorter (Campaign 3).

5.5.6.2 Full-sequence Prediction and Comparison with Mechanistic model

The second experiment evaluates the model in a deployment-like scenario. After training, the model is used to predict the fish's entire daily weight trajectory.

Unlike cross-validation, full-sequence prediction produces a single continuous trajectory rather than multiple folds. For this reason, averaging metrics across folds is not applicable. Instead, the predicted trajectory is directly compared with the observed daily weights, and a single set of error metrics is reported for each method.

This experiment highlights differences in long-term behavior, such as curvature, drift, or systematic bias, which are not visible in fold-level evaluations.

Table 5.5: Comparison of performance metrics for the mechanistic model, pure data-driven baseline, and the proposed hybrid approach across Campaigns 2 and 3.

Method	MSE	MAE	MAPE (%)
Campaign 2			
Mechanistic model	143.44	11.46	17.27
Pure data-driven	29.36	4.79	7.06
Proposed hybrid approach	11.40	2.67	4.01
Campaign 3			
Mechanistic model	9,777.72	98.22	45.40
Pure data-driven	492.12	16.53	7.51
Proposed hybrid approach	120.57	9.78	4.57

The full-sequence evaluation in Table 5.5 shows clear differences in long-term predictive behavior across the three modeling strategies. In Campaign 2, the hybrid approach achieves the lowest errors across all metrics, substantially outperforming both the mechanistic model and the pure data-driven baseline. This indicates that combining mechanistic knowledge with a data-driven approach helps maintain trajectory stability throughout the full growth period.

Campaign 3 is considerably more challenging, as reflected in the large errors of the mechanistic model and the greater difficulty for the data-driven baseline. The hybrid model again provides the most accurate trajectory, reducing MSE from 492.12 to 120.57 and lowering MAPE from 7.51% to 4.57%. These improvements suggest that the hybrid formulation is more resilient to the higher variability and noisier growth patterns characteristic of Campaign 3.

Overall, the full-sequence experiment confirms that the hybrid approach not only improves pointwise accuracy but also better preserves long-term trajectory prediction, which becomes visible only when predicting the entire growth curve at once.

To better illustrate the performance of the proposed hybrid method, this section presents various visualization techniques, including line plots, scatter plots, and residual distributions. about the complete sequence prediction of the second Campaign 2 of the dataset. The sequence-prediction results show that the proposed hybrid model performs effectively as a robust weight estimator. Instead of showing the fluctuations in average measurements produced by the DB system (blue line in Figure 5.10), the model produces a stable, more biologically realistic growth curve. Large drops in the sampled weights, such as those seen around Day 1 and Day 15, are correctly ignored as measurement noise or sampling inconsistencies. The predicted trajectory follows a smooth upward trend and captures periods of faster and slower growth, reflecting real physiological responses rather than assuming a constant growth rate.

The residual plot (Figure 5.12) supports this result: most errors fall within a narrow range, and their random distribution around zero indicates that the model is unbiased. The scatter plot (Figure 5.11) further shows that predictions, especially at higher weights, align closely with the true values, demonstrating improved accuracy as more historical context becomes available. Overall, the model successfully extracts the underlying biological signal from noisy observations and provides a clean, reliable growth estimate that is more useful for operational decision-making than raw manual measurements.

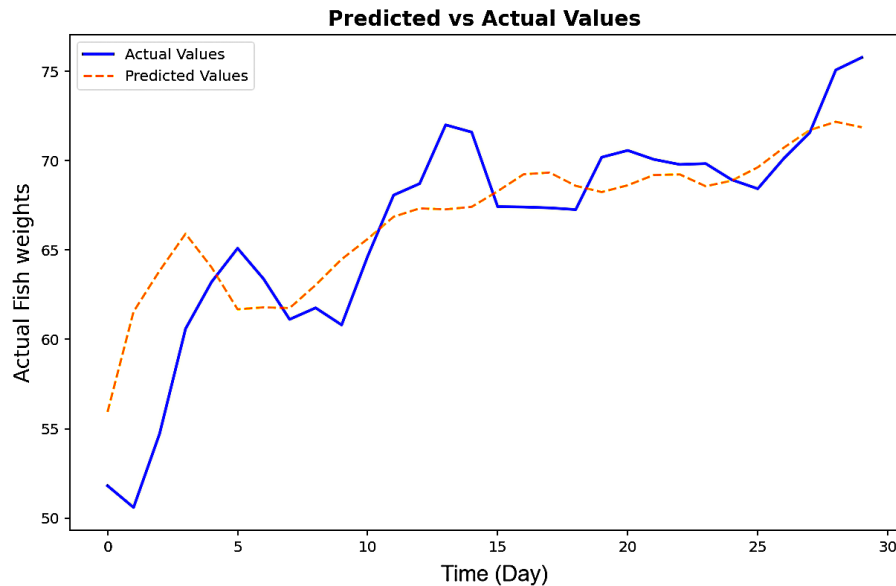


Figure 5.10: Line plot comparing the average weight predicted by the proposed hybrid model and the ground truth values produced by the BD system.

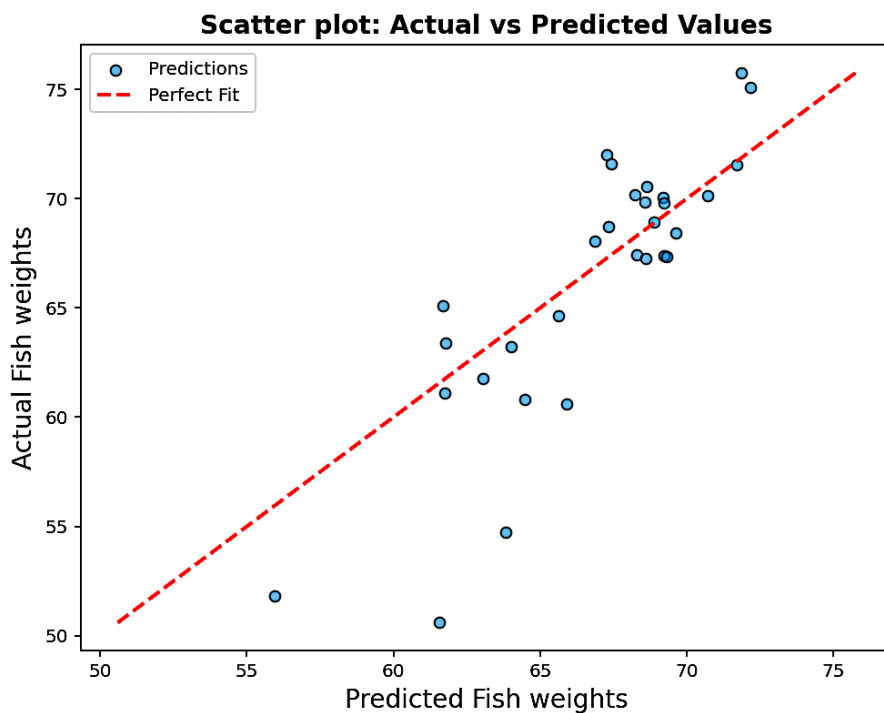


Figure 5.11: Scatter plot comparing the average weight predicted by the proposed hybrid model and the ground truth values produced by the BD system.

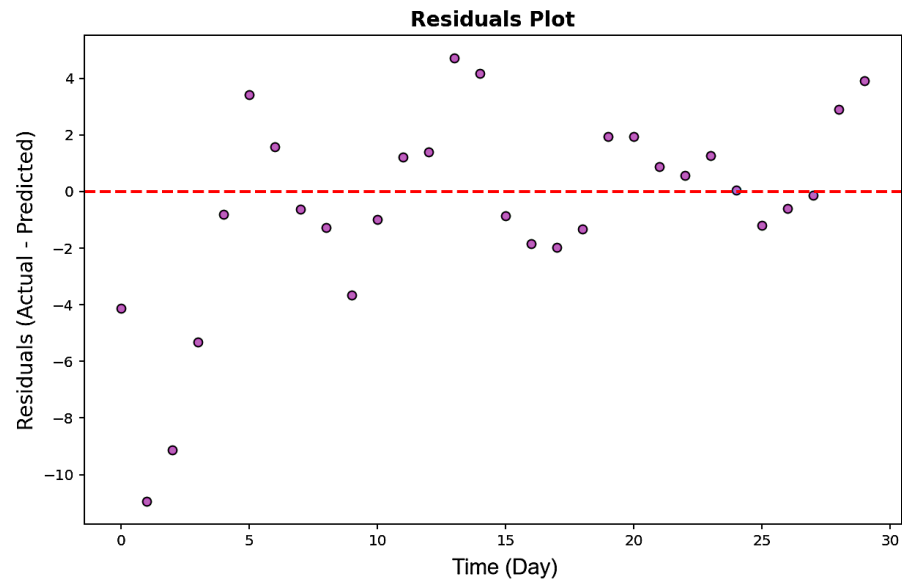


Figure 5.12: Residual plot illustrating the difference between the BD measures and the fish weight estimated by the hybrid model along the full sequence prediction.

5.5.7 Evaluation of Incremental Learning and Data Efficiency

As discussed in Section 5.3.4, incremental learning enables a model to improve continuously as new data becomes available. In this section, we investigate the proposed hybrid approach’s ability to improve accuracy over time relative to the pure data-driven baseline. Using data from Campaign 2, we divided the timeline into five sequential steps, with each part used to incrementally update and test the models. In this section, incremental learning was implemented using an expanding-window strategy. In step 1, the model was trained on the earliest block of data and validated on the next unseen block. In step 2, the training set was expanded to include both the first and second blocks, and validation was performed on the third block. This process continued until all blocks were used. At each step, the weights obtained from the previous step were reused and further updated, allowing the model to accumulate knowledge over time. This design mimics real deployment conditions and enables quantification of how additional historical data improves forecasting performance. Table 5.6 presents the performance of the proposed hybrid method across these steps.

Table 5.6: Performance comparison between the proposed hybrid method and the pure data-driven approach across training steps.

Method	Step	Train size	MSE	MAE	MAPE (%)
Proposed hybrid	Step 1	7	48.51	5.30	7.7
Proposed hybrid	Step 2	14	21.39	4.25	6.2
Proposed hybrid	Step 3	21	3.04	1.46	2.1
Pure data-driven	Step 1	7	58.16	5.92	8.7
Pure data-driven	Step 2	14	27.42	4.68	6.9
Pure data-driven	Step 3	21	44.59	5.90	8.4

The results in Table 5.6 clearly illustrate how incremental learning affects the predictive performance of both the proposed hybrid model and the pure data-driven baseline. In Step 1, where only 7 days of training data are available and both models are initialized from scratch, performance is limited for both approaches. The hybrid model achieves slightly lower errors than the data-driven baseline (MSE = 48.51 vs.

58.16), but both methods struggle due to the short training window and the lack of accumulated temporal context.

As the training window expands, the hybrid model shows a great and consistent improvement. In Step 2, doubling the training size reduces the MSE of the hybrid model by more than half (from 48.51 to 21.39), while the data-driven model improves by a more modest amount (from 58.16 to 27.42). This indicates that the hybrid approach is more effective at integrating additional historical information and refining its internal representation of growth dynamics.

By Step 3, the difference between the two approaches becomes substantial: with 21 days of training data, the hybrid model achieves a very low error (MSE = 3.04, MAE = 1.46), while the data-driven model deteriorates relative to Step 2, with MSE increasing to 44.59. This rebound suggests that the pure data-driven model becomes unstable as the temporal window grows, likely overfitting local fluctuations rather than capturing the underlying growth trajectory. In contrast, the hybrid model continues to improve, benefiting from the mechanistic knowledge that constrains learning and stabilizes updates across steps.

Overall, these results demonstrate that the hybrid approach is significantly more data-efficient and robust in an incremental learning setting. It consistently improves as more observations become available, while the pure data-driven model shows signs of instability and reduced generalization. This behavior aligns well with real deployment scenarios, where models must update continuously as new data arrives while maintaining reliable long-term predictive performance.

5.5.8 Performance and Training Data Requirements of the Proposed Hybrid Approach

As hypothesized in Section 5.1, integrating bioenergetic knowledge is expected to reduce the amount of historical data required for model convergence and high predictive accuracy.

To investigate how much historical information is required for stable biomass forecasting, an additional experiment was conducted in which the model was trained on progressively larger subsets of the early campaign. Rather than using the full dataset for training, only the first 20%, 40%, 60%, and 80% of the available 72-hour windows were used to fit the model. Importantly, all models were evaluated on the same fixed hold-out period at the end of the campaign.

This experiment is fundamentally different from the incremental-learning setup described in the previous section. In incremental learning, the model is updated sequentially as new data becomes available, allowing it to retain knowledge gained from earlier training phases. In contrast, the current experiment does not use incremental updates. Instead, for each specified training-history percentage (20%, 40%, 60%, 80%), a new model is trained from scratch with identical hyperparameters using only the corresponding portion of the early campaign data. This design isolates the effects of the amount of historical data available during training, ensuring that any performance differences come solely from the length of the training window, rather than from accumulated updates or sequential adaptation.

Table 5.7: Performance with different training set proportions.

Train %	Train size	Eval size	MSE	MAE	MAPE (%)
20%	6	9	62.40	6.33	8.70
40%	12	9	25.77	4.69	6.62
50%	15	9	30.31	5.10	7.18
60%	18	9	10.63	2.84	4.05
80%	24	9	5.73	1.67	2.29

The results in Table 5.7 show a clear trend: forecasting accuracy improves steadily as more historical

data are included in the training set. With only 20% of the campaign, the model lacks sufficient temporal context and performs poorly. However, accuracy increases rapidly once 40–60% of the data are available, indicating that the model begins to capture the underlying growth dynamics with a relatively modest amount of history. Notably, training on just 50% of the campaign already yields performance comparable to the pure data-driven baseline trained on the full dataset (Table 5.4). More precisely, when trained on only 50% of the available campaign data, the hybrid model achieves an MSE of 30.31 and a MAPE of 7.18% (Table 5.7). These values are already comparable to, and in some cases better than, the pure data-driven baseline trained on the full dataset (MSE = 32.79, MAPE = 5.80 for Campaign 2; Table 5.4).

When 60–80% of the data are used, the proposed hybrid model outperforms the pure data-driven model trained on 100% of the data by a wide margin. These findings highlight the data efficiency of the hybrid approach: by incorporating bioenergetic knowledge, the model achieves high accuracy even with limited early-campaign data, making it more suitable for real-time deployment where only a small amount of data may be available.

5.5.9 Interpretability and Feature Importance Analysis

Feature importance analysis using SHAP provides robust validation of the proposed hybrid data-driven and model-driven modeling framework by quantifying each feature’s contribution to model predictions. This assessment determines whether the model’s behavior aligns with established biological principles and, ultimately, differs from a purely data-driven approach. In particular, Figure 5.13 presents SHAP summary plots for both the proposed hybrid method and the pure data-driven method, enabling a direct comparison of feature utilization.

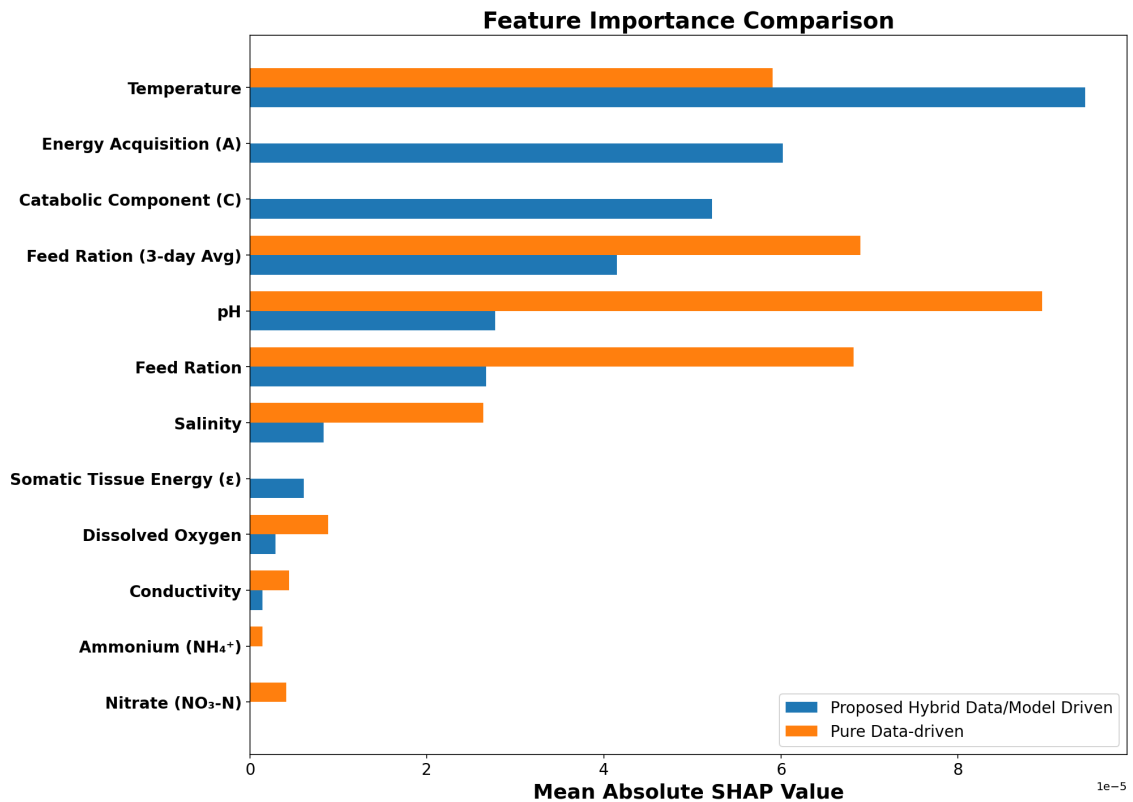


Figure 5.13: SHAP Mean absolute SHAP feature importance comparing the proposed hybrid method with the pure data-driven approach.

The SHAP analysis highlights clear differences in how the hybrid and pure data-driven models use input information to predict daily biomass growth. Temperature emerges as the dominant predictor in both models, consistent with its central role in regulating metabolic activity and growth efficiency in salmonids. However, the hybrid model assigns substantially higher importance to temperature, suggesting that the mechanistic structure amplifies biologically meaningful signals rather than diluting them.

A defining characteristic of the hybrid approach is its effective use of mechanistic bioenergetic variables, Energy Acquisition (A), Catabolic Component (C), and Somatic Tissue Energy (ϵ). These features rank among the most influential in the hybrid model, whereas they are absent in the pure data-driven baseline. Their prominence confirms that the hybrid architecture successfully integrates mechanistic knowledge with environmental observations, enabling the model to anchor its predictions in physiologically interpretable processes. This supports the broader claim that embedding domain knowledge improves both predictive accuracy and interpretability.

In contrast, the pure data-driven model distributes importance more broadly across feed-ration metrics, pH, salinity, and dissolved oxygen. This diffuse pattern suggests that, without a mechanistic structure, the model compensates by relying on a wider set of correlated environmental cues. The hybrid model, by comparison, exhibits a more focused and biologically coherent importance profile, relying heavily on temperature and mechanistic energy-balance components while assigning lower weight to secondary water-quality variables.

Overall, the SHAP comparison demonstrates that the proposed hybrid method not only improves accuracy but also effectively combines mechanistic bioenergetic features with environmental variables, producing a model whose internal reasoning aligns with established biological principles. This reinforces the value of hybrid modeling for aquaculture forecasting, particularly in settings where interpretability and data efficiency are essential.

5.5.10 Framework Generalization and Operational Scalability

One of the key properties of the proposed feature-space enrichment framework is its modular, lightweight architecture, which makes it adaptable to other fish species (e.g., Atlantic salmon, European seabass). Because the mechanistic bioenergetic component is decoupled from the NN optimization landscape, generalizing the framework does not require modifying the underlying DL layers or loss functions. Instead, adaptation relies on updating the bioenergetic module (such as the Dynamic Energy Budget) with species-specific physiological parameters. The LSTM-CNN architecture can then be retrained or incrementally updated on the newly enriched dataset to address the local, non-linear environmental stressors specific to that operational setup.

5.6 Conclusion

This study presents a lightweight, hybrid data-driven and model-driven framework that bridges the gap between mechanistic bioenergetic models and DL in aquaculture. By enriching the input space with calculated bioenergetic features, the approach preserves the optimization flexibility of standard DL while ensuring biological consistency.

Experimental validation on a real-world rainbow trout dataset demonstrates that the proposed hybrid data-driven, model-driven approach significantly outperforms both pure mechanistic and pure data-driven baselines, maintaining superior predictive accuracy with only 50% of the available training data. This confirms that mechanistic knowledge acts as an effective domain-specific regularizer, enabling robust performance even in data-scarce environments.

The incremental-learning experiment further demonstrates the practical value of the hybrid approach in dynamic aquaculture environments. By updating the model sequentially as new observations become available, the hybrid framework adapts more rapidly and with greater stability than the purely data-driven baseline, highlighting its suitability for real-time DT deployments where conditions evolve continuously.

Interpretability analysis using SHAP further demonstrates that the model balances raw environmental drivers with theoretical energy-state features, mitigating error propagation and reinforcing biological plausibility. Together, these results establish a foundation for accurate, interpretable, and data-efficient DTs that support real-time decision-making in precision fish farming.

Chapter 6

A Digital Twin Framework for Modeling *Salicornia* Growth

Chapter contents

6.1	Introduction	91
6.2	Related Work	92
6.2.1	Mechanistic (Model-Driven) Plant Growth Models	93
6.2.2	Data-driven Models for Plant Growth and Biomass Estimation	94
6.2.3	Phenotyping of Specific Plant Organs and Whole Plants	95
6.3	Phase 1: The Perception Validation Layer	95
6.3.1	Technical Framework: Software Libraries for Phenotyping	96
6.3.2	Implementation of the Perception Pipeline	98
6.3.3	Methodological Validation against Phenospex	105
6.4	Phase 2: Hybrid Growth Modeling for <i>Salicornia</i>	107
6.4.1	Pilot Structure of the BeBlue Aquaponic Systems	108
6.4.2	Automated Imaging Infrastructure for Monitoring <i>Salicornia</i> Growth	110
6.4.3	Digital Image Conditioning and Spatial Normalization	113
6.4.4	Morphological Feature Extraction	120
6.4.5	Feature Aggregation and Data Fusion	121
6.4.6	Mechanistic Modeling of Plant Growth Dynamics	125
6.4.7	Integration with Machine Learning	131
6.5	Experimental Results	131
6.5.1	Model Fitting and Numerical Evaluation of Candidate Growth Models	132
6.5.2	Validation of Morphological Proxies via Critical Point Analysis	133
6.5.3	Experimental Setup for Biomass Estimation	133
6.5.4	Feature Ablation Study	135
6.6	Conclusion	142

6.1 Introduction

Plant science has progressively moved from a purely descriptive discipline toward a more quantitative approach, which is crucial for predicting, optimizing, and understanding crop development [152].

The previous chapters (Chapter 3 and Chapter 4) established a framework for monitoring the inanimate microclimate of greenhouses using virtual sensing methods. However, transitioning to modeling biological traits, such as those of fish and plants, presents a fundamentally different challenge. In Chapter 5, we propose a hybrid data-driven and model-driven approach for estimating fish biomass, which is based on the use of historical data measured and captured using devices such as the BD system (Section 5.4.3) installed in tanks and models. However, measuring key biological characteristics in plants, such as total dry biomass, is inherently destructive; determining a plant's internal mass typically requires harvesting, drying, and weighing. This difference highlights a significant gap in DT applications. To ensure continuity and avoid harmful sampling, we need a “virtual biological sensor” that can accurately estimate biomass without damaging the plant.

Plant phenotyping refers to the quantitative characterization of structural, developmental, and physiological plant traits, providing essential insight into how genotype, environment, and management practices interact [153]. As agriculture faces increasing climate variability and resource constraints, precise analysis of plant growth dynamics has become increasingly important for optimizing production and ensuring sustainability. Historically, assessing plant growth has relied on manual, destructive techniques. While these methods are accurate, they are labor-intensive and time-consuming, making them unsuitable for continuous monitoring or large-scale cultivation [154, 155]. Additionally, their destructive nature increases the risk of errors in monitoring the same plant over time, reducing the ability to capture growth trajectories and dynamic responses. To overcome these limitations, modern plant phenotyping has shifted toward non-destructive, high-throughput sensing technologies. Automated imaging and environmental monitoring systems enable continuous observation of plant structure and physiology, supporting early detection of stress signals, such as nutrient deficiencies or microclimate disturbances, and facilitating timely management interventions [156, 157]. These advances also strengthen the link between genotype and phenotype by providing richer, more frequent measurements across diverse environments [153].

Within this context, this chapter introduces an integrated framework for automated *Salicornia* biomass estimation that moves beyond traditional destructive methods. The approach combines classical plant growth analysis with modern data-driven techniques, leveraging image-derived structural features and environmental sensor data to build a biologically grounded, computationally robust estimation pipeline. After reviewing the relevant literature on plant growth analysis, phenotyping technologies, and modeling approaches, the rest of the chapter is organized as follows. As illustrated in Figure 6.1, the chapter is structured into two primary phases:

- Phase 1 (The Perception Validation Pipeline) describes a 2D image-processing pipeline developed using general plant images. To ensure the reliability of the extracted features, this layer is benchmarked against a high-end commercial Phenospex 3D scanning device, demonstrating that low-cost 2D visual traits are accurate proxies for high-fidelity measurements.
- Phase II (The Predictive Modeling Layer) applies this validated pipeline specifically to *Salicornia* development. By merging validated morphological traits with environmental telemetry and mechanistic models, we develop a hybrid framework capable of analyzing *Salicornia* growth and non-destructively estimating biomass.

The remainder of this chapter follows a step-by-step structure that connects computer vision processing

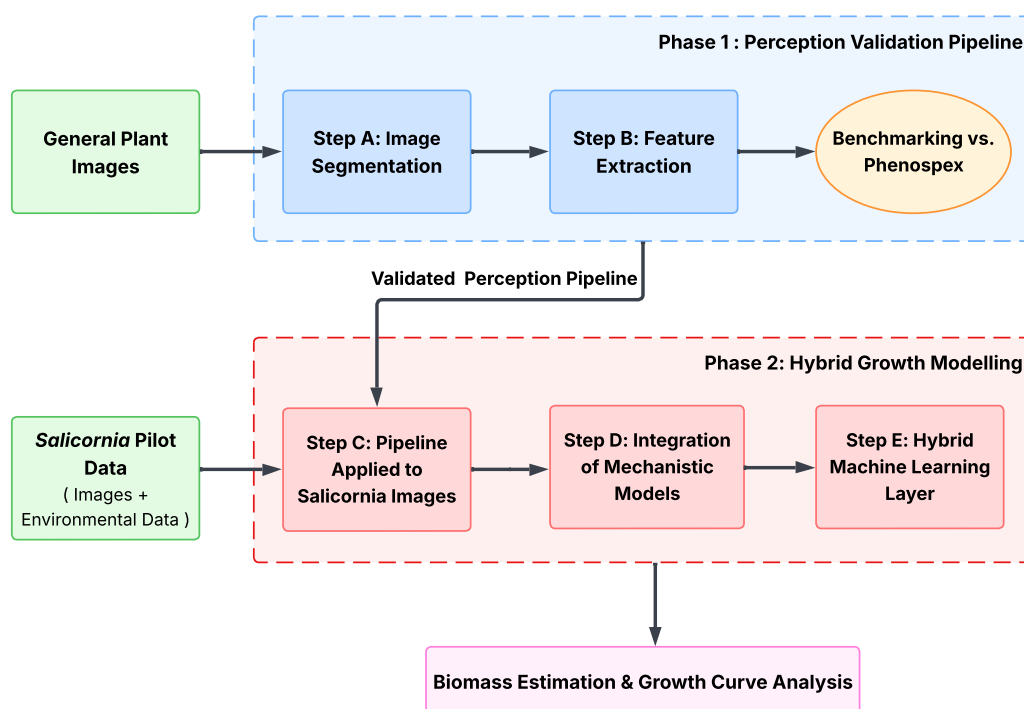


Figure 6.1: Overview of Chapter 6 Structure, Showing the Dual-Phase Pipeline for Perception Validation and Growth Modeling.

with crop-growth modeling. It begins in Section 6.2 with a review of ecophysiological prediction methods, covering classical growth models, functional-structural plant modeling, image-based pipelines, and organ-level phenotyping. Section 6.3 then introduces Phase 1 (see Figure 6.1), the Perception Validation Layer, where 2D morphological traits are extracted using PlantCV and Pixellib and validated against Phenospex 3D laser-scanning measurements. The chapter then moves to our main application in Section 6.4, describing Phase 2 (Figure 6.1), the Hybrid Growth Modeling framework for *Salicornia*. This section outlines the imaging setup, the derivation of growth-rate descriptors using sigmoidal models, and the gradient-boosting feature-enrichment pipeline. Finally, Section 6.5 presents the experimental results, including model-fitting performance, validation of morphological proxies, and an ablation study on biomass estimation, before concluding in Section 6.6.

6.2 Related Work

Plant phenotyping provides valuable quantitative insights into plant traits and is essential for understanding the interactions among genotype, environment, and management practices. Traditional methods for assessing growth, which often involve manual, destructive techniques, can be labor-intensive and limit continuous monitoring. In contrast, modern high-throughput, non-destructive technologies enable automated tracking of plant development and early stress responses.

The benefits of these advancements are further enhanced by the integration of imaging technologies, computational modeling, and machine learning. Together, these tools enable precise, scalable, and automated analysis of plant behavior [158]. This evolution in phenotyping shifts the focus from simple trait measurement to predictive, high-resolution monitoring and modeling of plant growth and performance. Recent research increasingly emphasizes modeling approaches that aim to capture and predict plant growth

dynamics with greater accuracy and biological relevance.

In this context, the following section reviews key approaches to plant phenotyping, including mechanistic models and data-driven methods.

6.2.1 Mechanistic (Model-Driven) Plant Growth Models

Mechanistic model-driven plant growth models describe how biomass accumulates and how plants develop by explicitly representing the biological and physical processes that govern plant behavior. These models are based on ecophysiological principles, including photosynthesis, respiration, carbon allocation, and phenology. They simulate how plants respond to environmental factors, including light, temperature, water, and nutrient availability. Since the parameters in mechanistic models have biological significance, they provide interpretability and enable predictions beyond the observed data.

Within this framework, a set of plant growth models uses light efficiency and photosynthesis to model plant growth. These models measure how much light the plant canopy captures and how effectively it converts that light into biomass. Biomass production depends on the light captured or on photosynthesis, while plant development depends on temperature over time or on the length of daylight. The products of photosynthesis are transported to different parts of the plant according to specific rules. These rules change as the plant grows, showing how the plant's needs (sources) and uses (sinks) shift over time. Models also account for environmental factors such as water shortages, salt levels, and nutrient deficiencies. They include responses that lower photosynthesis or increase energy loss through respiration. Simulations run daily, enabling the prediction of plant growth and final yield over time under different conditions. Representative implementations of this modeling approach include SUCROS, a foundational model for potential crop growth [159]; WOFOST, which provides detailed phenology and stress response formulations and is widely used in European crop monitoring [160]; APSIM, a modular systems model integrating soil, crop, and management components [161]; DSSAT, a suite of crop-specific models for daily biomass accumulation and yield prediction [162]; and STICS, which emphasizes soil-plant-atmosphere interactions [163]. These models have been calibrated for cereals, legumes, vegetables, and industrial crops and are widely used for yield forecasting and climate-impact assessments.

Another group of plant growth models focuses on functional-structural plant modeling (FSPMs). These models illustrate the structure of plants, including leaves, stems, and branches, and connect this structure to processes such as photosynthesis and carbon distribution. They simulate how a plant's structure affects its functions, such as capturing light, and how these functions influence overall growth. In FSPMs, plants are treated as assemblies of organs, following natural rules for growth patterns. Carbon uptake is calculated at the leaf or canopy level, and resources are allocated among plant parts based on their specific needs. This approach allows for the simultaneous evolution of plant structure and function, reflecting their ongoing interactions. Notable examples of FSPMs include GreenLab, which models source-sink relationships during organogenesis and has been applied to maize, wheat, cotton, and tomato [164, 165]; L-system-based models that utilize formal grammars to simulate branching structures, applied to rice, grapevine, and *Arabidopsis* [166]; and OpenAleaAmapSim, which combines 3D architectural modeling with light interception analysis [167]. FSPMs are particularly valuable in situations where plant architecture significantly impacts productivity, such as in dense canopies or vine crops.

Another set of plant growth models concentrates on how the individual parts of a plant, such as roots, leaves, and stems, grow and interact with their environment. Root growth models, for example, examine how roots elongate, branch out, respond to gravity, and explore the soil. They also consider how plants absorb water and nutrients. Notable examples include RootBox, which captures 3D root architecture and branching patterns [168], and OpenSimRoot, which models root growth and resource acquisition in

heterogeneous soils [169]. Leaf expansion and canopy development models describe leaf area growth as a function of factors such as temperature, turgor pressure, and carbon availability. Examples include models for *Arabidopsis* and cereals [170], as well as canopy development simulations for greenhouse lettuce and spinach [171]. Stem and internode biomechanical models investigate how plants grow longer, bend, and maintain stability against gravity and wind. These models are valuable for assessing the risk of lodging when crops fall over, especially in cereal crops [172]. These organ-specific mechanistic approaches provide detailed insights into how individual organs contribute to overall plant performance and adaptation to environmental stresses.

A widely used family of mechanistic plant growth models employs sigmoidal curves to describe the growth process. These curves effectively capture the distinct phases of biological development: slow initial growth, rapid expansion, and eventual saturation. The mathematical simplicity of these models, combined with their biological relevance, makes them particularly valuable when growth follows a smooth, cumulative trajectory. Notable examples of these models include the logistic model, which is commonly applied to crops such as lettuce, spinach, tomato seedlings, and tree saplings, and which illustrates symmetric sigmoidal growth [173, 174, 154]. The Gompertz model, on the other hand, is used for cereals, leafy vegetables, and fruit crops; it depicts asymmetric growth characterized by rapid early expansion [175, 176]. Additionally, the Richards model generalizes both the logistic and Gompertz curves and has been applied to grapevines, fruit trees, and perennials [177]. Sigmoidal growth models are especially advantageous when destructive biomass sampling is limited or when summarizing growth requires only a few biologically interpretable parameters.

Mechanistic plant growth models provide biologically interpretable simulations of photosynthesis, respiration, and assimilate allocation. However, these models require extensive species-specific parameterization and rely on simplifying assumptions about canopy structure and stress responses, which might limit their ability to capture nonlinear behaviors and stress-induced responses, particularly in dynamic or unconventional environments [170, 178].

With the growing availability of high-throughput imaging and sensor data, researchers have begun to adopt data-driven approaches that can learn complex growth patterns directly from observations.

6.2.2 Data-driven Models for Plant Growth and Biomass Estimation

Unlike mechanistic models, which rely on explicit physiological equations, data-driven models learn statistical relationships directly from empirical measurements. These methods range from classical machine learning with hand-crafted features to DL architectures that extract complex spatial and temporal patterns from images.

Initial methods used handcrafted descriptors of morphology, color, and texture from RGB images, along with regression models such as SVR and Random Forests, to estimate biomass or leaf area. [179, 180, 181]. With the rise of DL, CNNs have become the primary method for extracting plant traits directly from images. CNN-based models have been employed for various tasks, including leaf counting, biomass estimation, and stress detection, all without the need for manual feature engineering [182]. Additionally, temporal extensions such as CNN-LSTM architectures enable modeling of growth dynamics from image sequences. These models capture developmental transitions and temporal dependencies [183].

Modeling approaches describe how plants grow and distribute biomass, but their accuracy largely depends on the quality of the phenotypic data used for parameterization, validation, and application. Modern imaging-based phenotyping plays a crucial role in both mechanistic and data-driven frameworks by providing the structural, temporal, and organ-level information necessary to accurately capture growth dynamics. To place these modeling approaches within the broader context of plant measurement, the

following section reviews phenotyping methods for specific organs and whole plants. It emphasizes the imaging modalities and trait-extraction techniques essential to contemporary growth analysis.

6.2.3 Phenotyping of Specific Plant Organs and Whole Plants

Modern phenotyping approaches target both specific organs, such as roots, leaves, and stems, and whole-plant or canopy-level traits. These methods rely heavily on imaging, enabling non-destructive, time-resolved monitoring of plant morphology and dynamics.

Root phenotyping has traditionally relied on transparent growth systems or soil-free media, which enable direct imaging of root architecture. Tools like RootNav automatically extract root length, branching patterns, and growth trajectories from 2D images [184]. More advanced 3D techniques, such as X-ray CT and MRI, facilitate the reconstruction of root systems within soil. Additionally, computational models like RootBox and OpenSimRoot simulate root architecture and soil exploration [168, 169].

Leaf and shoot phenotyping typically employs top or side-view RGB imaging to measure leaf area, canopy cover, plant height, and organ movement. Time-lapse imaging has been used to analyze developmental transitions and tropic responses. For instance, Samiei et al. demonstrated automated detection of seedling developmental stages from top-view time-lapse sequences using DL [183]. Similarly, Gall et al. used the SLEAP framework (developed for animal pose tracking) to track leaf tips, midpoints, and stem bases in *Arabidopsis*, beans, and sunflowers, thereby facilitating a detailed reconstruction of organ trajectories and growth dynamics [185].

Whole-plant and canopy-level phenotyping involves integrating multiple imaging methods. RGB imaging provides information on projected area and color indices. In contrast, 3D point clouds generated from LiDAR, structured light, or multiview stereo techniques allow for the estimation of plant height, volume, and surface area [186]. Additionally, multispectral and hyperspectral imaging capture important physiological traits, such as chlorophyll content and water status [187]. Recent datasets, such as GroMo25, illustrate how image sequences can be used to model leaf count, plant age, and growth trajectories in controlled environments [188].

Together, these phenotyping approaches provide rich structural and temporal information that complements mechanistic modeling and enables data-driven analysis of plant growth.

While data-driven models offer strong predictive performance and adaptability to new species or environments, they lack the interpretability and physiological grounding of mechanistic models. This has motivated hybrid approaches that combine mechanistic structure with machine-learning flexibility. Such integrated frameworks form the conceptual basis for the modeling strategy adopted in this chapter.

6.3 Phase 1: The Perception Validation Layer

As discussed in Section 6.1, reliable and frequent measurement of plant traits is fundamental for analyzing plant growth and responses to environmental conditions. This section constitutes “Phase 1” of the framework illustrated in Figure 6.1, focusing on the development and validation of the Perception Layer. The objective of this phase is to design and validate a 2D image-processing pipeline on diverse plant images. Its performance is benchmarked against the Phenospex system to assess the accuracy and reliability of the extracted morphological features. This validation step is essential because it ensures that the morphological traits used in subsequent modeling stages are scientifically sound and robust for real-world deployment.

Conventional manual and destructive measurement techniques are labor-intensive and unsuitable for continuous monitoring. Although high-precision commercial devices provide excellent measurements, their cost and laboratory-oriented design limit accessibility for many growers.

In response to these challenges, this section presents a cost-effective, image-based alternative implemented with PlantCV (Section 6.3.1.2) and Pixellib (Section 6.3.1.1). We validate our approach by comparing it with *Phenospex*¹, a well-known commercial 3D plant phenotyping system known for its high precision. This methodology serves as the foundational processing framework for analyses of *Salicornia*.

Our methodology involves a comprehensive processing framework that integrates image analysis, identifies Region of Interest (ROI)s, extracts key features, calibrates metrics, and validates results against a separate, reliable measurement system.

6.3.1 Technical Framework: Software Libraries for Phenotyping

To implement the automated perception pipeline, the framework integrates two complementary software environments, Pixellib and PlantCV, which are introduced in the following. Their combination enables robust plant segmentation and reliable phenotypic feature extraction, forming a reproducible pipeline that is validated on commercial hardware before being applied to the *Salicornia* study.

6.3.1.1 Instance Segmentation: Pixellib

To automate the identification of individual plants within a complex greenhouse environment, this methodology employs the Pixellib library [189]. As an advanced framework for instance segmentation, Pixellib provides the computational tools needed to accurately isolate plant tissue from background noise.

Image segmentation is a fundamental step in computer vision-based plant phenotyping, where accurately separating plant biomass from structural infrastructure is critical for reliable trait analysis. Pixellib is a Python library that simplifies semantic and instance segmentation tasks by providing ready-to-use implementations of state-of-the-art DL models. This library is built on TensorFlow and Keras, leveraging cutting-edge architectures for image segmentation, including DeepLabV3+ for semantic segmentation and Mask R-CNN for instance segmentation (Figures 6.2 and 6.3).

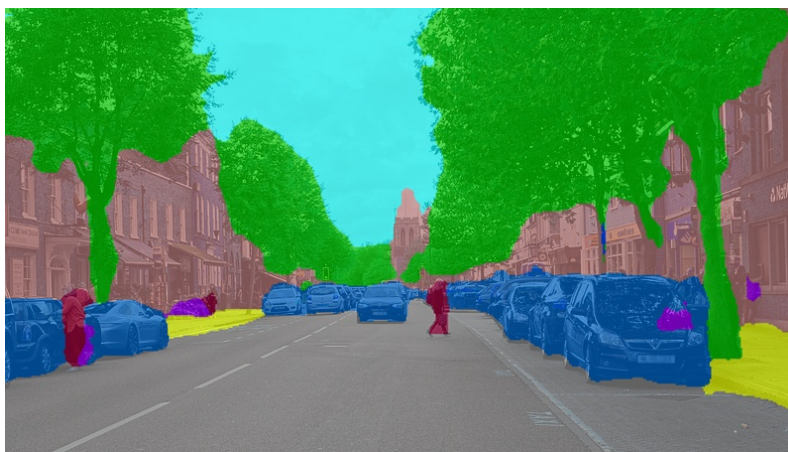


Figure 6.2: Semantic segmentation results produced by Pixellib.

In this study, we focused on Mask R-CNN, which integrates a region proposal network (RPN) together with a dedicated mask-prediction branch. This architecture allows the model to identify plant pixels and to distinguish individual plants even when they grow within the same container. In practice, these components work together to localize objects, classify them, and generate precise per-pixel masks.

The key features of Pixellib are summarized as follows:

¹<https://www.Phenospex.com/>

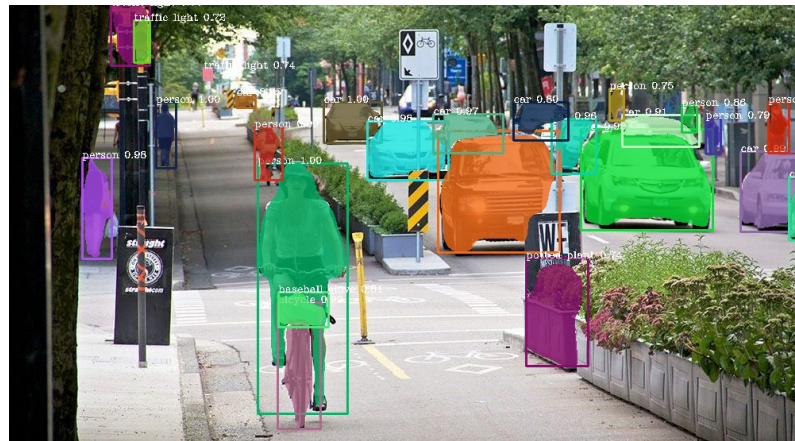


Figure 6.3: Instance segmentation results produced by Pixellib.

- **Instance Segmentation:** Unlike semantic segmentation, which classifies pixels by category, instance segmentation identifies and segments each object as a distinct entity. This capability is crucial for the multi-plant pot configurations used in the *Salicornia* study.
- **Pretrained Models:** The system utilizes a backbone model that is pre-trained on the COCO dataset (*mask_rcnn_coco.h5*). The COCO dataset is a large-scale benchmark containing annotated images for object detection and segmentation across 80 categories, widely used to train models like Mask R-CNN.
- **Class-Specific Filtering:** To enhance performance, the model is configured to detect only the “potted plant” class, effectively ignoring non-biological elements such as irrigation pipes, sensors, and greenhouse frames.
- **Transfer Learning Support:** The library supports fine-tuning for domain-specific tasks, enabling general-purpose models to adapt to the unique morphological characteristics of diverse plant species.

Pixellib enables automatic extraction of plant pixels from images, supporting high-throughput phenotyping workflows. A typical processing pipeline consists of the following steps:

1. **Input Acquisition:** Ingestion of raw RGB images.
2. **Model Selection:** Choice between semantic or instance segmentation models, depending on the application.
3. **Segmentation Output:** Generation of binary or multi-class masks that delineate plant structures.
4. **Post-processing:** Integration of the segmentation results into downstream morphological analysis pipelines (e.g., PlantCV).

This automated approach minimizes manual intervention and enhances the reproducibility of monitoring growth. By utilizing pre-trained models, the system offers immediate applicability while remaining flexible enough to be fine-tuned for specialized plant-specific datasets.

While these general-purpose models are not optimized for plant-focused applications by default, their ability to identify “potted plant” structures serves as a robust baseline for this thesis. Since the target species, *Salicornia*, consists of predominantly green vegetative tissue, the combination of Pixellib’s spatial isolation and color-space optimization provides a highly accurate proxy for biomass development.

6.3.1.2 The Morphological Analysis Framework: PlantCV

High-throughput plant phenotyping has become an essential tool in modern plant science, enabling fast and precise quantitative assessment of plant traits. A central component of the image-processing pipeline in this thesis is PlantCV, an open-source software package specifically designed for high-throughput plant phenotyping. PlantCV offers a suite of modular functions that facilitate the extraction of morphological, color, and shape-based features from plant images [190]. Its design allows for the automation of phenotyping pipelines, promoting reproducibility and scalability across large datasets. The library is implemented in Python and leverages established image processing libraries such as OpenCV and scikit-image. While PlantCV supports multiple imaging modalities, such as thermal and hyperspectral imagery, this study primarily uses its morphological analysis tools on standard RGB imagery.

This section focuses on PlantCV's morphological analysis capabilities, which are used to quantify the structural traits of *Salicornia*. These measurements provide essential insights into plant growth, development, and system responses to environmental stimuli.

6.3.1.2.1 Morphological Descriptors in PlantCV Morphological traits are fundamental descriptors of the plant phenotype. Quantifying features such as height, width, and canopy area enables assessment of developmental stages and reconstruction of longitudinal growth trajectories. Automated morphological analysis eliminates the subjectivity and labor intensity associated with traditional manual measurements. After generating binary masks, PlantCV is used to extract the following structural descriptors:

- **Shape-Based Features:** These metrics, including area, convex hull area, and solidity, offer a broad summary of canopy size and structural organization, capturing both the extent of the detected vegetation and the overall coherence of its spatial arrangement.
- **Dimensional Features:** These geometric descriptors, including width, height, and aspect ratio, summarize the overall spatial proportions of the canopy and provide a general indication of its horizontal-vertical balance and structural elongation.
- **Topological Features:** Skeletonization reduces the plant structure to a single-pixel-wide path to measure complexity and branching patterns.

PlantCV offers significant advantages in terms of accessibility and modularity. Unlike proprietary commercial systems that require significant capital expenditure, this open-source approach uses standard RGB hardware to provide highly reliable trait approximations, often comparable to 3D laser-scanning references. Furthermore, the software's modular architecture ensures the pipeline remains adaptable across different species and imaging conditions.

However, the methodology is subject to specific technical limitations. The accuracy of the morphological output is inherently tied to the quality of the initial segmentation; any errors in the binary mask directly propagate to the trait measurements. Complex architectures characterized by overlapping leaves or spatial occlusion in dense populations remain a challenge for 2D imaging, potentially leading to underestimation of traits. Additionally, while color-based segmentation is effective for vegetative species like *Salicornia*, it may exclude non-green tissues in other species.

6.3.2 Implementation of the Perception Pipeline

This section describes the implementation details of Phase 1, illustrated in Figure 6.4. The proposed pipeline integrates instance segmentation, color-space optimization, morphological analysis, and metric calibration to extract structural traits from standard RGB images.

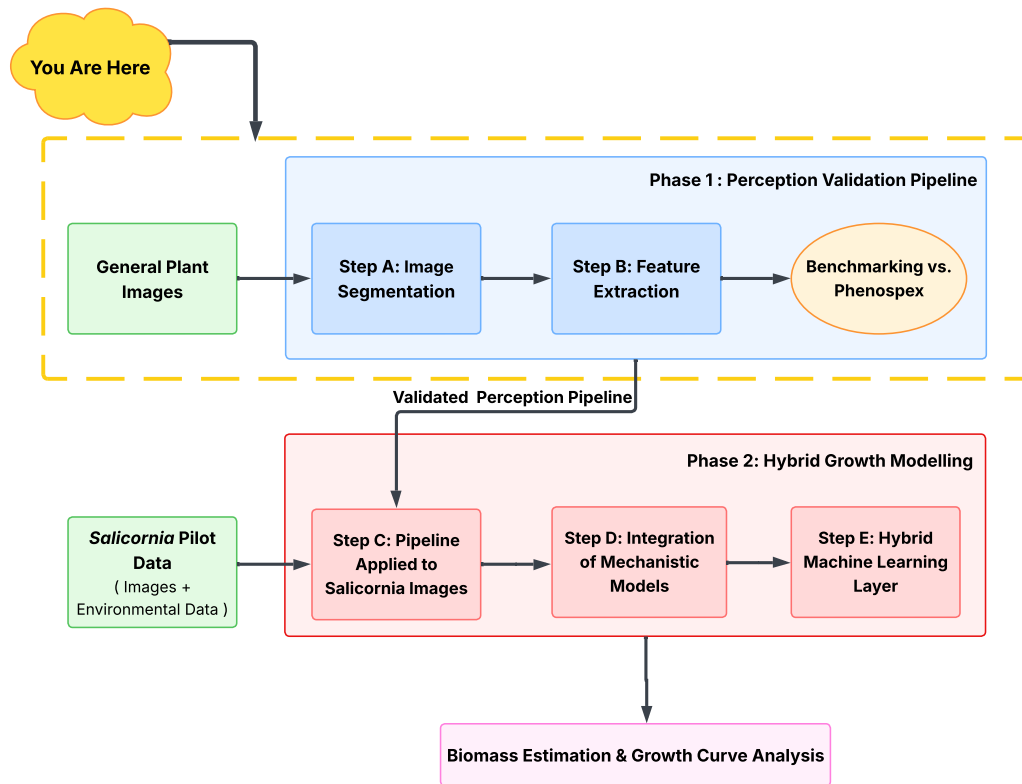


Figure 6.4: Overview of Chapter 6 Structure, Phase 1.

The workflow consists of four main stages:

- **Instance segmentation** to isolate plant pixels.
- **Color-space transformation** to enhance tissue visibility and improve thresholding.
- **Morphological trait extraction** using PlantCV.
- **Metric calibration** to convert pixel (px) measurements into physical units (millimeter (mm), mm^2).

This modular design, corresponding to Steps A and B in Figure 6.4, ensures that the proposed method remains adaptable to different plant species, imaging conditions, and camera configurations. Furthermore, the benchmarking against Phenospex is described in detail in Section 6.3.3.

6.3.2.1 Instance Segmentation

Utilizing the Pixellib framework described in Section 6.3.1.1, the acquisition images were processed to define the ROI which is the bounding boxes generated to isolate individual plants from the background. To ensure that only plants were retained, the segmentation model was configured to detect only the “potted plant” class, effectively filtering out all other objects in the scene. This configuration ensured that the resulting segmentation masks corresponded solely to the targeted plants.

As illustrated in Figure 6.5, for each detected plant, Pixellib returned both a bounding box and its associated instance mask, enabling precise ROI extraction for subsequent processing. The segmentation outputs provided:

- **Bounding boxes** delineating each plant, accompanied by a confidence score indicating detection reliability.
- **Instance masks** defining the relevant plant regions.
- **Clear separation** of plant tissue from background clutter.

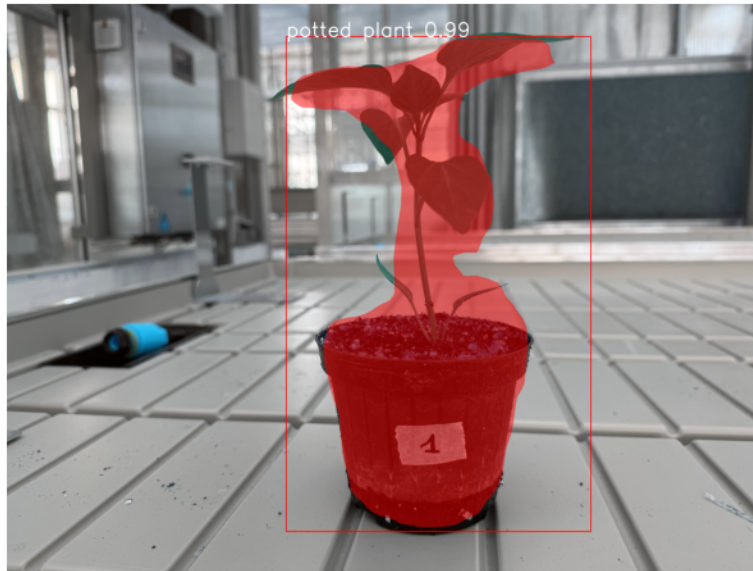


Figure 6.5: Representative instance segmentation output using Pixellib, showing the detected potted plant. Bounding boxes and masks define the Region of Interest for subsequent analysis.

We then utilized the ROI mask generated by Pixellib to ensure that all further operations with PlantCV focused only on the plants, effectively excluding any noise, soil and background from our trait computations.

6.3.2.2 Color-Space Optimization for Tissue Detection

Choosing an appropriate color space from an RGB image can enhance the visibility of plant features and improve subsequent processing steps. Figure 6.6 displays different color spaces corresponding to a sample plant.

Various color spaces offer distinct properties:

- **HSV (Hue, Saturation, Value):** Separates color into hue (type of color), saturation (color intensity), and value (brightness), enabling robust segmentation under varying lighting conditions.
- **CIELAB (L^* , a^* , b^*):** Perceptually uniform color space where L^* represents lightness, a^* the green–red axis, and b^* the blue–yellow axis; This arrangement is particularly useful for distinguishing plant tissue from its background.
- **CMYK (Cyan, Magenta, Yellow, Key/Black):** Subtractive color model used in printing, representing colors as combinations of cyan, magenta, yellow, and black; useful for isolating specific color components in image analysis.

As shown in Figure 6.6, plant segments are more easily distinguishable from the background and surrounding environment in the a^* channel of the CIELAB color space compared with other channels. Research supports this observation and shows that CIELAB can achieve much better results for separating plants from soil than HSV or RGB, especially when lighting conditions are not ideal [191]. Unlike RGB,

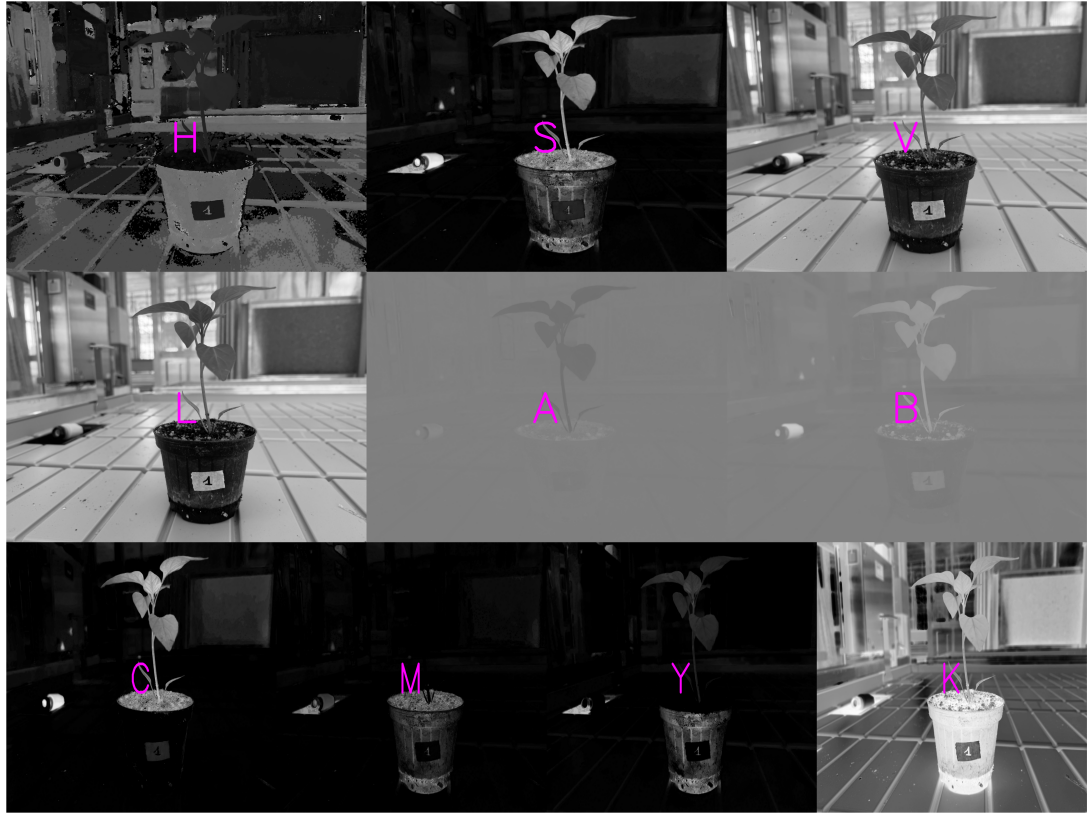


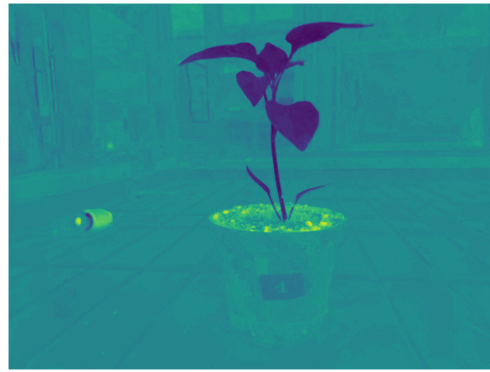
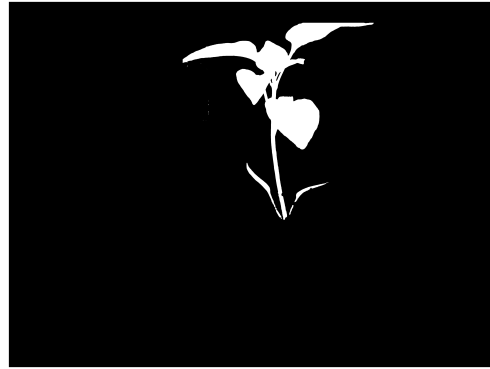
Figure 6.6: Sample plant visualized in different color spaces (HSV, CIELAB, and CMYK) used for segmentation and feature extraction.

which combines color and intensity, the CIELAB color space separates luminance (L^*) from chromatic components (a^* and b^*). This separation is particularly advantageous for plant segmentation. The a^* channel highlights green-magenta differences, enhancing the detection of vegetation, while the b^* channel captures blue-yellow differences, which are helpful for distinguishing soil or pots. The L^* channel represents lightness, helping to mitigate the effects of variable lighting or shadows. Additionally, CIELAB is perceptually uniform, meaning that numerical changes in its channels closely correspond to human perception of color differences, making it robust for detecting subtle variations in plant tissue coloration [192]. PlantCV exploits this property by enabling thresholding on individual channels to improve segmentation accuracy [190].

As illustrated in Figure 6.7, the segmentation of plant regions involved generating a Black and White (BW) mask through thresholding. To facilitate this process, the a^* channel from the CIELAB color space was first inverted. In this inverted representation, plant regions appear brighter than the background, allowing a single global threshold to effectively isolate the plants. An appropriate threshold value was first applied to generate the initial BW mask, as shown in Figure 6.7(c). To refine this mask, small spurious objects were removed using a size-based threshold that filters out components below a minimum area. Finally, the ROI obtained from PixelLib segmentation was applied to suppress remaining noise and irrelevant regions, yielding the cleaned mask illustrated in Figure 6.7(d).

6.3.2.3 Morphological Trait Extraction

After generating binary masks from the segmented plant images, structural features were extracted using PlantCV. Different image perspectives were employed depending on the trait of interest: plant height was

(a) a^* channel of the original image(b) Inverted a^* channel highlighting plant regions(c) Thresholded BW mask of the inverted a^* channel

(d) Final BW mask after ROI masking and noise removal

Figure 6.7: Segmentation process using the CIELAB a^* channel

estimated from front-view images, while plant convex hull area was computed from top-view images.

Plant height was measured using the front-view BW masks. An example of the size analysis output produced by PlantCV is shown in Figure 6.8, where the plant's vertical extent is quantified in px units. Conversely, the plant convex hull area was derived from top-view images to characterize the plant canopy's horizontal spatial extent. A representative example of the original top-view image and its corresponding BW mask is shown in Figure 6.9. The PlantCV size analysis for the top-view measurements is presented in Figure 6.10. In this example, the plant width is estimated as 2,131 px, and the convex hull area is 2,404,237.5 px².

6.3.2.4 Metric Calibration

The structural measurements provided by the Phenospex system are expressed in physical units (mm), whereas the proposed image-based pipeline initially produces measurements in px units. To enable a direct comparison between the two methods, a pixel-to-millimeter conversion was therefore required. Under standard conditions, this calibration is achieved by introducing reference objects of known dimensions (e.g., rulers or calibration markers) within the imaging scene. However, no dedicated calibration targets were available in the experimental setup considered in this study.

As a practical alternative, the pot's known physical dimensions were used as reference measurements for metric calibration. While this approach enables approximate conversion to physical units, it assumes uniform image scaling and does not explicitly account for camera perspective or lens distortions.

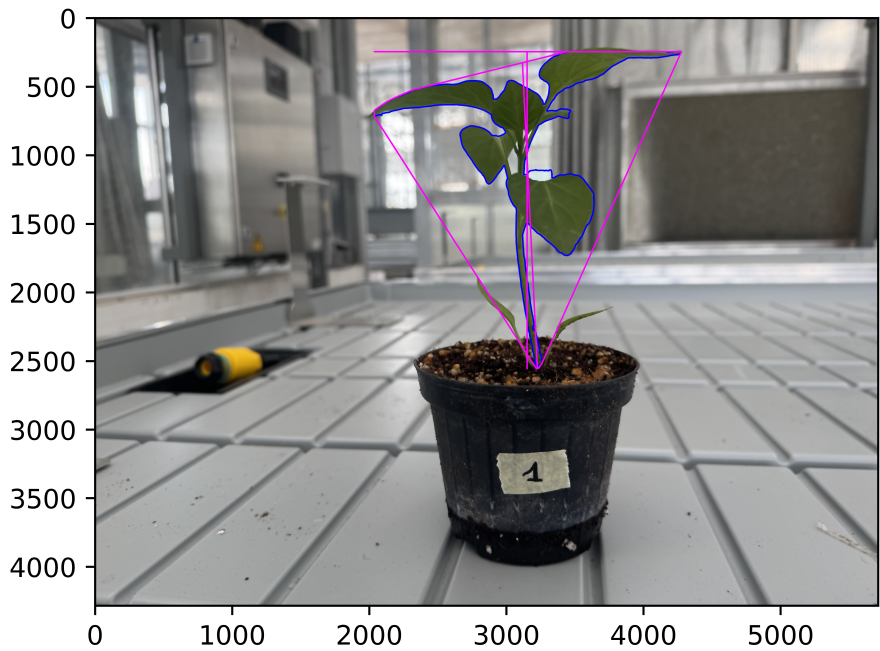
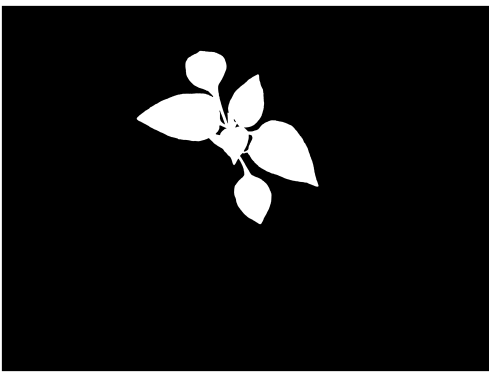


Figure 6.8: PlantCV size-analysis output showing plant height estimated from the front view (2,312 pixel).



(a) Original top-view image



(b) Top-view BW mask

Figure 6.9: Top-view images used for plant width and convex hull computation: (a) original image and (b) corresponding binary mask.

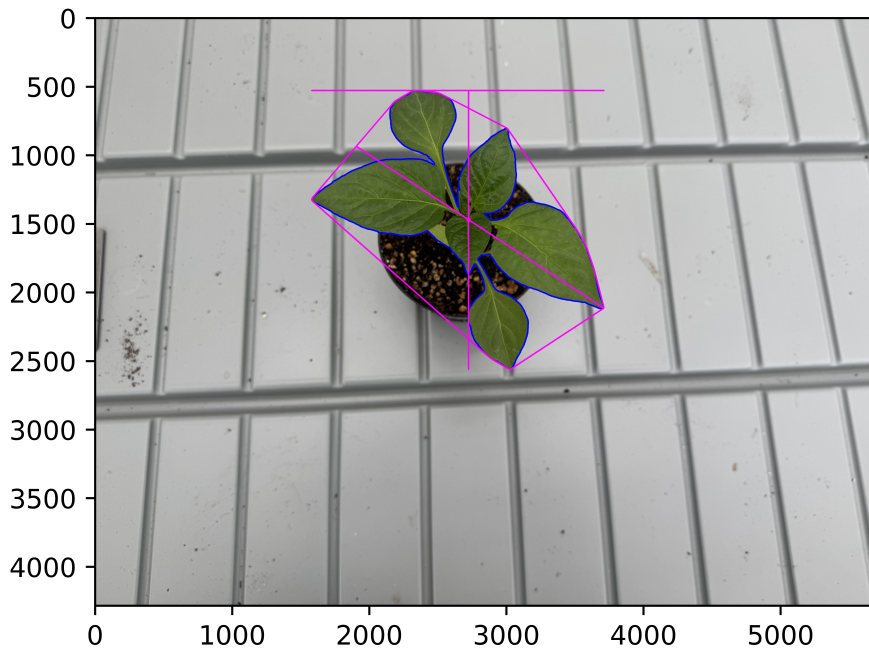


Figure 6.10: PlantCV size analysis output for top-view measurements (width: 2,131 px; convex hull area: 2,404,237.5 px²).

6.3.2.4.1 Pot height estimation in pixel units To estimate the pot height in px units, the bounding box of the potted plant was first obtained using Pixellib segmentation (see Figure 6.5). The height of the pot was then inferred by subtracting the detected plant height, computed as described in Section 6.3.2.3, from the total bounding box height of the potted plant. This procedure provides an estimate of the pot height in px without requiring explicit pot segmentation.

A pixel-to-millimeter scaling factor was subsequently computed as the ratio between the known physical pot height (in mm) and its corresponding px height. This scaling factor was applied to convert pixel-based plant height measurements into millimeters.

6.3.2.4.2 Pot detection for top-view calibration To enable conversion of convex hull area measurements, the pot was explicitly detected in the top-view images. For top-view measurements, the scaling factor was determined by fitting a least-squares circle to the pot rim in the CMYK magenta channel. More precisely, given the pot's approximately circular shape, an image-based detection strategy was adopted (Figure 6.11). The RGB top-view image was first converted to the CMYK color space, and the magenta (M) channel was extracted because it provided enhanced contrast between the pot and the background. Otsu's thresholding method was then applied to generate a binary mask.

The nonzero pixels of the resulting mask were used to compute a convex hull approximating the pot boundary. A circle was then fitted to the convex hull points using least-squares, yielding the pot center and radius in px units. The output corresponding to Figure 6.11 indicates a detected center at (2672.24, ; 1616.26) px with an estimated diameter of 1131.97 px.

A pixel-to-millimeter scaling factor was subsequently computed as the ratio between the known physical pot diameter (in mm) and its corresponding px radius.

6.3.2.4.3 Convex hull area conversion Convex hull area measurements, initially computed in px units from top-view images, were converted to square mm using the mentioned pixel-to-millimeter scaling factor

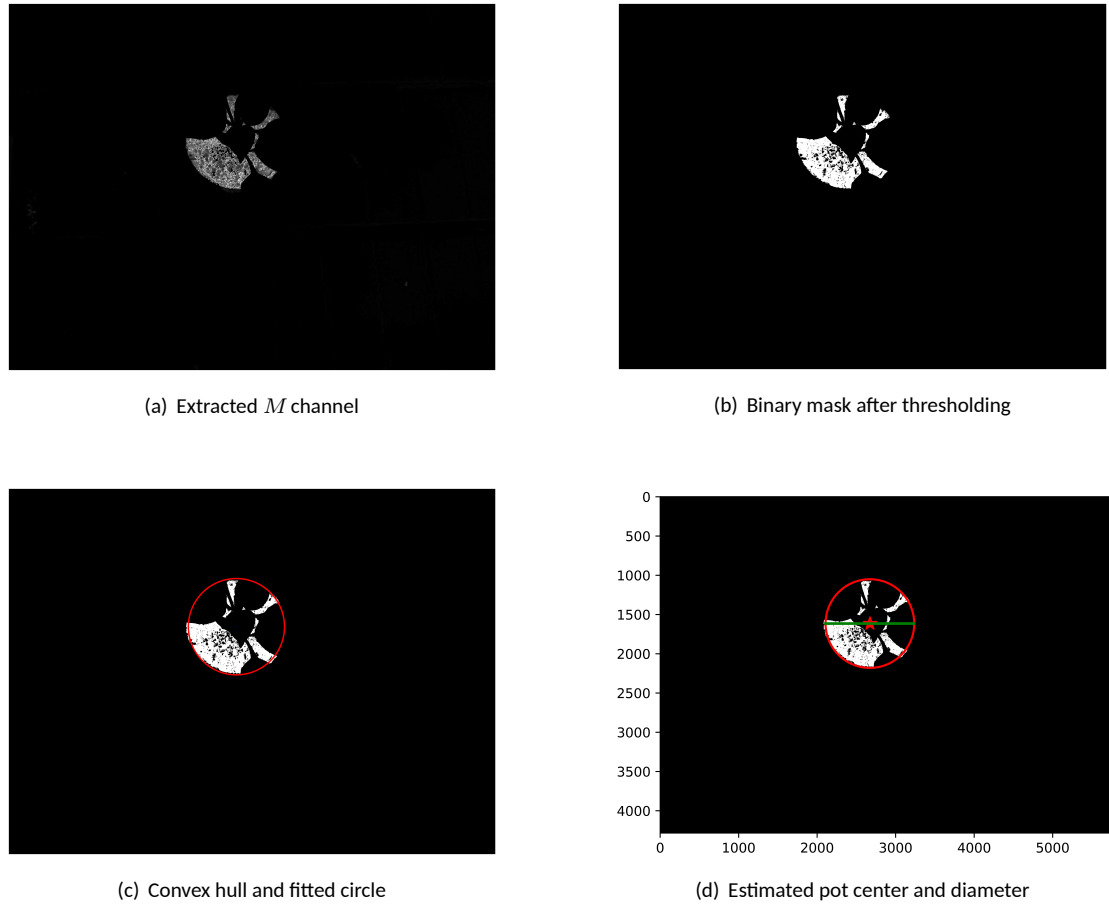


Figure 6.11: Image-based estimation of pot diameter used for pixel-to-millimeter calibration in top-view images.

obtained from pot-based calibration:

$$A_{mm^2} = A_{px^2} \times \left(\frac{D_{known}}{D_{measured}} \right)^2 \quad (6.1)$$

where D_{known} is the physical diameter in mm and $D_{measured}$ is the diameter in px.

This conversion assumes isotropic scaling along both image axes and allows canopy-related metrics to be expressed in physical units comparable to those produced by the Phenospex system.

Although this calibration strategy is approximate, it provides a practical solution in the absence of dedicated calibration targets and enables meaningful validation of the proposed method against a high-precision 3D laser-scanning reference.

6.3.3 Methodological Validation against Phenospex

To evaluate the accuracy of the proposed image-based pipeline, we compared the aforementioned measurements with Phenospex (Figure 6.12), a commercial 3D multispectral laser-scanning platform commonly used in high-throughput plant phenotyping. At the heart of Phenospex's systems is the PlantEye F600, which generates dense 3D point clouds enriched with spectral reflectance information, enabling precise extraction of key structural and physiological traits, including plant height, width, convex hull, leaf area, and spectral indices.



Figure 6.12: Phenospex 3D scanning system for high-precision plant trait measurement under controlled environments.

Phenospex systems are known for their high measurement accuracy and substantial capital cost, placing them in the category of high-end commercial phenotyping solutions. The objective of this validation is to assess whether the proposed low-cost, RGB-based pipeline can approximate Phenospex measurements for two key structural traits: height and convex hull area.

Table 6.1 reports the comparison results for 8 plant instances in three plant types. In Table 6.2, absolute errors are summarised as mean $\pm$ SEM (Standard Error of the Mean), where the Standard Error of the Mean (SEM) quantifies the uncertainty associated with the estimated mean error and reflects how precisely the sample mean represents each species.

Table 6.1: Comparison of Convex Hull Area (mm^2) and Plant Height (mm) between the Proposed Method (PM) and Phenospex (PHX).

Plant	Convex Hull (PM)	Convex Hull (PHX)	Height (PM)	Height (PHX)
Pepper 1	13,556.3	10,764.4	128.1	127.9
Pepper 2	11,063.2	12,950.8	116.8	117.6
Pepper 3	17,682.6	12,956.8	142.3	151.4
Cabbage 1	13,550.5	12,487.9	90.7	84.6
Cabbage 2	10,482.5	11,926.4	94.4	74.9
Cabbage 3	8,154.6	7,975.1	87.9	75.1
Begonia 1	13,025.5	19,341.2	134.9	133.8
Begonia 5	10,698.9	17,669.1	144.3	111.5

Table 6.2: Mean $\pm$ SEM of absolute errors for Convex Hull Area and Plant Height across species.

Species	Convex Hull Error (mm^2)	Height Error (mm)
Pepper	3,135.08 $\pm$ 852.98	3.37 $\pm$ 2.88
Cabbage	895.34 $\pm$ 307.63	12.74 $\pm$ 3.87
Begonia	6,642.99 $\pm$ 327.50	16.95 $\pm$ 15.84

The comparative analysis reveals that the proposed 2D image-based method provides a highly reliable approximation for vertical plant traits, particularly in species with distinct vegetative structures. For Pepper samples, the height estimation demonstrated exceptional accuracy, with a MAE of only 3.37 ± 2.88 mm,

representing minimal deviation from the high-precision Phenospex 3D reference. While the error slightly increased for Cabbage (12.74 ± 3.87 mm) and *Begonia* (16.95 ± 15.84 mm), the results remain within an acceptable range for longitudinal growth monitoring.

In contrast, the estimation of the convex hull area exhibited a clear sensitivity to plant morphology and spectral characteristics. While the error for Cabbage was relatively low (895.34 ± 307.63 mm²), a significant systematic underestimation was observed in *Begonia* ($6,642.99 \pm 327.50$ mm²).

The relatively high errors observed for *Begonia* can be attributed to its morphological characteristics. *Begonia* plants often produce flowers in various colors that contrast sharply with the green foliage. Since PlantCV primarily relies on color, or spectral-based segmentation, to distinguish plant tissue from the background, non-green structures, such as flowers, are often excluded from the detected plant mask. As a result, specific plant structures are not captured, leading to underestimation in traits such as convex hull area. This limitation underscores the need to further refine the segmentation pipeline for such plant species, for example by incorporating multispectral indices, machine-learning classifiers, or species-specific color models to ensure that flowers and other non-green tissues are included in the analysis. Since the target species investigated in this work, *Salicornia*, is characterized by predominantly green vegetative tissue and lacks visually prominent non-green structures such as flowers, the limitations associated with color-based segmentation have a negligible impact on the proposed analysis. Consequently, addressing this issue is beyond the scope of the present thesis.

Additionally, the unit conversion procedure employed in this study, based on reference objects such as pot height or diameter, introduces an additional source of imprecision. While practical in the absence of dedicated calibration targets, this approach assumes uniform scaling and does not account for potential distortions due to camera perspective or lens characteristics. A more precise calibration strategy would involve using rulers, grids, or fiducial markers with known dimensions within the imaging scene, ensuring robust, accurate pixel-to-millimeter conversion across all images. Implementing such measures would reduce scaling errors and improve the reliability of trait estimation in future studies.

Despite these limitations, the proposed method offers clear advantages. Unlike Phenospex, which requires specialized, costly hardware and proprietary software, the proposed approach relies on open-source tools such as PlantCV and PixelLib. It can be applied to standard RGB images acquired with low-cost cameras. This makes the method highly accessible to a wide range of research groups, particularly those with limited resources. While Phenospex delivers highly precise, automated 3D measurements, the results presented here demonstrate that the proposed method can provide comparable estimates of key structural traits at a fraction of the cost. With continued refinement of segmentation algorithms and calibration strategies, this approach represents a promising, cost-effective alternative for commercial plant phenotyping.

Having validated the image-based perception pipeline against commercial 3D-scanning standards and established that the extracted morphological features are a reliable proxy for physical growth, we now transition from the data-acquisition and validation pipeline to biological modeling. In the following sections, this validated pipeline is used to process *Salicornia* images and develop a hybrid framework that estimates continuous biomass growth dynamics.

6.4 Phase 2: Hybrid Growth Modeling for *Salicornia*

Following the validation of our perception algorithms in Phase 1 (Section 6.3), where 2D feature extraction was benchmarked against commercial hardware, we now transition to Phase 2, illustrated in Figure 6.13.

As highlighted in this figure, our focus shifts from general plant datasets to the specific environmental and biological data generated by the BeBlue pilot-scale aquaponic pilot. This physical testbed provides

the multimodal data necessary to implement the hybrid growth models and finalize the DT framework for *Salicornia*.

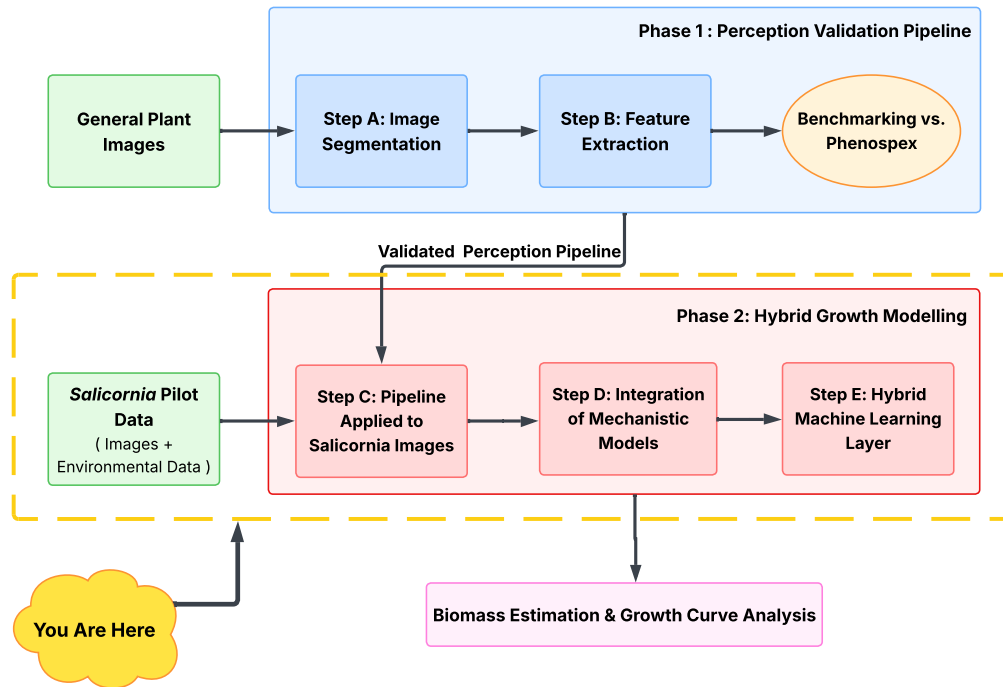


Figure 6.13: Overview of Chapter 6 Structure, Phase 2.

The architecture of this DT framework is divided into four primary phases: Data Acquisition via the multi-camera system (Section 6.4.2); Segmentation and Feature Extraction using the validate image processing pipeline described in Section 6.3 on *Salicornia* images (Sections 6.4.3.3, 6.4.4 and 6.4.5); Growth Modeling through logistic curve fitting and critical point analysis (Section 6.4.6); and finally, Biomass Prediction using the Gradient Boosting regressor (Section 6.5.3).

6.4.1 Pilot Structure of the BeBlue Aquaponic Systems

The method developed in this chapter is implemented and evaluated in the BeBlue pilot-scale aquaponic system, which acts as the physical testbed for data acquisition, state estimation, and DT validation.

The BeBlue project established two pilot-scale aquaponic systems as experimental benchmarks for developing a DT framework. These pilots serve as the physical layer for real-time state estimation, closed-loop management, and model validation within a resource-efficient circular economy. The project utilizes two distinct ecological configurations to test the robustness of the management algorithms:

- **IP-UNIVE (Marine Configuration):** Located at Ca' Foscari University of Venice. This system focuses on the co-production of seabream (*Sparus aurata*) with Ulvarigo and *Salicornia* spp, presenting unique challenges in salinity-driven control.
- **IP-UNILJU (Freshwater Configuration):** Located at the University of Ljubljana, utilizing rainbow trout (*Oncorhynchus mykiss*) and lettuce (*Lactuca sativa*).

6.4.1.1 BeBlue Aquaponic System Topology and Components

As illustrated in Figure 6.14, the system architecture is defined by four primary interconnected functional units:

1. **Fish Tank (FT):** The aquaculture subsystem acting as the primary nutrient injector (source) via fish metabolism.
2. **Mechanical Filter (MF):** A solid-liquid separation unit for the removal of suspended solids.
3. **Greenhouse (GH):** Hosts plants or algae acting as the nutrient sink and oxygen generator.
4. **Biofilter (BF):** A fixed-bed reactor housing nitrifying bacteria for the conversion of Total Ammonia Nitrogen (TAN) into Nitrate (NO_3^-).

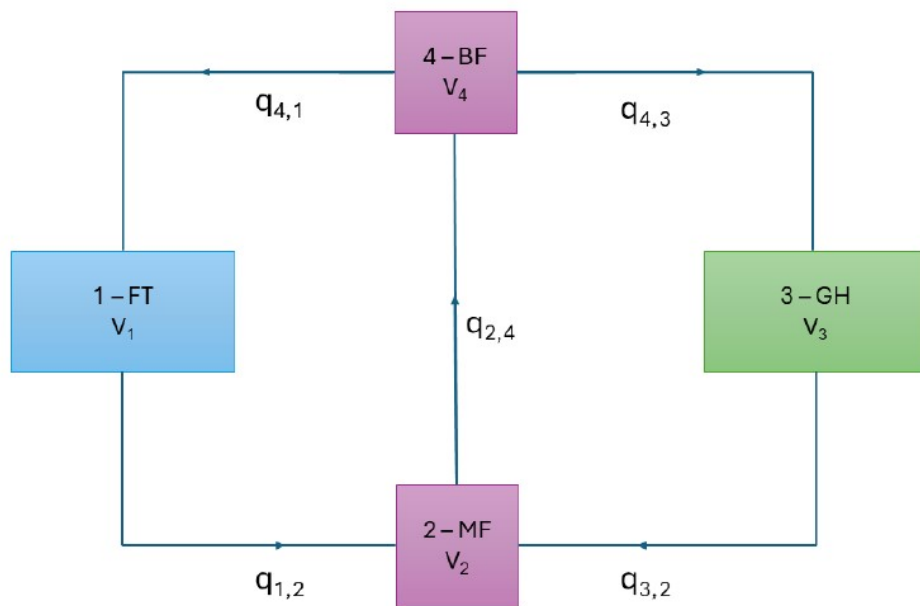


Figure 6.14: Main components of the marine aquaponic pilot plant are located at the scientific Campus of Ca' Foscari University: Fish Tank (FT); Mechanical Filter (MF); Green House (GH); BioFilter (BF).

These components are linked via a recirculating Aquaponic System loop. Actuators, including circulation pumps, LED arrays, chillers, and blowers, enable active modulation of the system's physical states. Figure 6.15 and Figure 6.16 provide the physical layout and biomass cultivation details at the UNIVE site.

6.4.1.2 Information Layer and Data Quality Control

The monitoring infrastructure adopts a multi-rate sampling strategy to capture system dynamics across different timescales:

- **High-Frequency (Automatic):** IoT sensors capture pH, temperature, DO, and salinity at 5–15 minute intervals.
- **Low-Frequency (Manual):** Biochemical parameters (TAN , NO_3^- , NO_2^- , PO_4^{3-}) and biomass stocks are sampled daily to weekly, representing the “hidden states” that require periodic manual synchronization.

The total data acquisition rate corresponds to approximately 2,000–2,500 data points per day per pilot.



Figure 6.15: View of the hydroponic greenhouse at the UNIVE pilot site, showing the aquaponic integration of fish tanks (Right), and plant cultivation beds (Left) used for continuous environmental monitoring.

6.4.1.3 BeBlue Aquaponic Modeling Framework: The Digital Twin Core

The aquaponic system is conceptualized as a coupled, nonlinear dynamical system in which the evolution of biological and chemical states is governed by first-order mass-balance principles. Rather than treating components in isolation, the DT framework integrates the complex interactions between the biotic state vector, comprising fish biomass, plant/algal growth, and nitrifying bacterial populations, and the abiotic state vector, which tracks water volume, DO, nitrogen species (TAN and nitrate), and suspended solids.

The system trajectory is determined by the interplay between stochastic environmental forcing (eg, ambient temperature, humidity, and solar radiation) and controllable operational inputs (eg, feed frequency, artificial lighting intensity, and hydraulic loading rates). By formalizing these nutrient transformations and energy exchanges through simulation, the DT serves as a predictive engine. It enables the assessment of biomass accumulation and system stability by reconciling real-time sensor data with the expected physiological behaviors of the organisms within the loop.

6.4.2 Automated Imaging Infrastructure for Monitoring *Salicornia* Growth

A dedicated imaging infrastructure was deployed in the greenhouse section of the IP-UNIVE pilot facility at Ca' Foscari University of Venice to enable continuous monitoring of *Salicornia* plants. This system was designed to overcome the limitations of manual phenotyping, which is inherently low frequency, labor-intensive, and unable to capture rapid morphological changes. By integrating an autonomous sensing layer, the platform provides high-temporal image streams essential for quantifying growth dynamics and supporting data-driven modeling of *Salicornia* biomass development.

6.4.2.1 Monitoring System Architecture and Operational Reliability

The monitoring nodes consisted of five ESP32-CAM modules (Figure 6.17), each integrating a 2 MP OV2640 sensor. These units operated as edge devices, performing autonomous image capture and network-based



Figure 6.16: *Salicornia* plants cultivated on net supports in the UNIVE hydroponic greenhouse.

transmission. From a system-management standpoint, maintaining uninterrupted operation of the monitoring platform was essential. To achieve this, the custom firmware incorporated multiple fail-safe mechanisms designed to preserve data acquisition. This custom firmware implemented several mechanisms:

- **Fault Tolerance:** A hardware watchdog timer was utilized for autonomous recovery from software hangs.
- **Network Resilience:** An exponential backoff algorithm was employed for WiFi re-establishment to prevent network congestion during recovery phases.
- **Data Integrity:** Local buffering enabled recovery of data packets after transient connectivity losses.

The system's digital backbone was designed to handle large amounts of data reliably over a long period. We used a modern data-handling framework (FastAPI) that allowed the system to receive images from all five cameras simultaneously every 15 minutes, without any "traffic jams" or lost files. Additionally, by using a technology called "containerization" (Docker) [193], we ensured that the software settings stayed identical from the first day to the last. This was crucial for plant science, as it guaranteed that the timing and labels for every *Salicornia* image were perfectly consistent throughout the entire six-month experiment.

To ensure the dataset's temporal accuracy, all monitoring nodes were synchronized via the Network Time Protocol. This synchronization was managed by the backend server, which assigned a Coordinated

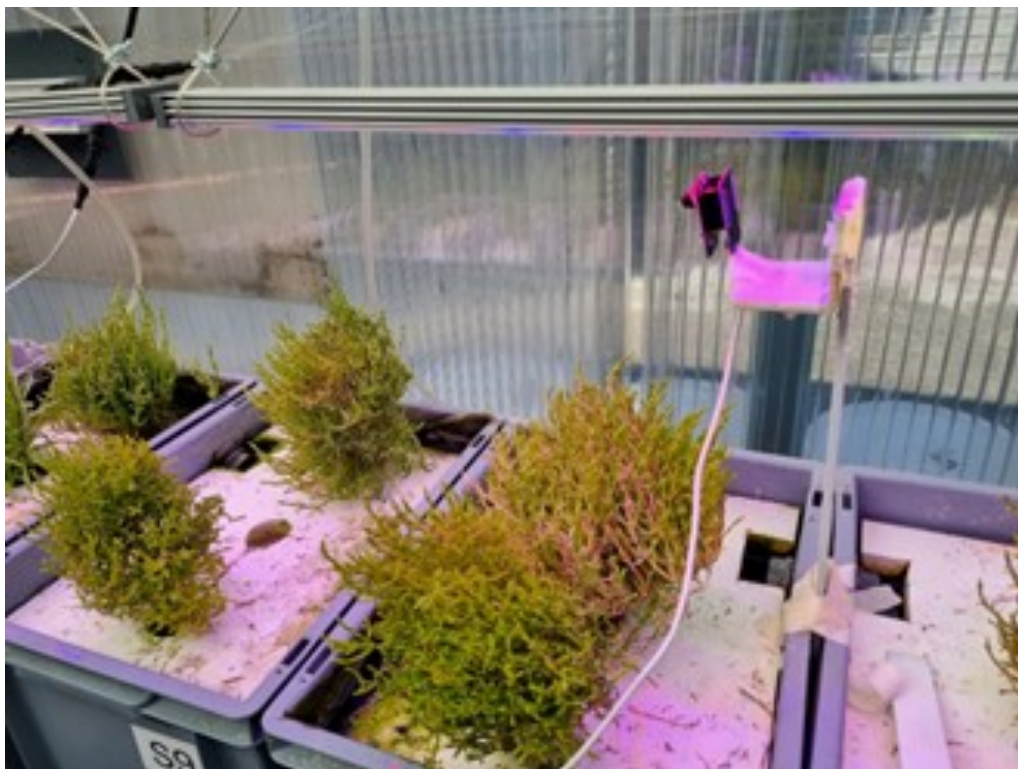


Figure 6.17: Camera installation used in the IP-UNIVE greenhouse for continuous *Salicornia* image acquisition.

Universal Time (UTC) timestamp to each image upon ingestion. By maintaining consistent clock alignment across the distributed ESP32-CAM network. This temporal precision was essential for the subsequent temporal filtering and photoperiod-based analysis, ensuring that day-night transitions were accurately identified across the experimental time window.

6.4.2.2 Image Acquisition Protocol

The acquisition system operated continuously from May to October 2025. Despite environmental stressors such as thermal fluctuations and electrical storms, the platform achieved an average operational uptime of 98.3%. Network stability was monitored throughout the deployment; devices recorded between 13 and 23 reconnection events, all of which were successfully managed by the firmware's exponential backoff algorithm. The resulting dataset comprises approximately 61,313 images, providing a high-resolution temporal record of *Salicornia* development. This corpus served as the primary input for the detection, segmentation, and biomass estimation models discussed in the following sections.

6.4.2.3 Image Availability Across Growth Phases

Two acquisition phases were defined according to the *Salicornia* growth cycle:

- **Phase I (Established Growth):** Plants were already in a mid-growth state at the start of monitoring.
- **Phase II (Full Life-cycle):** This phase captured approximately the entire developmental trajectory from planting. Natural biological variability was more pronounced, and some individuals terminated growth early due to germination failure or weak establishment.

Table 6.3 provides a comprehensive overview of the availability periods for the images. To maintain the robustness of subsequent data-driven models, only continuous sequences with high spatial clarity were

retained. Samples identified as having establishment failure (e.g., germination failure or early seedling death) were excluded from the final training set. The exclusion of these trajectories was essential to avoid biasing the non-linear regression models with biological artifacts that do not represent the species’ actual growth potential. Similarly, sequences affected by canopy occlusion, where the lateral expansion of neighboring plants rendered individual-pixel-area calculations unreliable, were truncated to ensure that the extracted morphological features reflected true plant vigor rather than spatial crowding effects.

Table 6.3: Summary of *Salicornia* image acquisition periods across Phase I and Phase II.

Sensor Node	Plant Position	Phase I (Mid-Growth)	Phase II (Full Life-cycle)
Cam1	Right	2025-05-29 – 2025-06-16	2025-06-19 – 2025-08-26
	Center	2025-05-29 – 2025-06-11	2025-06-18 – 2025-08-26
	Left	Spatial Occlusion	Establishment Failure
Cam2	Right	Spatial Occlusion	Establishment Failure
	Center	2025-05-29 – 2025-06-13	2025-06-18 – 2025-08-26
	Left	2025-05-29 – 2025-06-13	Establishment Failure
Cam3	Right	2025-05-29 – 2025-06-06	2025-07-25 – 2025-08-31
	Center	2025-05-29 – 2025-06-06	Establishment Failure
	Left	2025-05-29 – 2025-06-04	Establishment Failure
Cam4	Right	Spatial Occlusion	2025-06-18 – 2025-08-26
	Center	Spatial Occlusion	2025-06-19 – 2025-08-06
	Left	2025-05-29 – 2025-06-04	2025-06-29 – 2025-08-12
Cam5	Right	Sensor Displacement	2025-06-23 – 2025-08-12
	Center	Sensor Displacement	2025-06-24 – 2025-08-12
	Left	Sensor Displacement	Establishment Failure

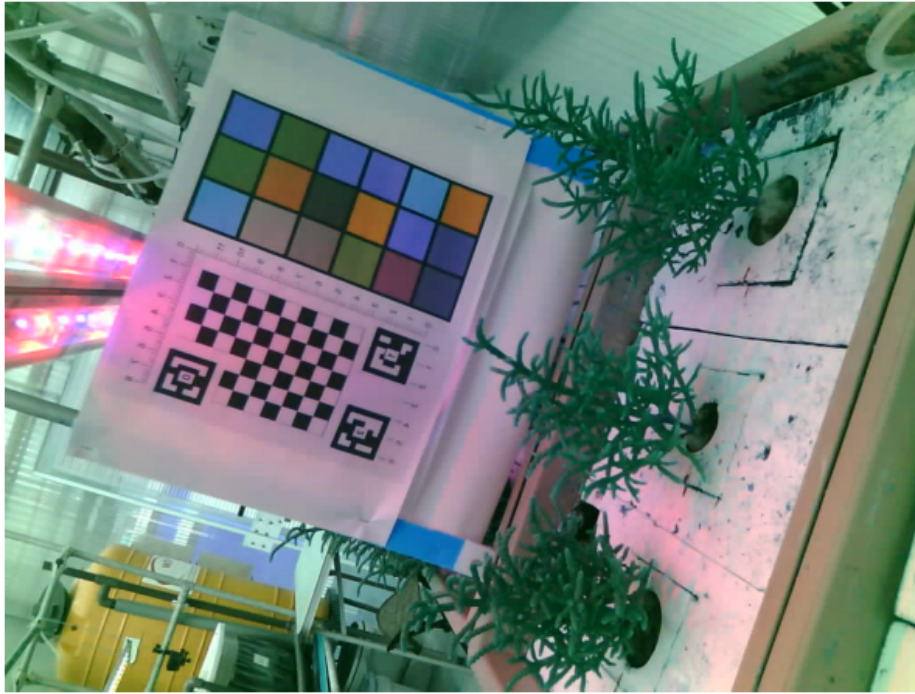
6.4.3 Digital Image Conditioning and Spatial Normalization

The image preprocessing pipeline was designed to enhance data consistency and reliability before feature extraction and downstream analysis. It consisted of two main operations: correcting image orientation to ensure uniform alignment across the dataset, and filtering nighttime images based on acquisition timestamps to retain only frames with sufficient illumination.

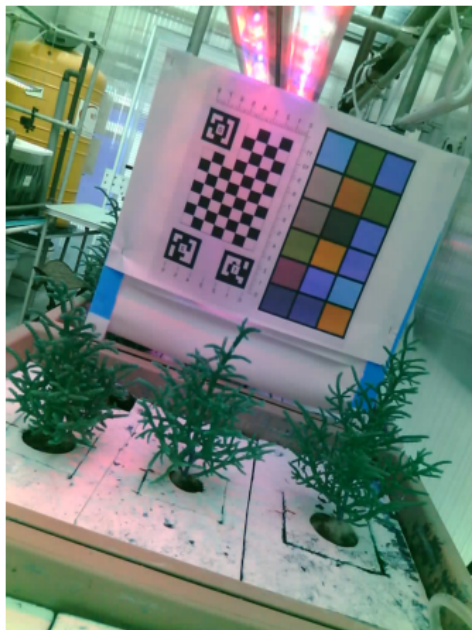
6.4.3.1 Automated Angular Correction

As shown in Figure 6.18(a), images acquired from different cameras exhibited varying angular misalignment due to differences in camera mounting configurations. To ensure a consistent orientation across the entire corpus, a two-stage calibration process was implemented:

1. **Coarse Alignment:** A static correction was applied based on the predefined installation orientation of each camera (Figure 6.18(b)).
2. **Algorithmic Refinement:** To ensure that the pot and pattern sheet appeared horizontally aligned in every image, the algorithm estimated the dominant geometric orientation of the scene. Canny edge



(a) Raw acquisition: Initial image exhibiting angular misalignment from the sensor mounting.



(b) Coarse alignment: Application of a static camera-specific correction factor.



(c) Fine calibration: Final spatial normalization (19.44°) derived from structural reference lines.

Figure 6.18: Multistage angular calibration pipeline: From raw sensor output to a normalized reference frame suitable for automated phenotyping.

detection was applied to extract strong structural contours, with particular emphasis on the high-contrast horizontal edges of the pot rims and the pattern sheet. The orientations of the minimum-area bounding rectangles fitted to these features were then analyzed, and their median angle was used as the correction factor. This approach leverages stable, non-biological structures to normalize image orientation, preventing drift caused by plant growth or minor sensor vibrations (Figure 6.18(c)).

All transformations were performed using the OpenCV and NumPy frameworks, ensuring that the resulting image series remained spatially consistent throughout the entire experimental period.

6.4.3.2 Temporal Filtering

Images acquired during the dark period were excluded to maintain the reliability of the segmentation algorithms, as low-light conditions resulted in insufficient contrast for accurate tissue detection. This temporal filtering was performed using acquisition timestamps synchronized via the backend server. The exclusion window was empirically defined (e.g., 19:00 to 05:00) to retain only frames with full natural or artificial illumination for biomass estimation.

6.4.3.3 Spatiotemporal Plant Segmentation

To analyze individual development within a multi-plant pot configuration (Figure 6.19(a)), a spatiotemporal segmentation framework was implemented. The primary challenge in long-term monitoring is the lateral expansion of the canopy, which can cause neighboring plants to intersect.

The process began by detecting spatial ROIs for the left, center, and right plant positions using only the first image of each sequence. Initial plant masks were generated using a Mask R-CNN model from the PixelLib library (Section 6.3.1.1), providing high-precision segmentation during the early growth stages (Figure 6.19(b)). To avoid re-segmenting every subsequent frame, the ROIs were propagated forward in time using a linear temporal expansion strategy. The expansion rate was empirically calibrated by observing lateral and vertical growth patterns over a full imaging period. By applying a constant scaling factor to the bounding box dimensions, the system ensured that the entire canopy remained within the computational window while minimizing interference from adjacent plants. This systematic isolation enabled each plant to be extracted as an independent sub-image series (Figure 6.19), allowing the derivation of individual growth trajectories rather than aggregated pot-level measurements.

This segmentation framework is spatiotemporal because it preserves the biological identity of each individual throughout the entire experimental timeline. By anchoring the initial detection (spatial component) to a fixed coordinate system and evolving the boundaries through a linear temporal scaling (temporal component), the system ensures that every extracted sub-image is linked to a specific, unique plant. This approach allows the transition from simple snapshot analysis to the reconstruction of longitudinal growth trajectories, in which the variance in biomass can be traced back to the specific developmental history of each individual plant.

This spatiotemporal segmentation strategy represents one of the important methodological advancements in this work. Unlike traditional per-frame segmentation methods, which often fail when plants overlap or deform over time, the proposed framework maintains the identity of each individual plant throughout the entire experiment. It achieves this by combining single-frame spatial initialization with tracking of ROIs over time. As a result, it enables reliable long-term monitoring in densely populated multi-plant environments and allows reconstruction of true individual growth trajectories rather than merely aggregated pot-level trends. This identity-preserving segmentation is crucial for understanding population variability and serves as a vital foundation for creating biologically meaningful DTs.

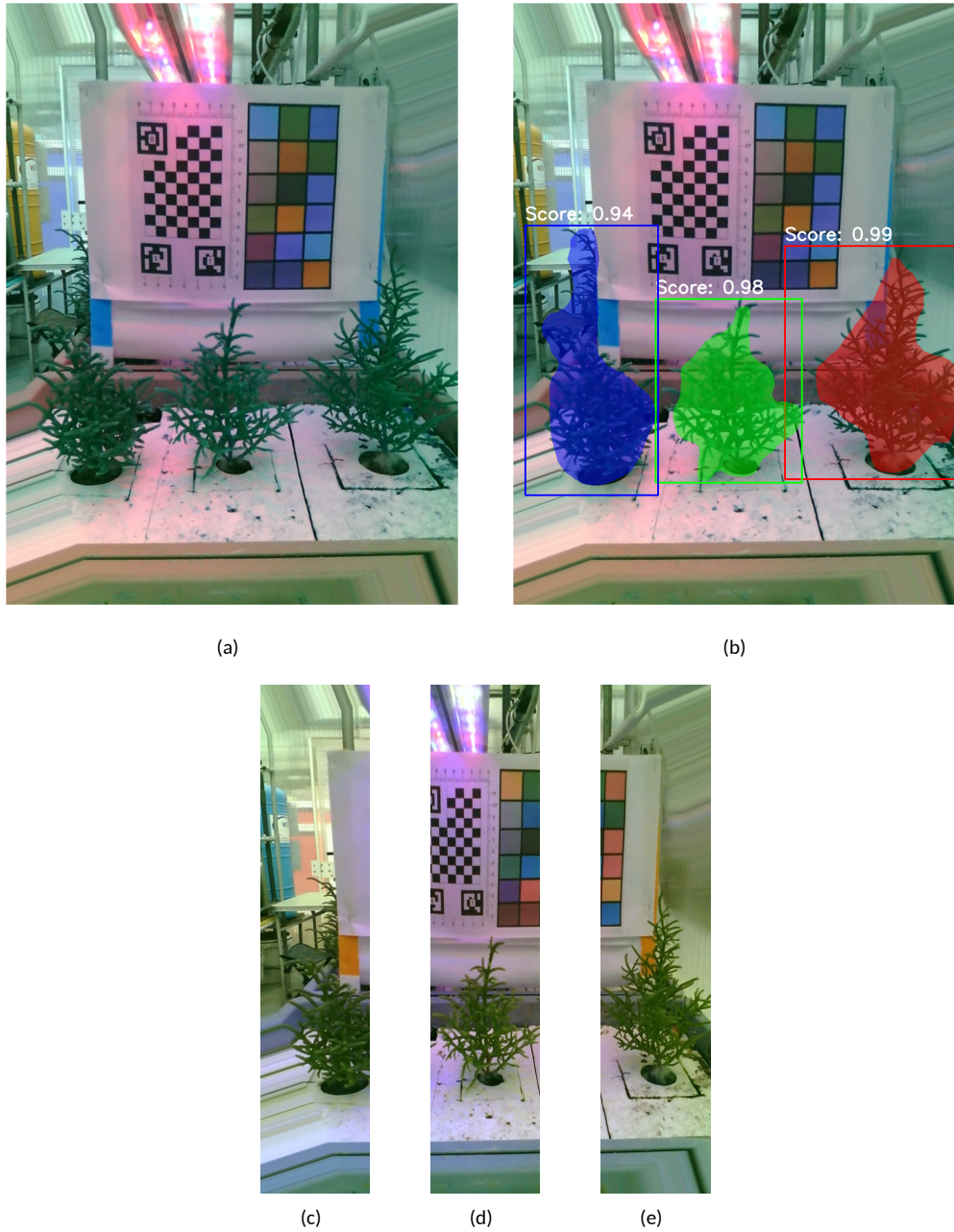


Figure 6.19: Spatiotemporal segmentation and instance isolation pipeline. (a) Original input frame; (b) Instance segmentation using PixelLib; (c), (d), (e) Final isolated plant instances.

6.4.3.4 Metric Calibration: Pixel-to-Millimeter Mapping

To enable quantitative comparison across the distributed sensor network, pixel-based measurements were mapped to physical units (mm). A planar reference grid was positioned behind each pot and used as a spatial calibration anchor (Figure 6.20). By comparing the known physical spacing of the grid with the detected pixel distance, a scaling factor was derived to normalize installation tolerances in camera-to-target distances.

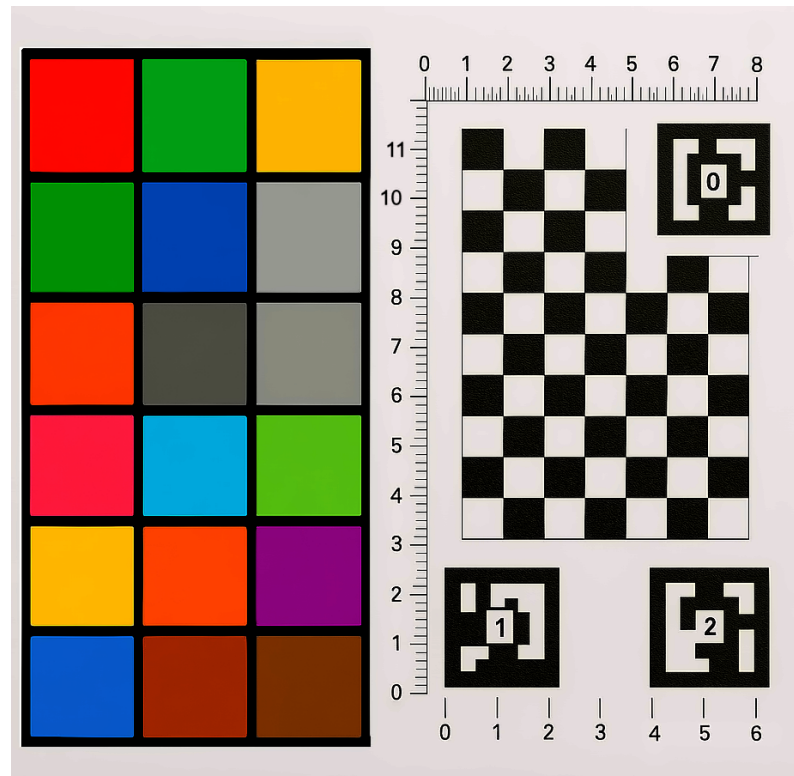


Figure 6.20: Pattern sheet used for spatial calibration and pixel-to-millimeter ratio estimation.

Because full camera calibration parameters (e.g., focal length, lens distortion, and principal point) were not available, this approach relies on a geometric approximation. The method assumes that cameras were installed with approximately consistent orientation and minimal tilt relative to the plant plane. Although minor deviations in camera angles may introduce some measurement uncertainty, the controlled installation setup and use of the reference pattern provide a consistent basis for scale estimation.

The potential sources of uncertainty include:

- **Angular Misalignment:** Minor deviations in the camera's tilt relative to the reference plane.
- **Lens Distortion:** Potential scale variance which could cause local deviations in scale, especially at image edges.

This conversion method enabled the integration of all plant measurements from different cameras into a unified dataset expressed in millimeters. Despite inherent uncertainties, the approach provides a practical and robust solution for quantifying structural traits under field or semi-controlled imaging conditions where full calibration data are unavailable.

In the following section, the procedure for detecting and extracting the pattern sheet patches used in this scaling process is described in detail.

6.4.3.5 Automated Detection of Calibration Reference Patches

To automate the derivation of the scaling factor, a computer vision pipeline was developed to identify the reference geometry within the pattern sheet. The procedure integrates classical image processing techniques to ensure high localized precision. The detection procedure was implemented in Python using OpenCV and NumPy, following the steps outlined below.

The detection routine is constrained to a Static ROI to reduce computational overhead and prevent false positives caused by plant biomass. The input image is selected such that the pattern sheet is clearly and unambiguously visible. The overall procedure follows a structured and systematic sequence:

1. **Preprocessing:** Image is converted to grayscale and smoothed using a Gaussian kernel (σ adjusted for noise reduction) to prepare for stable thresholding.
2. **Binarization:** A global thresholding operation is applied to isolate the high-contrast patches from the background (Figure 6.21(b)).
3. **Morphological Refining:** Morphological closing operators are utilized to bridge structural gaps and ensure the topological integrity of the detected patches.

Contours are extracted from the binary mask and approximated as polygons to verify quadrilateral topology. Candidates are retained only if they exhibit quadrilateral topology and satisfy a minimum area threshold, effectively isolating the calibration markers from background noise.

To ensure a deterministic relationship between the detected patches and the physical grid, the components are sorted lexicographically (top-to-bottom, left-to-right). The largest identified patches are selected as the primary geometric reference for the pixel-to-millimeter conversion, providing a robust baseline for the structural quantification of the plants.

This procedure was applied to all cameras, and the four largest detected patches were selected to compute the scaling factor for converting pixel dimensions to millimeters.

6.4.3.6 Depth-Corrected Scaling Factor Estimation

To translate pixel-based observations into metric units (mm), a scaling model was developed to account for the “axial distance” between the sensor, the reference plane, and the target biomass.

The initial scaling ratio (σ_r) was derived by comparing the projected dimensions of the detected rectangular patches (Section 6.4.3.5) with their known metric references (D_{ref}). This ratio represents the spatial resolution at the depth of the pattern sheet (Z_r):

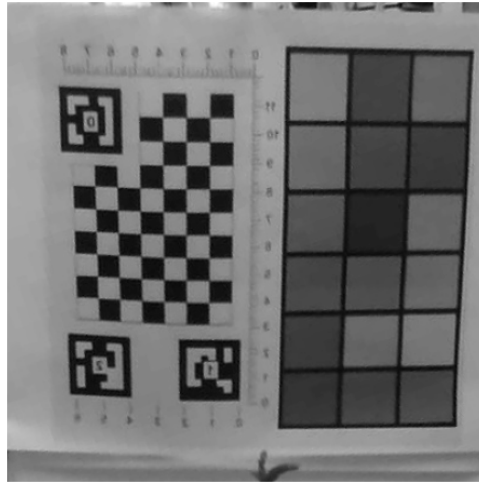
$$\sigma_r = \frac{D_{ref} [\text{mm}]}{D_{pixel} [\text{px}]} \quad (6.2)$$

Consistent with the pinhole camera model, the projected size of an object is inversely proportional to its distance from the optical center. Therefore, to estimate the scale at the plant's focal plane (Z_p), a depth-compensation factor was applied:

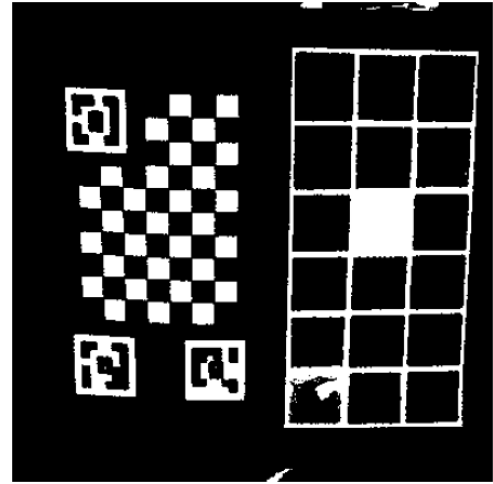
$$\sigma_p = \sigma_r \times \frac{Z_p}{Z_r} \quad (6.3)$$

where:

- σ_p denotes the corrected scaling factor (mm/px) at the plant position;
- Z_p represents the measured distance from the camera lens to the plant canopy;



(a) Grayscale pattern sheet



(b) Binary mask of pattern sheet



(c) Detected pattern patches

Figure 6.21: Sequence of the automated patch detection routine: (a) Localized grayscale enhancement; (b) Binary segmentation after morphological refinement; and (c) Final quadrilateral identification and spatial indexing.

- Z_r indicates the measured distance from the camera lens to the reference pattern sheet.

By ensuring that $Z_p < Z_r$, the resulting scaling factor σ_p is appropriately reduced, preventing the artificial inflation of biomass-related measurements. This calibration procedure was performed independently for each of the five sensor nodes in order to account for differences in individual installation geometries. For each acquisition, horizontal and vertical scaling ratios were computed and then averaged using a multi-patch consensus approach, leveraging up to 4 detected reference rectangles to mitigate quantization and localization errors.

Overall, this procedure provides a robust, depth-corrected metric basis for converting extracted

morphological traits, such as plant height and projected canopy area, into standardized real-world units.

6.4.4 Morphological Feature Extraction

In the feature extraction stage, the segmented plant instances were processed to extract morphological descriptors, converting each plant image into quantitative traits suitable for growth modeling. This step transforms raw visual data into a structured representation that supports statistical analysis and temporal modeling. Figure 6.22 illustrates the complete feature extraction pipeline described in this section.

6.4.4.1 Multi-Stage Binary Segmentation

Following the framework described in Section 6.3.2.3, a two-stage masking strategy was applied to ensure high-fidelity feature extraction.

1. **Instance Masking:** The plant ROI was first identified using the same Pixellib-based instance segmentation workflow introduced in Section 6.4.3.3. This ROI served as the spatial constraint for all subsequent feature-extraction steps (Figure 6.22(b)).
2. **Chromatic Thresholding:** A binary plant mask was then generated using the color-space optimization procedure described in Section 6.3.2.2. As summarized earlier, the inverse of the CIELAB a^* channel was thresholded to isolate vegetation from the background (Figures 6.22(c) and 6.22(d)). The resulting mask was refined using the previously extracted ROI to remove residual noise and maintain spatial consistency (Figure 6.23(a)).

The dual-masking approach leverages Mask R-CNN's robust spatial localization to eliminate background clutter, while chromatic thresholding ensures the final biomass estimate is based on precise pixel-wise vegetation detection.

6.4.4.2 Quantification of Structural Descriptors

After generating the binary plant mask, structural descriptors were extracted using the PlantCV framework (Section 6.3.1.2) to characterize the plant's geometric and spatial development. These metrics serve as the biological system's physical state variables and form the basis for growth analysis. The following descriptors were computed for each binary mask:

- **Area (A):** projected canopy area in px^2 (Figure 6.23(d)).
- **Convex Hull Area (A_{conv}):** area of the smallest convex polygon enclosing the plant px^2 (Figure 6.23(d)).
- **Solidity:** ratio A/A_{conv} describing canopy compactness.
- **Width (W):** maximum horizontal extent of the plant in px.
- **Height (H):** maximum vertical extent of the plant in px.
- **Aspect Ratio:** ratio W/H indicating elongation.
- **Path Lengths:** skeletal path measurements representing structural complexity and size in px.
- **Branch Count:** number of detected branch segments from plant skeletonization.
- **Tip Count:** number of terminal endpoints in the plant skeleton.

For species with complex morphologies, such as *Salicornia*, which feature high-density, intersecting, and succulent branches, skeletal traits (Path Length, Branch Count, and Tip Count) may not reflect absolute biological counts. Instead, these parameters are treated as proxies for structural complexity. They provide a relative metric for monitoring the transition from simple seedling architectures to the highly branched, mature canopy typical of succulent halophytes.

The combination of these structural descriptors provides a comprehensive quantitative basis for analyzing plant morphology and assessing growth variability under different experimental conditions. All features were extracted for the full image set listed in Table 6.3 and stored in per-camera, per-position CSV files for subsequent modeling and analysis.

6.4.4.3 Dimensional Normalization and Scaling

As established in Section 6.4.3.4, morphological features extracted in pixel units must be normalized into a metric coordinate system (mm) to allow for cross-camera comparison. This normalization is critical to account for the unique installation geometry and focal depth of each sensor node.

Following the depth-correction procedure described in Section 6.4.3.6, the dimensional normalization process relied on the independent derivation of vertical ($\sigma_{p,h}$) and horizontal ($\sigma_{p,w}$) scaling factors to ensure metric accuracy across both image axes. These factors were calculated by comparing the known physical height and width of the reference patches (D_{ref}) with their respective pixel dimensions ($D_{pixel,h}$ and $D_{pixel,w}$) detected during the automated patch extraction phase. By treating the horizontal and vertical scales as independent variables, the system effectively compensated for minor perspective tilts and potential non-square-pixel geometries inherent in the OV2640 sensors. In practice, the calculated difference between $\sigma_{p,h}$ and $\sigma_{p,w}$ across the five sensor nodes was found to be negligible (eg, < 1%), confirming that the cameras were mounted approximately parallel to the reference plane and that the image sensors exhibited a standard 1:1 pixel aspect ratio.

$$\begin{aligned}
 H_{mm} &= \sigma_{p,h} H_{px} \\
 W_{mm} &= \sigma_{p,w} W_{px} \\
 A_{mm^2} &= (\sigma_{p,h} \sigma_{p,w}) A_{px^2} \\
 A_{conv, mm^2} &= (\sigma_{p,h} \sigma_{p,w}) A_{conv, px^2}
 \end{aligned} \tag{6.4}$$

where H_{px} and W_{px} denote the plant dimensions in pixels, and A_{px^2} and A_{conv, px^2} represent the projected and convex hull areas, respectively. This transformation ensures that all measurements are mapped to a standardized metric space, enabling the reliable aggregation and comparison of individual growth trajectories across the entire distributed sensor network.

6.4.5 Feature Aggregation and Data Fusion

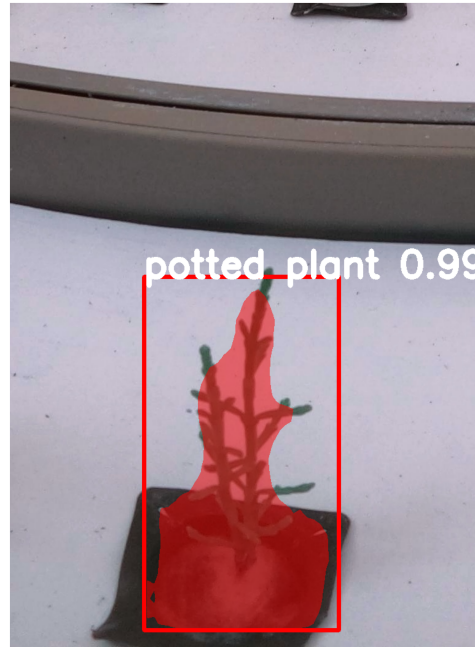
To enable a thorough analysis of growth dynamics, we developed a data processing pipeline that combines image-derived morphological features with environmental and biomass data. This step ensures that the resulting dataset is temporally synchronized and statistically prepared for predictive modeling.

6.4.5.1 Environmental Feature Engineering and Integration

High-resolution abiotic variables were incorporated into the feature set to account for environmental factors influencing plant development. As described in Section 6.4.1.2, water quality parameters, such as pH, water temperature, DO, and salinity, were recorded at intervals of 5 to 15 minutes at both the inlet ($S_x Load$) and



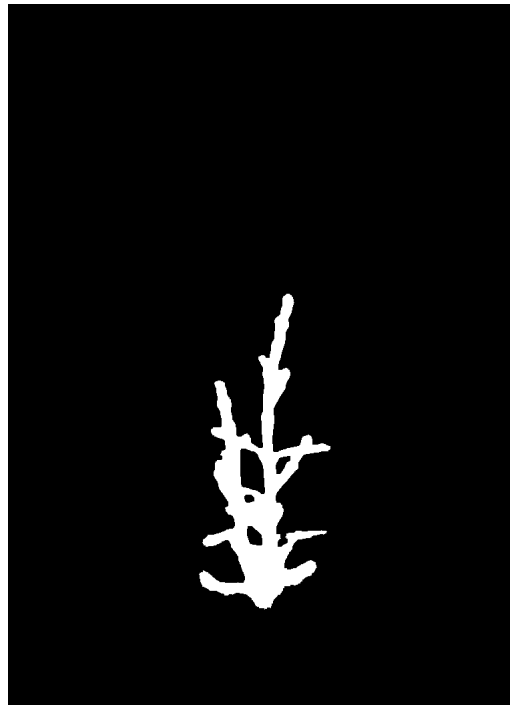
(a) Original plant image



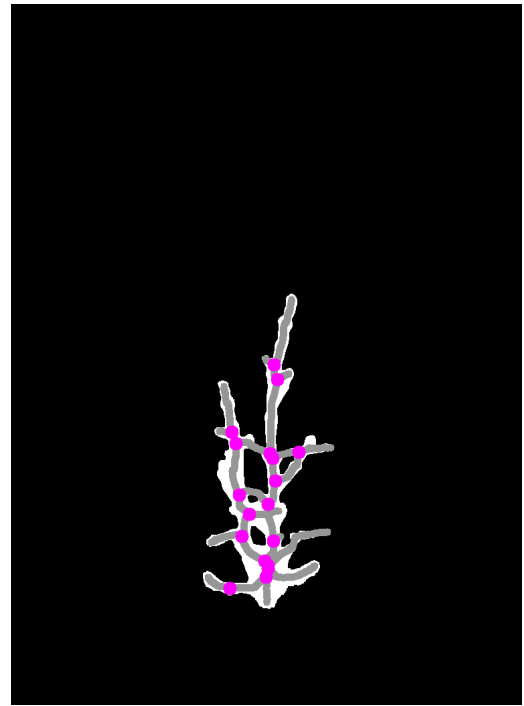
(b) Instance segmentation with ROI overlay

(c) α^* channel (CIELAB)(d) Inverted α^* channel for thresholding

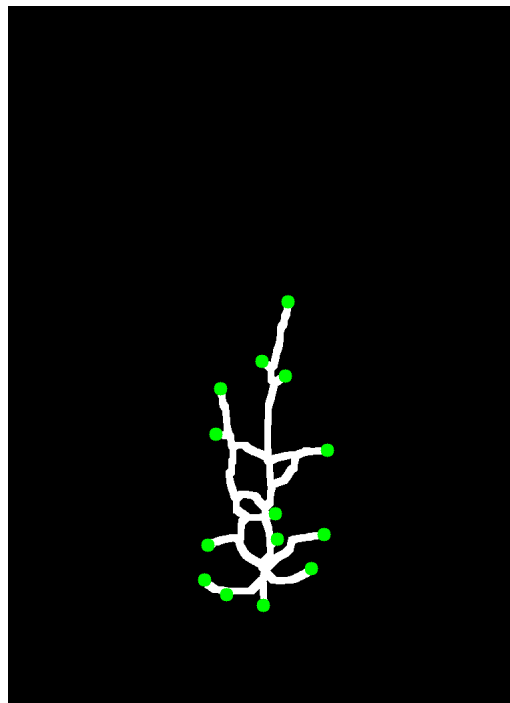
Figure 6.22: Feature extraction pipeline: instance segmentation and chromatic preprocessing (Part I).



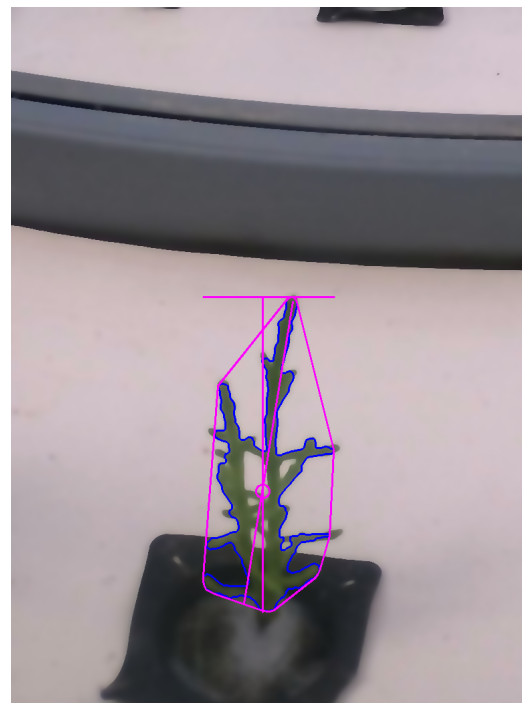
(a) Binary plant mask



(b) Skeletonized plant structure



(c) Detected skeletal tips



(d) Convex hull and area estimation

Figure 6.22: Feature extraction pipeline: binary masking, skeletonization, and geometric quantification (Part II).

the outlet (S_xWaste) of the plant tanks. These measurements, available through the pilot data portal², were temporally aligned with each image using timestamp matching.

6.4.5.1.1 Environmental Data Cleaning and Quality Control Raw environmental measurements were initially processed using a protocol for detecting outliers based on specified ranges. Specific physical limits for each parameter were established according to known biological and chemical constraints in aquatic cultivation systems. For example, pH values were restricted to a valid range of 0 to 14, as measurements outside of this interval indicate a sensor malfunction rather than a true chemical condition [194]. Values that exceeded these limits were flagged as anomalous and corrected using linear time interpolation.

6.4.5.1.2 Environmental Feature Engineering To characterize the interactions between plants and their environment, synchronized measurements were categorized into two main feature types:

- **Baseline Features:** These are the raw inlet values (S_xLoad), representing the initial state of the nutrient solution.
- **System Impact Features (Δ):** These represent the differences between outlet and inlet values, calculated as $\Delta = S_xWaste - S_xLoad$. This differential captures the net metabolic activity of the plant, where a negative Δ in dissolved oxygen, for instance, serves as a proxy for root respiration rates [195].

Since plant growth is influenced by cumulative rather than instantaneous environmental conditions, each variable was further refined into 24-hour and 72-hour rolling averages. These smoothed features encapsulate the integrated thermal and chemical history that governs physiological processes, including photosynthesis, respiration, and nutrient uptake [196].

6.4.5.2 Temporal Resampling and Biomass Alignment

To align the high-frequency image-derived features with the periodic manual biomass measurements, a two-stage temporal alignment strategy was employed.

1. **Daily Aggregation:** Since plant growth occurs gradually, hourly structural measurements do not accurately reflect growth dynamics. To align the high-frequency image data with the biological timescale of plant growth, all extracted morphological features were resampled to a daily resolution. This temporal aggregation serves as a low-pass filter, reducing high-frequency noise and diurnal fluctuations (e.g., segmentation inconsistencies or leaf reorientation due to changes in light) that do not reflect actual biomass accumulation.

To maintain the integrity of the morphological signal, a grouped outlier correction procedure was applied before aggregation. Specifically, an Interquartile Range (IQR) filter was calculated independently for each day. This localized filtering ensures that transient artifacts, such as sudden changes in illumination or segmentation errors, are identified relative to the plant's developmental state on that day. For each day d , observations falling outside this range were Winsorized, capped to the nearest non-outlier boundary to suppress structural noise while preserving the monotonicity of the growth trajectory. This approach suppresses structural noise while maintaining the monotonicity of the growth trajectory.

²<https://www.tecnosnet.com/wmc/Modello.asp>

$$\text{Range}_d = [Q_{1,d} - 1.5 \cdot \text{IQR}_d, Q_{3,d} + 1.5 \cdot \text{IQR}_d] \quad (6.5)$$

The daily mean feature value $\bar{x}_d$ was then computed as:

$$\bar{x}_d = \frac{1}{n_d} \sum_{i=1}^{n_d} x_{i,d} \quad (6.6)$$

where $x_{i,d}$ denotes the i -th valid measurement on day d , and n_d is the total number of successful acquisitions for that day. This procedure establishes a consistent temporal baseline to synchronize image-derived features with manual destructive biomass sampling.

2. **Biomass Interpolation:** Manual biomass measurements were collected as ground-truth calibration points at approximately 7-day intervals, with minor variations across the experimental campaign. These discrete measurements were transformed into a continuous daily signal, $w(t)$, using piecewise linear interpolation. While biological growth typically follows non-linear trajectories, a linear approximation was adopted for these short intra-sampling intervals to provide a deterministic estimate of daily accumulation. This approach avoids the overfitting risks associated with higher-order models while ensuring temporal alignment with the daily-aggregated morphological features.

The finalized dataset provides a synchronized system state record. Each entry (t_d) consists of a feature vector $\mathbf{X}_d$ containing depth-corrected morphological traits, input environmental variables with their 24h/72h rolling averages, and the interpolated daily biomass $w(t_d)$, forming the multi-dimensional basis for the DT's predictive engine. The full dataset, including RGB imagery, morphological traits, and water-quality measurements, is publicly available on Zenodo [197].

6.4.6 Mechanistic Modeling of Plant Growth Dynamics

As previously noted, plant phenotyping offers quantitative measurements of structure, development, and physiology that support the analysis of plant growth. These measurements are crucial for modeling frameworks, whether they are mechanistic or data-driven.

High-end phenotyping systems such as LiDAR scanners [198], multi-view stereo rigs [199], hyperspectral cameras [200], and commercial 3D phenotyping platforms [201] offer detailed structural and physiological information but remain prohibitively expensive and operationally complex. Consequently, their adoption is limited in many research and commercial agricultural settings.

In this context, the study investigates whether low-cost 2D camera modules can effectively capture biologically relevant growth indicators in *Salicornia*. We analyze growth dynamics using images acquired with simple imaging hardware and assess how well these measurements approximate the information typically obtained from advanced phenotyping systems and 3D reconstruction methods. Demonstrating the viability of low-cost imaging would substantially lower the barriers to scalable phenotyping and support broader deployment in practical agricultural applications.

Building on the conceptual background introduced in Section 6.2, this section focuses on how nonlinear growth models are applied to image-derived structural features to characterize *Salicornia* development. Rather than revisiting the full mathematical derivations, which are provided in Appendix A, the emphasis here is on the biological rationale and modeling strategy.

The nonlinear growth models utilized in this work were fitted using the entire growth sequence of each plant, from transplanting to harvest. However, the parameters derived from these models, along with their secondary features, such as instantaneous growth rate or acceleration, incorporate information about

later developmental stages. Therefore, these model-based descriptors are primarily used for retrospective phenotyping, in which the aim is to understand how characteristic growth patterns correlate with biomass accumulation once the full growth trajectory is known. This retrospective analysis establishes an upper limit on the accuracy achievable when the full phenological context is available.

Additionally, this thesis includes complementary experiments designed to assess real-time *Salicornia* biomass estimation. In these experiments, predictions are made using only the information available up to a specific day of growth. These real-time scenarios reflect practical deployment conditions and enable comparison between the theoretical performance of full-cycle modeling and the constraints of on-the-fly prediction.

This section aims to clearly define the methodological foundation of the proposed approach.

6.4.6.1 Linearity vs. Nonlinearity in Biological Systems

Choosing between linear and nonlinear modeling is more than just a statistical preference; it reflects our understanding of how a system operates. A linear model assumes a constant rate of change and represents relationships as simple additive combinations of parameters. In contrast, nonlinear models are employed when growth or change cannot be adequately approximated by a straight-line relationship. These models capture more complex dynamics, interactions, and constraints that linear models cannot represent [202].

Linear regression, represented by the equation:

$$Y = \beta_0 + \beta_1 t + \epsilon \quad (6.7)$$

This formula assumes that the dependent variable Y (such as biomass) changes at a constant rate, denoted as β_1 , over time t . This interpretation is standard in linear biological models, where the slope indicates a constant rate of change. However, maintaining a constant Absolute Growth Rate (AGR) throughout an organism's life cycle is physiologically unrealistic. In reality, biological growth typically follows non-linear patterns rather than remaining fixed indefinitely [203]. Therefore, while linear models provide computational simplicity, they are fundamentally ill-suited for describing biological growth over extended periods. Linear regression assumes a constant rate of change, which implies infinite expansion and violates the system's physical and biological constraints.

Archontoulis and Miguez (2015) note that nonlinear models are more effective for agricultural research because they account for growth limitations and variations in plant efficiency across developmental stages [204]. In particular, as displayed in Figure 6.23, the characteristic "S-shaped" or sigmoidal curve represents the three critical phases of a plant's life cycle [205]:

- **Lag phase (Initial growth):** slow growth due to establishment of roots and early leaf development.
- **Log phase (Rapid growth):** accelerated biomass accumulation and canopy expansion.
- **Stationary phase (Maturity):** growth slows as the plant reaches physiological maturity or resource limitations.

This sigmoidal growth pattern has been observed in a wide variety of plant species and organ types, including leaves, stems, fruits, and roots. This growth trajectory arises from fundamental biological processes such as cell division, cell elongation, and tissue differentiation. These processes are also influenced by environmental factors such as light availability, temperature, and nutrient availability. Together, these internal and external factors drive the transition from slow initial growth to rapid expansion and, eventually, to physiological saturation.

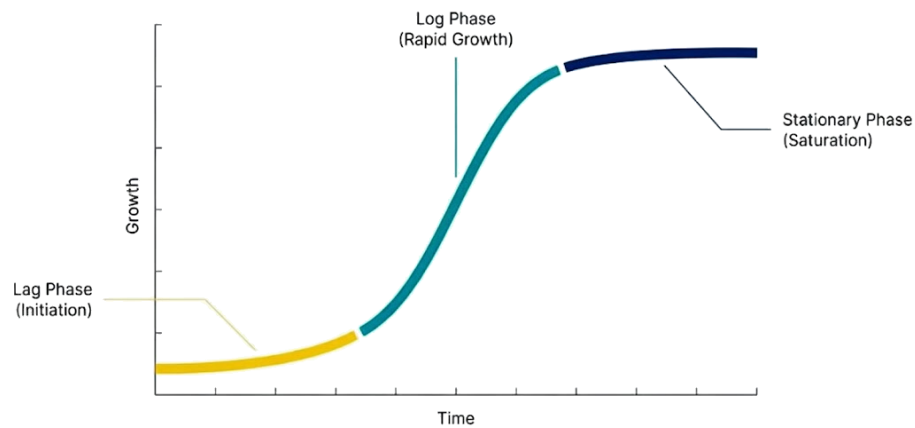


Figure 6.23: Sigmoid plant growth curve showing lag, log, and stationary growth phases.

Nonlinear growth models offer a clear and interpretable framework for quantifying these dynamics. These models enable the direct estimation of key parameters that define the timeline of a biological process: the asymptote, which represents the maximum size achievable; the intrinsic growth rate, which determines the steepness of the exponential growth phase; and the inflection point, the moment at which growth shifts from acceleration to deceleration. These parameters are biologically significant, as they correlate with measurable physiological realities. For example, differences in the asymptotic parameter across treatments may indicate changes in resource availability or developmental constraints, while variations in the growth rate parameter can reflect differences in metabolic efficiency or responsiveness to environmental conditions [206].

6.4.6.2 Candidate Non-Linear Models: Mathematical Formulations and Biological Meanings

As previously discussed, determinate plant growth generally follows a sigmoidal pattern. To accurately capture this growth trajectory, it is crucial to select an appropriate nonlinear growth function from the broader family of sigmoidal models. While these functions all produce S-shaped curves, they vary significantly in terms of symmetry, inflection point location, and parameter complexity.

From a computational perspective, it is possible to evaluate many candidate functions and select the one that minimizes the fitting error. However, as noted by Paine et al. (2012) [207], such data-driven model selection can lead to overfitting and diminished biological interpretability. To mitigate these issues, we limited our analysis to models that:

- Has been successfully validated in similar phenotyping domains.
- Possess parameters with distinct biological or physical meanings (e.g., maximum potential, growth velocity)

Following the work of Li et al. (2022) [154] and relevant biometric literature, the models considered in this study include the Logistic, Gompertz, and Grey Verhulst growth functions. Each captures a different aspect of plant growth:

- **Logistic Model:** A symmetric sigmoidal growth function where growth accelerates until the inflection point at $K/2$ and then decelerates as the system approaches its carrying capacity. Commonly used for bounded biological growth such as biomass, leaf area, or plant height. Its practical relevance has been demonstrated in several phenotyping studies. For example, Li et al. (2022) reported that the Logistic model yielded the most accurate representation of plant height dynamics.

- **Gompertz Model:** An asymmetric sigmoidal function characterized by slow initial growth, rapid early expansion, and an early inflection point at K/e , where e is Euler's number, approximately 2.718. Suitable for species in which early vegetative growth dominates and maturation proceeds gradually. This early inflection indicates that the Gompertz model is well-suited to describing biological systems in which rapid growth takes place during the initial stages of development, followed by a gradual approach to maturity. Consequently, it is frequently used to model biomass accumulation or volumetric growth in crops such as maize or tomato, where early vegetative growth is pronounced while later maturation progresses more slowly [208].
- **Grey Verhulst Model:** A growth model derived from Grey System Theory, designed for noisy or incomplete datasets. Uses an Accumulated Generating Operation (AGO) to smooth fluctuations before fitting a sigmoidal curve. Li et al. (2022) demonstrated that the Grey Verhulst model achieved the best predictive performance for estimating top projected canopy area and canopy perimeter.

The mathematical formulations and parameter interpretations for these models are provided in Appendix A.

6.4.6.3 Model Evaluation and Selection Criteria

To determine the optimal nonlinear growth model, we employed a two-stage selection process. First, models were evaluated based on their “statistical goodness-of-fit” to the observed growth data, ensuring numerical accuracy and convergence. Second, the models were assessed for their “biological representativeness”, specifically by examining the correlation between the “critical points” (developmental milestones) derived from morphological features and actual plant weight. This dual approach ensures that the selected model is not only mathematically precise but also biologically synchronized with the underlying biomass accumulation.

6.4.6.3.1 Stage 1: Model Fitting and Statistical Convergence Candidate nonlinear models were fitted using nonlinear least squares with biologically informed initialization and bounded parameter constraints. These procedures ensured stable convergence even with left- or right-truncated data, which are common in dense plant populations. The full optimization workflow is detailed in Appendix B.

Rather than prioritizing a single metric, the comparison of the Logistic, Gompertz, and Grey Verhulst models was based on a combined interpretation of RMSE, Coefficient of Determination (R^2), and MAPE introduced in Section 2.4. This integrated evaluation ensures that the selected model achieves strong absolute accuracy, explains the underlying variance, and maintains stable relative performance across the full growth trajectory.

6.4.6.3.2 Stage 2: Biological Validation via Critical Point Analysis While statistical metrics confirm numerical accuracy, they do not guarantee that a model captures the underlying biological transitions of plant development. In this second stage of selection, we evaluate the models based on their ability to synchronize image-derived morphological proxies (“Area” and “Convex Hull”) with destructive manual measurements (“Weight”). The selection is based on the alignment of “critical points”, specific timestamps extracted from the model's derivatives that represent physiological milestones.

Establishing temporal synchronization between morphological and biomass dynamics is essential for the DT framework. If image-derived dynamic features align with physiological growth stages, they can be used as biologically meaningful indicators of metabolic vigor.

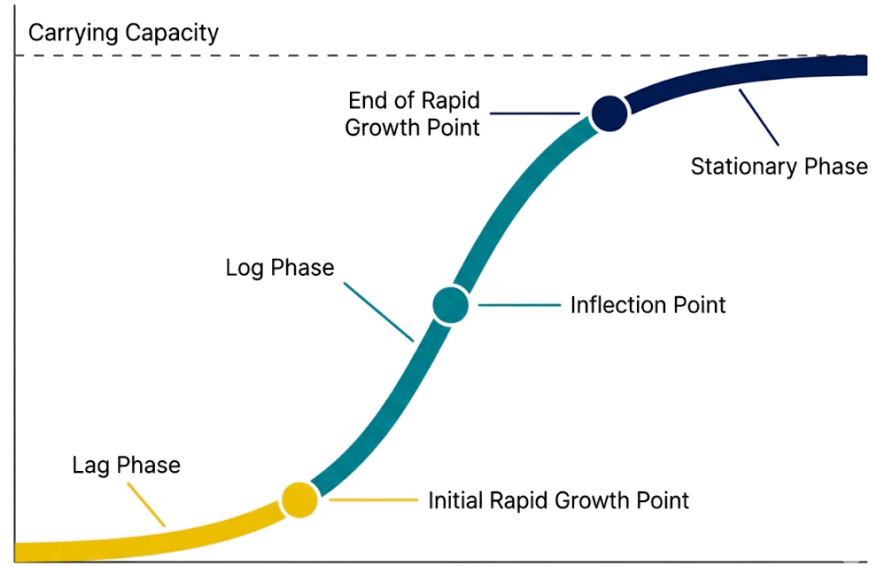


Figure 6.24: Illustration of critical growth points (inflection and rapid-growth interval) of sigmoidal models.

6.4.6.3.3 Critical Point Extraction After fitting nonlinear models to data, the following analytical features were computed from the second and third derivatives of the fitted functions:

- **Inflection Point (t_i):** The time at which the second derivative equals zero, corresponding to the maximum growth rate.
- **Rapid Growth Stage (t^S – t^E):** The interval between the roots of the third derivative, marking the onset (t^S) and termination (t^E) of the explosive growth phase.
- **Cross-Feature Correlation:** Pearson correlation (r) between the first derivatives of morphological traits and Weight, quantifying the temporal alignment of growth dynamics.

Figure 6.25 displays critical points computed from the fitted Logistic model on the weight data of the middle plant of the pot, which was captured by camera 5.

A candidate model is considered superior if it exhibits strong temporal correlation among features. This ensures that the DT can reliably use non-destructive visual data to predict the physiological behavior of the plant.

6.4.6.4 Extraction of Model-Based Features

Once the optimal growth function $f(t)$ is identified, the continuous nature of the function allows for the derivation of features that are not directly observable in static images. These features transform the analysis from a static assessment of size to a dynamic assessment of physiological behavior. We extracted four key features for every time point t :

- **Model-Predicted Value ($\hat{y}(t)$):** The estimated value of the size variable (Area or Convex Hull) at time t obtained from the fitted growth model. In contrast to the raw observation $y_{\text{obs}}(t)$, which may exhibit high-frequency fluctuations caused by leaf motion, illumination changes, or segmentation noise, $\hat{y}(t)$ reflects a smoothed and biologically coherent representation of the plant's structural state.

$$\hat{y}(t) = f(t, \hat{\theta}), \quad (6.8)$$

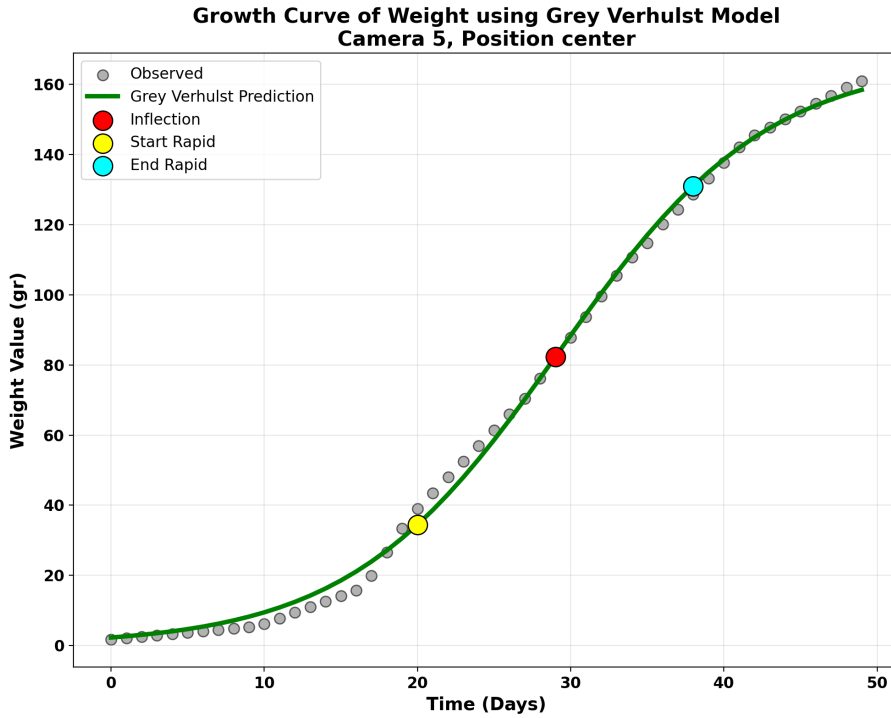


Figure 6.25: Critical points estimated from the fitted logistic growth model for the weight data of a sample plant.

where $\hat{\theta}$ denotes the optimized model parameters (e.g., K , r , t_0). This feature serves as a stable magnitude descriptor and forms the basis for deriving higher-order growth characteristics.

- **Instantaneous Growth Rate (Velocity):** Defined as the first derivative of the fitted growth curve with respect to time, this metric quantifies the absolute rate at which the selected morphological feature (e.g., Area or Convex Hull) increases at a given moment:

$$v(t) = \frac{d\hat{y}}{dt}. \quad (6.9)$$

Velocity reflects the plant's metabolic activity and structural expansion. High values correspond to the active vegetative phase, whereas values approaching zero indicate either early establishment (lag phase) or the onset of maturity and growth cessation.

- **Acceleration:** Defined as the second derivative of the fitted curve, acceleration describes the *change* in growth rate over time:

$$a(t) = \frac{d^2\hat{y}}{dt^2}. \quad (6.10)$$

This metric distinguishes whether the plant is speeding up or slowing down in its growth dynamics:

- $a(t) > 0$: Growth is accelerating, characteristic of the exponential (log) phase.
- $a(t) = 0$: The plant reaches its inflection point, corresponding to maximum growth velocity.
- $a(t) < 0$: Growth is decelerating as the plant approaches physiological limits or environmental constraints.

- **Relative Saturation Index (Fraction of Max):** A normalized indicator of how close the plant is to its theoretical maximum size. It expresses the proportion of the carrying capacity that has been reached at time t :

$$S_{rel}(t) = \frac{\hat{y}(t)}{K_{bounded}}, \quad (6.11)$$

where K is the estimated carrying capacity. This metric distinguishes between plants that are genuinely small because they are early in development ($S(t) \approx 0.1$) and plants that are small because growth has plateaued due to stress or maturity ($S(t) \approx 0.9$). Values approaching 1 indicate that the growth process is nearing completion. Because K is estimated from right-truncated data, $S_{rel}(t)$ represents the mathematical progress toward the predicted asymptote rather than a confirmed physiological limit.

- **Residual:** The difference between the observed measurement and the model-predicted value at time t :

$$\text{Residual}(t) = y_{\text{obs}}(t) - \hat{y}(t).$$

Positive residuals indicate that the plant is larger than predicted, while negative residuals indicate underperformance relative to the modeled trajectory. This feature captures deviations caused by biological variability, measurement noise, or transient environmental effects.

6.4.7 Integration with Machine Learning

The primary motivation for this extensive curve-fitting exercise is not merely description, but “feature engineering” for predictive modeling. Raw morphological features (e.g., Area, Width) provide a static snapshot of the plant’s dimensions. By calculating the derivatives (Velocity and Acceleration) of the growth curve, we generate “Hybrid Features” that provide the machine learning model with temporal context. These features enable regression algorithms (discussed in Section 6.5) to improve the accuracy of weight estimation.

6.5 Experimental Results

The primary objective of the experimental phase was not only to estimate *Salicornia*’s biomass but also to develop a modeling framework that captures and interprets its underlying biological growth dynamics. Instead of viewing each observation as an isolated measurement, the objective of experiments is to reveal how the plant evolves over time, how its structural features accelerate, decelerate, and ultimately reach a saturation point as it progresses through distinct developmental stages. Table 6.4 describes the list of features set used in the following experiment, containing structural (physical measurements), Environmental (sensor data), and model-based (derived growth metrics), along with concise descriptions.

By integrating nonlinear growth models with ML methods, the DT goes beyond simple curve-fitting or snapshot-based predictions. The growth models offer a physiologically grounded representation of the plant’s trajectory, while the ML component utilizes these model-derived features to infer biomass and future states from partial or noisy observations. This combination allows the system to function in both retrospective and real-time scenarios, using early-stage measurements to anticipate later outcomes.

In this study, the DT is defined as a virtual representation of the *Salicornia* plant that maintains continuous synchronization between physical observations and mathematically inferred growth states. Unlike a conventional predictive model, the twin integrates multi-modal data streams, combining real-time environmental measurements (such as salinity and pH) with non-destructive visual traits. Anchored in a physiologically grounded, nonlinear growth framework, this system establishes a dynamic feedback loop that estimates not only current biomass but also the plant’s metabolic trajectory and overall biological condition relative to its developmental stage.

Table 6.4: Feature categorization for *Salicornia* growth analysis. [Var] denotes each environmental variable (Oxygen, Temperature, SAT, pH, Salinity).

Category	Feature Name	Description
Structural	branch_count, tips_count	Morphological complexity indicators extracted via skeletonization.
	path_lengths	Total cumulative length of the plant skeletal structure.
	width, height	Bounding box dimensions of the segmented plant.
	convexHull	Area of the minimum convex polygon enclosing the plant.
	solidity, aspect_ratio	Shape descriptors capturing density and geometric proportions.
Environmental	initial_weight	Initial biomass used as a baseline for growth modeling.
	Load_[Var]	Raw sensor measurements for each environmental variable.
	Load_[Var]_MA_72h	72-hour moving average representing long-term environmental trends.
	Delta_[Var]	Daily variation of each variable.
	Delta_[Var]_MA_72h	Smoothed rate of change over a 72-hour window.
Model-Based	days_since_start	Elapsed time since the beginning of the experiment.
	area_predicted	Area estimated by the fitted non-linear growth model.
	area_growthRate	First derivative of area, representing instantaneous growth rate.
	area_acceleration	Second derivative of area, describing changes in growth dynamics.
	area_fractionOfMax	Ratio between current area and maximum observed or expected area.
	area_residual	Difference between observed area and model-predicted area.

6.5.1 Model Fitting and Numerical Evaluation of Candidate Growth Models

As described in Section 6.4.6.3, the initial step of model evaluation focused on assessing the numerical performance and convergence stability of the three candidate nonlinear functions across the longitudinal datasets. In this experiment, the Logistic, Gompertz, and Grey Verhulst sigmoidal models were fitted to the daily aggregated measurements obtained from Phase 2, as summarized in Table 6.3. Phase 2 was selected because it provides a near-complete representation of the full growth cycle, encompassing the initial lag phase, the period of rapid expansion, and the final saturation stage. This extended temporal coverage enables reliable parameter estimation and supports the extraction of features that meaningfully reflect the biological processes underlying growth.

Results for the first stage of model selection, concerning fitting precision and error distribution (RMSE, R^2 , and MAPE), are summarized in Table 6.5.

Across the Logistic and Gompertz models, differences in RMSE were relatively small, and both achieved consistently high explanatory power ($R^2 > 0.95$ for all three features), with the Logistic model producing marginally lower RMSE and higher R^2 values throughout, particularly for the weight feature, where it achieved a near-perfect fit ($R^2 = 0.99$). This strong performance indicates that sigmoidal growth behavior is an intrinsic characteristic of the experimental plants, and that is, continuous closed-form sigmoidal functions can reliably capture the dominant growth dynamics of *Salicornia*.

The Grey Verhulst model, by contrast, performed substantially worse across all three features, with RMSE values an order of magnitude higher than those of the Logistic and Gompertz models and negative R^2

Table 6.5: Numerical comparison of the Logistic, Gompertz, and Grey Verhulst growth models fitted to Convex Hull Area, Area, and Weight.

Feature	Model	RMSE	R ²	MAPE
Convex Hull Area	Logistic	1793.66	0.97	0.19
	Gompertz	2016.59	0.96	0.19
	Verhulst	46840.28	-0.93	0.75
Area	Logistic	1402.24	0.97	0.22
	Gompertz	1645.22	0.95	0.21
	Verhulst	37115.27	-0.98	0.78
Weight	Logistic	3.28	0.99	0.18
	Gompertz	4.973	0.98	0.29
	Verhulst	63.64	-0.57	0.83

values for Convex Hull Area and Area, indicating a fit poorer than the sample mean. This underperformance is consistent with the discrete, accumulation-based nature of the GM(1,1) formulation: unlike the continuous sigmoidal models, the Grey Verhulst reconstruction depends on differencing an accumulated (AGO) series, which can amplify estimation error when projected back into the original data space, particularly for series with the sampling density and noise characteristics of our dataset. On this basis, the Grey Verhulst model was excluded from the second stage of model selection (biological alignment) and from subsequent hybrid feature extraction.

6.5.2 Validation of Morphological Proxies via Critical Point Analysis

As discussed in Section 6.4.6.3, beyond statistical fitting, the second stage of the results evaluates the biological validity of the candidate models by analyzing the temporal synchronization between morphological critical points and actual biomass accumulation. While the initial selection (Table 6.5) indicated that the Logistic and Gompertz models provide high numerical accuracy, this follow-up validation reveals critical differences in their predictive stability.

Table 6.6 summarizes the mean $\pm$ SEM delays and the correlations between the growth rates of image-derived traits and Weight, computed by fitting the Logistic and Gompertz models on Area, Convex Hull Area, and Weight across plants of Phase 2. As shown in the table, the Logistic model demonstrates stronger temporal consistency, particularly in the alignment of inflection points. Moreover, Area emerges as a more reliable proxy for Weight than Convex Hull Area: it exhibits a substantially shorter Mean Inflection Delay (2.75 ± 0.70 vs. 5.00 ± 1.30 days) and higher reliability (89% vs. 78%). These results indicate that projected 2D surface area captures biomass accumulation more precisely than canopy boundary extent, reinforcing the suitability of the Logistic model applied to Area as the mechanistic backbone of the DT.

Table 6.7 reports the detailed growth milestones extracted for Phase 2 plants using the selected Logistic model. Furthermore, Figure 6.26 illustrates the model's performance by showing a sample of the logistic fit to the middle plant data captured by Camera 5.

6.5.3 Experimental Setup for Biomass Estimation

A central purpose of this study is to convert morphological and environmental telemetry into reliable biomass estimates. This section describes the ML architecture and the validation procedures that support robust generalization across biological samples.

Table 6.6: Synchronization performance between image-derived proxies and plant weight. Metrics report mean $\pm$ SEM delays of critical growth milestones and the correlation between the growth rates of morphological features and Weight across plants of Phase 2. Reliability quantifies the proportion of plants whose fitted proxy-weight dynamics achieve $r > 0.85$.

Logistic Model (Selected)		
Synchronization Metric	Area vs. Weight	Convex Hull vs. Weight
Inflection Delay (Days)	2.75 ± 0.70	5.00 ± 1.30
Rapid Growth Start Delay (Days)	6.03 ± 1.50	8.06 ± 1.72
Rapid Growth End Delay (Days)	1.15 ± 0.57	1.69 ± 0.62
Mean Correlation (r)	0.93	0.82
Reliability ($r > 0.85$)	89%	78%
Gompertz Model		
Metric	Area vs. weightt	Convex Hull vs. weightt
Inflection Delay (Days)	3.88 ± 2.10	7.25 ± 2.80
Rapid Growth Start Delay	7.62 ± 1.77	9.35 ± 1.89
Rapid Growth End Delay	1.86 ± 0.92	2.72 ± 0.83
Mean Correlation (r)	0.90	0.77
Reliability ($r > 0.85$)	77%	67%

6.5.3.0.1 Model Development and Optimization The estimation engine is built upon a Gradient Boosting Regressor (GBR) [209], an ensemble learning technique chosen for its efficacy in modeling the complex, non-linear dependencies inherent in biological systems. GBR is particularly well-suited for agricultural datasets, which often exhibit high dimensionality and noise [210, 211].

6.5.3.0.2 Hyperparameter Optimization via Grid Search To identify the optimal configuration and prevent overfitting on a relatively small dataset, a systematic “Grid Search” was performed. Hyperparameter optimization via Grid Search is a standard approach for improving generalization in such models [210]. Table 6.8 shows the search space designed to prioritize strong regularization and stochastic boosting.

The resulting optimal parameters are detailed in Table 6.9.

6.5.3.0.3 Robust Optimization via Huber Loss To optimize the model’s objective function, the “Huber loss” ([209]) was selected over standard MSE. The loss function is defined as:

$$L_{\delta}(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{for } |y - f(x)| \leq \delta, \\ \delta(|y - f(x)| - \frac{1}{2}\delta) & \text{otherwise.} \end{cases} \quad (6.12)$$

By transitioning from a quadratic penalty to a linear penalty at the threshold δ , the model maintains sensitivity to standard biomass fluctuations while remaining robust against heavy-tailed noise, such as sensor calibration drifts or manual data entry errors.

6.5.3.0.4 Data Splitting and Validation: Group K-Fold A critical challenge in plant growth modeling is spatial data leakage, in which data from the same plant appear in both the training and test sets. To prevent this, a “Group K-Fold” cross-validation strategy was implemented.

Mechanism: In each fold, all data points belonging to a single unique “plant-id” are removed from the training set and reserved exclusively for testing.

Unseen Data Testing: This ensures that the model’s performance metrics reflect its ability to estimate the weight of an entirely “unseen” plant, simulating a real-world application where the DT must monitor a new biological sample.

Table 6.7: Extracted biological milestones for Phase 2 plants using the Logistic growth model.

Plant ID	Feature	Inflection (t^*)	Start Rapid (t^S)	End Rapid (t^E)	Correlation (r)
1_right	Area	58.00	33.17	65.79	0.98
	ConvexHull	57.00	32.08	65.72	0.98
	Weight	53.00	37.71	65.25	1.00
1_center	Area	51.00	37.99	56.91	0.98
	ConvexHull	46.00	34.55	56.31	0.90
	Weight	53.00	42.72	57.17	1.00
2_center	Area	49.00	30.44	56.76	0.96
	ConvexHull	42.00	26.30	56.08	0.83
	Weight	49.00	38.36	56.59	1.00
3_right	Area	57.00	30.35	70.48	0.98
	ConvexHull	53.00	28.98	70.30	0.97
	Weight	56.00	39.94	70.19	1.00
4_right	Area	28.00	13.57	35.71	0.98
	ConvexHull	27.00	10.06	35.66	0.96
	Weight	32.00	13.02	36.28	1.00
4_center	Area	29.00	13.01	31.59	0.99
	ConvexHull	28.00	11.50	31.38	0.99
	Weight	29.00	17.27	31.52	1.00
4_left	Area	29.00	9.19	41.40	0.85
	ConvexHull	22.00	5.76	37.44	0.12
	Weight	34.00	23.47	41.61	1.00
5_right	Area	29.00	18.08	39.76	0.83
	ConvexHull	31.00	18.44	41.22	0.85
	Weight	32.00	27.01	36.75	1.00
5_center	Area	26.00	17.09	33.94	0.89
	ConvexHull	26.00	16.52	36.02	0.93
	Weight	29.00	20.13	38.28	1.00

6.5.4 Feature Ablation Study

To quantify the contribution of different data modalities to biomass estimation, a feature ablation study was conducted. The experiment was repeated under three distinct configurations:

1. **Full Model (Comprehensive DT):** Includes all available features—raw morphological traits, environmental telemetry (e.g., salinity, pH, temperature), and model-derived descriptors (velocity, acceleration, saturation index) extracted from Logistic growth curves fitted to Area.
2. **Without Model-Based Features:** Excludes features derived from fitted growth curves. This tests whether the temporal context provided by curve-based descriptors improves prediction beyond static daily snapshots.
3. **Without Environmental Data:** Removes sensor-derived features to assess the influence of abiotic stressors on biomass accumulation and tissue density.

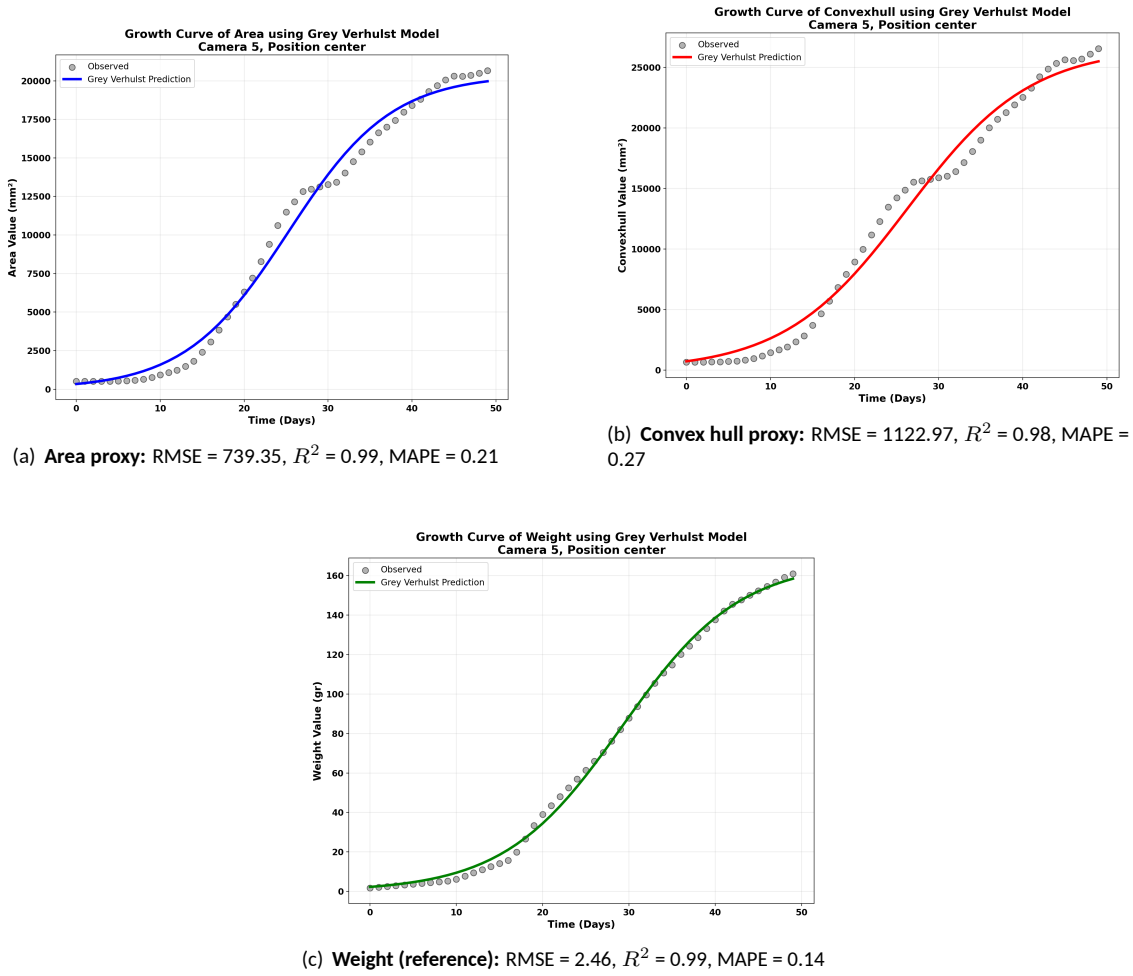


Figure 6.26: Logistic growth curve fitting for the central plant observed by Camera 5. The top row shows fits for the Area and Convex Hull proxies, and the bottom plot shows the reference Weight fit. Goodness-of-fit metrics are reported with each curve.

Table 6.8: Hyperparameter Search Space for Grid Search

Hyperparameter	Search Values
$n_estimators$	[100, 200, 300, 500]
Learning Rate (η)	[0.01, 0.05, 0.1]
max_depth	[2, 3]
$min_samples_leaf$	[5, 10, 20]
$min_samples_split$	[10, 20]
Subsample	[0.6, 0.8]
Loss Function	[Squared Error, Huber]

Table 6.9: Optimal Gradient Boosting Parameters

Hyperparameter	Selected Value	Objective
Loss Function	Huber	Robustness to measurement outliers
Learning Rate (η)	0.1	Iterative optimization speed
$n_estimators$	200	Number of sequential trees
max_depth	3	Control of interaction complexity
$min_samples_leaf$	5	Minimum samples for a leaf node
$min_samples_split$	10	Minimum samples for a split
Subsample	0.6	Stochastic variance reduction
$max_features$	sqrt	Feature decorrelation

The ablation study results (Table 6.10) demonstrate that the inclusion of model-derived features consistently yielded the lowest prediction errors. The Full Model achieved an RMSE of 20.35 g, representing a 12.9% improvement over the configuration excluding model-based descriptors (23.37 g). While the standard errors (SEM) reflect the inherent relatively small sample size of plants, the performance gain observed in the Full Model was consistent across cross-validation folds. This suggests that the temporal context provided by the Logistic growth curves, specifically velocity and acceleration, enables the Gradient Boosting Regressor to better differentiate between plants of similar size but different metabolic states. Removing environmental telemetry resulted in a moderate increase in error (21.73 g), indicating that abiotic factors serve as secondary modulators of biomass. At the same time, morphological expansion and its underlying dynamics remain the primary predictors.

The Full Model maintained the lowest MAE (15.24 g), demonstrating strong generalization to unseen individuals and validating the robustness of the DT framework.

Table 6.10: Ablation study results for *Salicornia* biomass estimation using Gradient Boosting. Metrics are reported as Mean $\pm$ Standard Error (SEM) across 7 folds using Group K-Fold cross-validation.

Feature Configuration	RMSE (g)	MAE (g)
Full Model (Morphological + Env. + Model-Based)	20.35 $\pm$ 4.70	15.24 $\pm$ 3.81
Without Environmental Data	21.73 $\pm$ 4.03	16.23 $\pm$ 3.04
Without Model-Based Features	23.37 $\pm$ 5.15	18.18 $\pm$ 4.14

To illustrate model behavior, Figure 6.27 compares prediction trends for two representative plants using the Full Model. The left column shows a high-fidelity case with minimal error and strong alignment between predicted and observed biomass. The right column shows a moderate case with increased dispersion.

To interpret model behavior, SHAP analysis was performed. Figure 6.28 shows the mean absolute SHAP values.

The SHAP analysis confirms the importance of temporal and structural features. The top contributors include:

- **path_lengths** and **days_since_start**: These encode developmental age and structural complexity, acting as proxies for cumulative growth.
- **area_fractionOfmax** and **area_predicted**: These model-derived features reflect saturation status and fitted biomass trajectory.
- **velocity** and **acceleration** (embedded in **area_growthRate** and **area_acceleration**): These capture dynamic growth behavior and metabolic vigor.

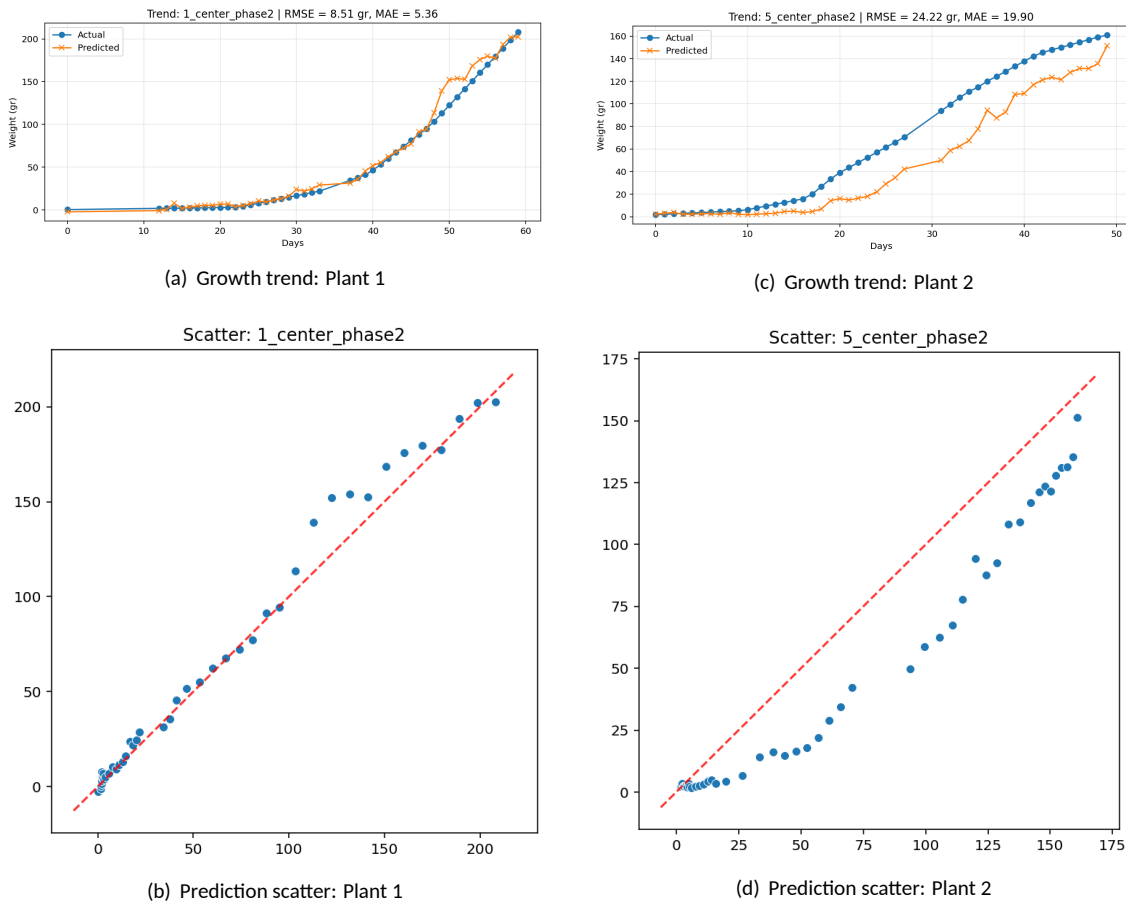


Figure 6.27: Comparison of prediction fidelity for two representative plants using the Full Model. Left: high-fidelity case ($RMSE = 8.51$ g, $MAE = 5.36$ g) with strong temporal alignment. Right: moderate case ($RMSE = 24.22$ g, $MAE = 19.90$ g) showing increased scatter and trend deviation.

- **environmental features** (e.g., Load_Salinity_ppm, Delta_SAT): Though lower ranked, they modulate tissue density and explain residual variance.

The SHAP summary plot reveals that high values of path_lengths and area_fractionOfmax consistently push predictions upward, confirming their role in identifying mature, high-biomass plants. Conversely, low values of convexHull and solidity are associated with underprediction, likely reflecting sparse or irregular canopy structures. In addition, while days_since_start, which represents time, is one of the strongest predictors, the model-based features (velocity/acceleration) provide the biological nuance that time alone cannot capture (e.g., detecting whether a plant is underperforming for its age).

Illustration of Prediction Differences Across Ablation Scenarios To illustrate the effect of feature exclusion, Figure 6.29 compares the predicted biomass for the same plant across all three model configurations. The Full Model yields the lowest error, while exclusion of model-based features leads to the largest deviation.

SHAP Analysis: Model-Based vs. Model-Free Configurations To further investigate the role of model-derived features, SHAP importance plots were generated for two configurations: one excluding environmental data but retaining model-based features (left), and one excluding both model-based and environmental features (right). These results are shown in Figure 6.30.

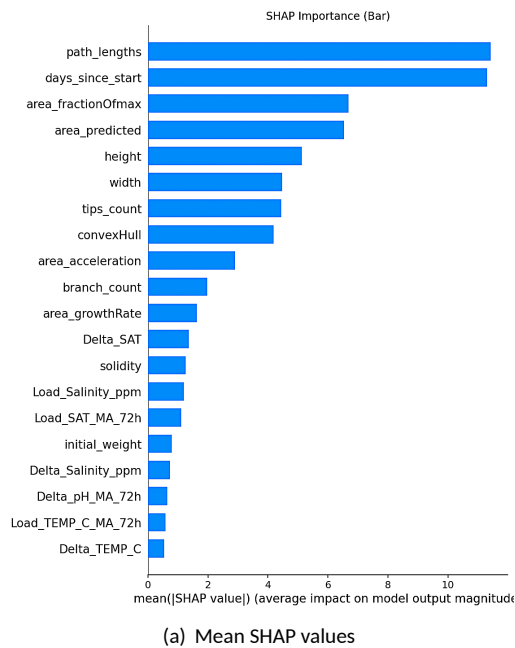


Figure 6.28: SHAP analysis of feature importance, Mean absolute SHAP values indicate average impact on model output

The SHAP analysis reveals a clear shift in feature importance when model-based descriptors are excluded. In the model-based configuration (Figure 6.30(a)), features such as `area_predicted`, `area_acceleration`, and `area_fractionOfmax` rank among the top contributors. These features encode temporal dynamics and saturation status, offering a biologically grounded representation of growth progression. Their prominence confirms that fitted curve features enhance the model's ability to interpret developmental stage and biomass accumulation.

In contrast, the model-free configuration (Figure 6.30(b)) shows increased reliance on static morphological traits such as `height`, `width`, and `aspect_ratio`. While these features capture geometric structure, they lack temporal context and are more susceptible to noise from leaf orientation and segmentation variability. The reduced predictive power in this configuration aligns with the ablation results, which showed significant increases in RMSE and MAE.

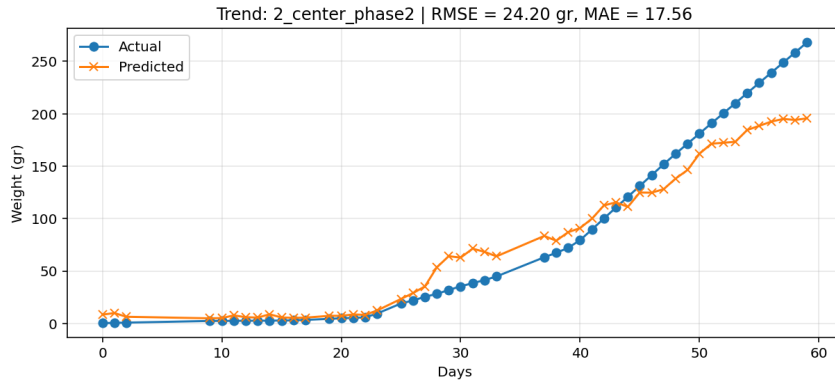
Notably, `path_lengths` and `days_since_start` remain influential across both configurations, suggesting that structural complexity and developmental age are robust predictors of biomass regardless of modeling approach.

These findings reinforce the value of integrating curve-derived features into the DT framework. By embedding biological memory and growth dynamics, the model gains interpretability and resilience, especially under conditions of limited environmental data.

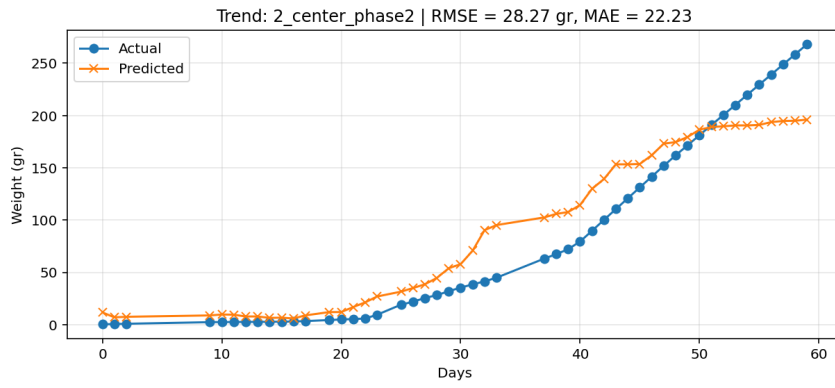
6.5.4.1 Results on Aggregated Data

To evaluate the DT's generalization across broader biological and temporal variability, data from Phases 1 and 2 were aggregated into a unified dataset. This combined set includes 11 features, 400 observation days, and 15 plant samples.

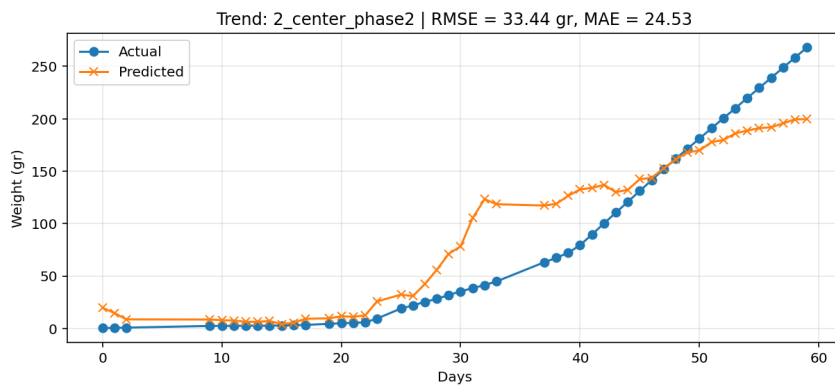
Using 5-fold Group K-Fold cross-validation, the model achieved the performance reported in Table 6.11. As expected, the increased heterogeneity of early-stage plants, variable canopy structures, and differing environmental conditions resulted in higher error compared to Phase 2 alone. Nevertheless, the model maintains reasonable predictive accuracy, demonstrating robustness across diverse growth scenarios.



(a) Full Model



(b) No Environmental Data



(c) No Model-Based Features

Figure 6.29: Prediction comparison for the middle plant captured by Camera 2, across the three ablation scenarios: the full model, the model without environmental data, and the model without model-based features.

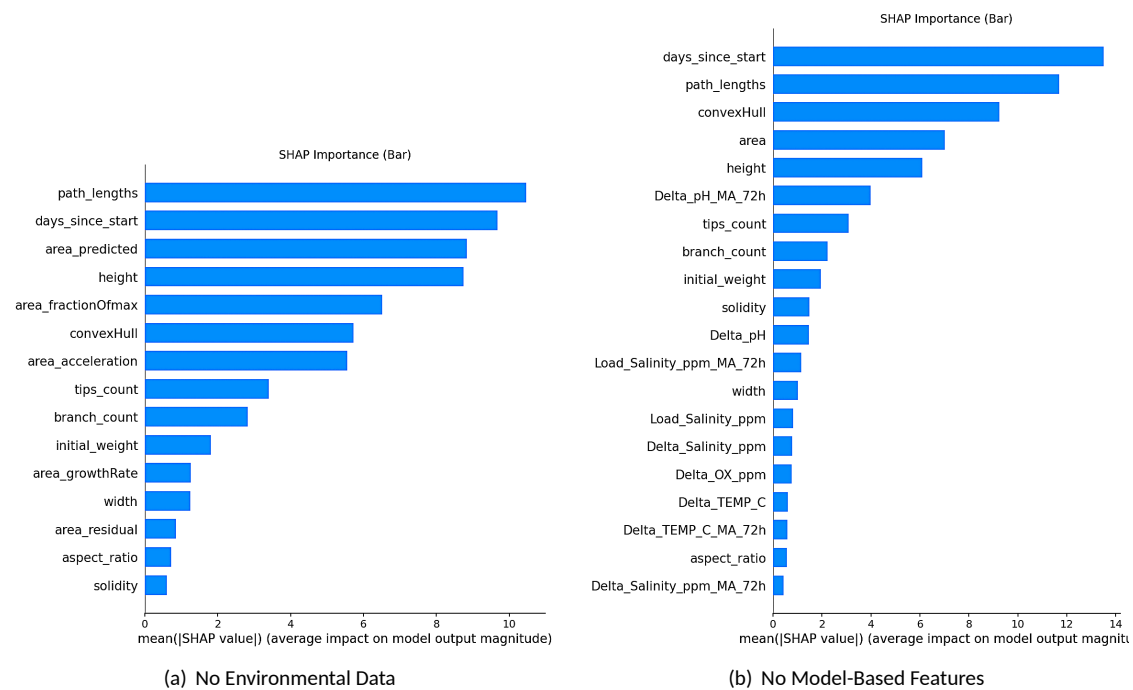


Figure 6.30: SHAP importance comparison under reduced feature configurations.

Table 6.11: Biomass estimation performance on aggregated Phase 1 + Phase 2 dataset using Group K-Fold cross-validation. Values represent Mean $\pm$ SEM across 5 folds.

Dataset	RMSE (g)	MAE (g)
Aggregated Phase 1 + Phase 2	30.47 ± 4.89	22.52 ± 3.75

Trend Visualization for Phase 1 Samples To illustrate model behavior on early-stage plants, Figure 6.31 presents prediction trends for two representative Phase 1 samples.

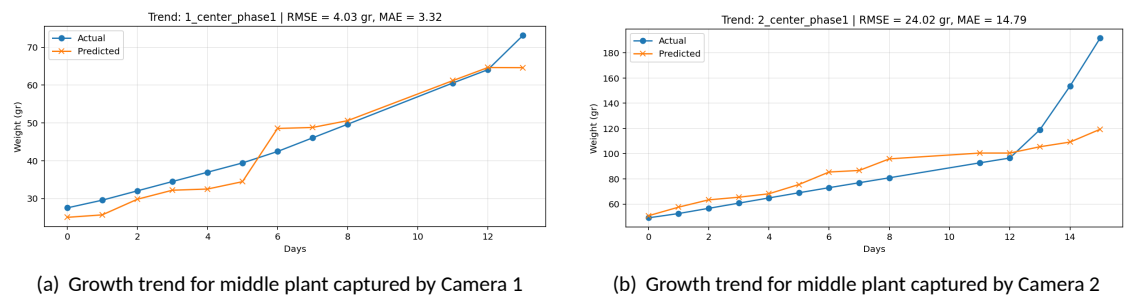


Figure 6.31: Predicted vs. observed biomass trends for two representative Phase 1 plants.

The aggregated dataset introduces substantial variability in plant size, canopy geometry, and developmental timing. Despite this, the model maintains stable performance. These results confirm that the DT framework generalizes beyond controlled Phase 2 conditions and remains effective across heterogeneous growth stages.

6.6 Conclusion

This chapter successfully establishes a robust DT framework for modeling the growth of *Salicornia* within the BeBlue aquaponic systems. By integrating model-based insights with data-driven methodologies, the framework provides a high-fidelity virtual representation of the plant's development.

The key achievements and methodological strengths of this study are as follows:

- **High-Resolution Monitoring:** The deployment of a multi-camera system based on low-cost ESP32-CAM modules enabled continuous and autonomous plant phenotyping with very high operational reliability. This approach addressed the limitations of manual data collection by providing a dense time series of images.
- **Precise Trait Extraction:** The image processing pipeline, using the free frameworks Pixellib and PlantCV, successfully transformed raw images into quantitative structural traits, including height, width, and convex hull area. The use of CIELAB color space inversion proved particularly effective for accurate vegetation segmentation under variable lighting conditions. By integrating these features derived from images captured during the plant's growing period, the dataset provides a rich spatiotemporal representation of plant development.
- **Multi-Source Data Integration:** The framework effectively combined image-derived structural features with real-time environmental covariates, including water temperature, pH, and salinity. This holistic approach ensured that the DT reflects both the plant's physical state and the external factors influencing its growth.
- **Superior Predictive Performance:** The application of a Gradient Boosting regressor demonstrated the power of the hybrid model-driven, data-driven approach. By incorporating model-derived growth descriptors alongside structural and environmental data, the system achieved high accuracy in biomass estimation. Furthermore, the model achieved a strong correlation with plant weight data via logistic curve fitting, particularly in critical point analysis, enabling a precise understanding of growth stages and physiological transitions.

This DT framework provides a scalable and cost-effective alternative to commercial phenotyping platforms such as Phenospex. Enabling real-time monitoring and intelligent decision support in aquaponics supports more efficient and sustainable closed-loop farming practices. Future work will focus on extending the framework to coordinate multiple autonomous agents operating collaboratively to provide comprehensive monitoring across large agricultural environments.

Chapter 7

Conclusion

Chapter contents

7.1	Conclusion	144
7.1.1	Overall Summary and Future Directions	144
7.1.2	Detailed Contributions	144

7.1 Conclusion

7.1.1 Overall Summary and Future Directions

This thesis explored how hybrid modeling, combining mechanistic knowledge with data-driven learning, can improve the construction of DTs for complex biological and environmental systems. Across three representative domains, greenhouse microclimates, aquaculture, and plant phenotyping, the work demonstrated that integrating structural knowledge, biological principles, and contextual information consistently leads to models that are more accurate, interpretable, and data-efficient than purely black-box data-driven approaches.

The first direction focused on environmental perception and virtual sensing. Hybrid LSTM-CNN architectures and transformer-based temporal models achieved high fidelity microclimate reconstruction, while DRL-optimized rover trajectories significantly improved the spatial and informational quality of the collected data. The second direction addressed biological modeling in aquaculture, where embedding mechanistic bioenergetic descriptors into learning pipelines produced more robust and data-efficient growth predictions. The third direction examined multimodal plant phenotyping, showing that low-cost imaging combined with nonlinear growth modeling can yield physiologically meaningful developmental indicators.

Together, these results highlight the practical impact of hybrid modeling: improved sensing accuracy, reduced data requirements, and enhanced interpretability across diverse real-world systems.

In this context, it is important to clarify the role and limitations of the purely data-driven environmental components developed in Chapters 3 and 4. While the environmental components of the DT (Chapters 3 and 4) rely primarily on data-driven architectures, this choice reflects the practical limitations of applying physics-informed or mechanistic climate models in commercial greenhouse settings. Detailed energy-balance or CFD-based formulations require extensive calibration, high-resolution structural information, and computational resources that are incompatible with real-time DT operation. For this reason, structural descriptors were incorporated as contextual features rather than as explicit mechanistic constraints. These chapters, therefore, provide a data-driven environmental baseline that complements, rather than contradicts, the hybrid modeling strategy developed in later chapters.

At the same time, several open challenges remain, including cross-site generalization, robustness under operational uncertainty, and the integration of hybrid models into real-time decision support systems. Addressing these challenges represents a promising avenue for future research and for advancing DTs as reliable tools for biological and environmental management.

7.1.2 Detailed Contributions

This thesis, titled *Models from Models: Modeling Complex Systems by Combining Model-Based and Data-Driven Learning*, develops a comprehensive framework for constructing DTs of complex systems. The primary focus is not on a specific domain but on the broader methodological challenge of modeling systems exhibiting complexities such as nonlinear dynamics, partial observability, and multimodal data. Biological and environmental domains, such as aquaculture, greenhouse microclimates, and plant phenotyping, serve as representative case studies of such complexity. Through these examples, the thesis demonstrates that integrating mechanistic knowledge or biological principles with data-driven ML yields models that are more accurate, interpretable, and data-efficient than black-box learning or purely mechanistic formulations.

The conclusion of the work can be summarized through its primary contributions across three layers:

1. Environmental Perception and Data Acquisition

- **Virtual Sensing for Greenhouse Microclimate Reconstruction:** The thesis shows that greenhouse microclimates can be accurately reconstructed in unseen locations by combining structural knowledge of the greenhouse with data-driven learning. Three hybrid architectures, LSTM-only, and a combined LSTM, CNN model, were compared. Among these, the integrated LSTM-CNN hybrid achieved the strongest performance, reducing the MAPE to 6-8%. Building on this, the Time-LLM framework further improved reconstruction accuracy, achieving a superior 1.0-1.5% MAPE, demonstrating the potential of transformer-based temporal representations to significantly enhance the development of environmental DTs.
- **Autonomous Sensing:** The thesis demonstrates that autonomous rovers can enhance the quality and spatial diversity of data used for virtual sensing. Several DRL strategies were evaluated to optimize rover trajectories, including a DQL method and a PPO policy. Both DRL approaches outperformed fixed-trajectory sampling by actively guiding the rover toward locations with higher expected information gain. Among them, PPO proved the most effective, achieving a superior balance between exploration, information value, and operational constraints such as battery capacity and travel distance. By generating richer and more representative microclimate measurements, these DRL-optimized rovers produced higher-quality datasets that improved the accuracy of the virtual sensing models developed in this thesis.

2. Biological Modeling and Rainbow Trout Weight Prediction

In the aquaculture case study, the thesis demonstrates that integrating mechanistic bioenergetic descriptors, such as energy intake, catabolic losses, and somatic energy content, substantially improves fish-growth prediction. Rather than imposing hard mechanistic constraints during optimization, the bioenergetic model serves as a principled transformation layer that enriches the input space with physiologically meaningful features. The biological plausibility of the hybrid framework is therefore preserved through the structure of the bioenergetic equations, which enforce consistent physiological relationships, while the data-driven component learns the complex nonlinear mapping between these descriptors and fish growth. This lightweight integration enables the model to capture growth dynamics that purely data-driven approaches often fail to represent, while remaining computationally efficient. Future work may further strengthen the interaction between mechanistic and data-driven components by incorporating physics-informed regularization or constraint-based feedback mechanisms.

A key finding is the strong data efficiency of the proposed hybrid model. Even when trained on only 50% of the available dataset, the hybrid approach outperformed a fully data-driven baseline trained on the entire dataset. This result highlights the value of the hybrid mechanistic, data-driven integration for biological DTs, particularly in domains where high-quality ground truth data are costly or difficult to obtain.

3. Multimodal Plant Phenotyping

In the plant phenotyping case study, the thesis develops a DT of *Salicornia* growth by synchronizing low-cost visual traits, such as projected area and convex hull area, with nonlinear growth models including the Logistic and Gompertz functions. These models were used not only to estimate growth trajectories but also to analyze developmental stages. By fitting nonlinear functions to structural features extracted from simple 2D images, the framework directly derives biologically meaningful critical points, such as the inflection point, rapid-growth interval, and onset of deceleration, from low-cost visual data.

Through the derivatives of these fitted growth curves (velocity and acceleration), the DT produces physiologically interpretable indicators that characterize plant growth stages. This demonstrates that even minimal imaging hardware, when combined with principled nonlinear modeling, can yield developmental insights typically associated with high-end phenotyping systems.

To support this analysis, the thesis also develops a multimodal dataset that integrates environmental measurements, such as temperature and salinity, with structural features extracted from plant images. This dataset enables the DT to link environmental drivers with morphological development, providing a richer foundation for both plant weight estimation and developmental stage analysis.

Taken together, these case studies illustrate the broader contribution of this thesis: a unified approach to modeling complex systems that combines mechanistic understanding with data-driven learning. Across environmental monitoring, aquaculture, and plant phenotyping, the DTs developed here demonstrate how hybrid modeling can overcome the limitations of purely black-box data-driven methods or purely mechanistic approaches. Virtual sensing benefited from deep temporal architectures and transformer-based representations; autonomous sensing was strengthened through DRL policies such as DQL and PPO that generated richer, more informative datasets; biological growth prediction improved through the integration of mechanistic bioenergetics; and plant phenotyping advanced through nonlinear growth modeling and the extraction of critical developmental points from low-cost image-derived traits, supported by a multimodal dataset linking environmental and structural information.

Collectively, these results show that hybrid modeling is not domain-specific but a generalizable strategy for capturing the dynamics of complex, partially observed, and multimodal systems, providing a foundation for more interpretable, data-efficient, and biologically grounded DTs.

Appendix A

Mathematical Formulations of Nonlinear Growth Models

A.1 Logistic Model

The logistic model is one of the most commonly used functions to describe bounded growth. Pierre-François Verhulst originally introduced it to model population dynamics under limited resources and has since been widely adopted in biological and phenotyping studies [212]. The key idea of the model is that growth is initially fast but gradually slows down as physical or biological limitations are reached.

The Logistic growth process can be expressed in differential form as

$$\frac{dy}{dt} = ry \left(1 - \frac{y}{K}\right) \quad (\text{A.1})$$

In this formulation, y denotes the size of the population or biological quantity of interest at time t , such as biomass, leaf area, or plant height. The parameter r represents the intrinsic growth rate, capturing the maximum potential rate of increase under ideal conditions where resources are abundant. The term K corresponds to the carrying capacity, which defines the maximum attainable size that the environment can sustain given limitations in nutrients, light, water, or physical space. The expression $(1 - \frac{y}{K})$ quantifies the remaining growth potential relative to this upper limit. When y is small compared to K , this term is close to one, and growth proceeds rapidly. As y approaches K , the term declines toward zero, indicating increasing competition and diminishing resources, which gradually slow the growth rate until it stabilizes at the carrying capacity.

This equation states that the rate of change of the variable y depends on its current value and on the remaining capacity for growth. When y is small compared to the maximum attainable value K , growth is approximately exponential. As y increases and approaches K , the growth rate progressively decreases, eventually tending toward zero. In this way, the model naturally represents saturation effects without requiring additional constraints.

For practical purposes, such as curve fitting and regression analysis, the integrated (closed-form) solution of the Logistic equation is typically employed as:

$$y(t) = \frac{K}{1 + ae^{-rt}} \quad (\text{A.2})$$

where K denotes the asymptotic maximum value, r controls the overall growth speed, and a is a constant related to the initial condition.

An alternative but equivalent formulation explicitly introduces the time of maximum growth, known as the inflection point t_i :

$$y(t) = \frac{K}{1 + e^{-r(t-t_i)}} \quad (\text{A.3})$$

This parameterization is often preferred for interpretation, as t_i directly indicates the time at which the growth rate is highest. Biological Interpretation of this model is as follows:

The Logistic model is distinguished by its inherent symmetry: the point of maximum growth rate (the inflection point, where dy/dt reaches its peak) occurs precisely at $y = K/2$, corresponding to 50% of the

final attainable size [212]. This property makes the Logistic curve particularly suitable for processes in which growth accelerates and decelerates in a balanced manner around a central midpoint.

A.2 The Gompertz Model

The Gompertz model was originally developed for mortality and survival analysis but has since been widely applied to describe biological growth processes, including plant biomass accumulation, particularly when the growth trajectory is asymmetric [213]. Compared with the Logistic model, the Gompertz function allows for a more gradual initial growth phase and an earlier slowdown.

The Gompertz growth process can be written in differential form as:

$$\frac{dy}{dt} = ry \ln \left(\frac{K}{y} \right) \quad (\text{A.4})$$

In this formulation, the growth rate depends on the current size y , the growth coefficient r , and a logarithmic term that reflects the remaining potential for growth. Unlike the Logistic model, where growth slows linearly as the maximum value is approached, the logarithmic term $\ln(K/y)$ causes growth to decelerate more rapidly as y increases, leading to an asymmetric growth curve.

For estimation and regression purposes, the integrated (closed-form) expression of the Gompertz model is commonly used:

$$y(t) = K e^{-b e^{-rt}} \quad (\text{A.5})$$

where K represents the asymptotic maximum value and b is a shape parameter related to the initial condition and the horizontal displacement of the curve.

This model is intrinsically asymmetric, meaning that the growth and saturation phases are not mirror images of each other. In this model, the point of maximum growth rate (the inflection point) occurs relatively early, at approximately $y = K/e$, corresponding to about 37% of the asymptotic maximum value [213].

A.3 The Grey Verhulst Model

The Grey Verhulst model originates from Grey System Theory, a framework developed to analyze systems in which information is incomplete, uncertain, or partially unknown conditions commonly described as “grey” [214]. In contrast to classical regression methods, which typically require large, fully observed datasets, Grey models are explicitly designed to operate effectively under data scarcity and noise. This makes the Grey Verhulst formulation particularly suitable for biological growth processes where measurements may be limited, irregular, or affected by environmental variability.

Traditional regression methods attempt to fit a curve directly to observed data, including short-term fluctuations and measurement noise. In contrast, Grey System Theory assumes that raw observations consist of two components: a meaningful underlying signal (“white” information) and random disturbances (“black” noise). To reduce the influence of noise and reveal the dominant growth trend, Grey models apply an “Accumulated Generating Operation” (AGO). The AGO step in the Grey Verhulst model effectively acts as a low-pass filter, suppressing high-frequency noise while preserving the underlying sigmoidal growth trend. Instead of fitting all observed fluctuations, which can bias the estimation of growth-rate parameters, this model focuses on the dominant trend [215].

Given raw data

$$X^{(0)} = (x(1), x(2), \dots, x(n)),$$

the accumulated sequence is constructed as:

$$x^{(1)}(k) = \sum_{i=1}^k x^{(0)}(i) \quad (\text{A.6})$$

This accumulation transforms a noisy and potentially oscillatory sequence into a smooth, monotonically increasing series. The growth model is then fitted to this filtered representation rather than to the raw measurements.

In this study, we consider the Grey Verhulst GM(1,1) power model, expressed as:

$$\frac{dx^{(1)}}{dt} + az^{(1)} = b(z^{(1)})^2 \quad (\text{A.7})$$

where $z^{(1)}$ denotes the mean of two consecutive accumulated data points. The model parameters a and b are estimated using least squares, and predictions in the original data space are obtained through the Inverse Accumulated Generating Operation (IAGO) [154].

Appendix B

Nonlinear Model Fitting: Optimization Procedures and Parameter Constraints

Accurately fitting nonlinear growth models requires reliable optimization methods alongside careful management of incomplete biological observations. In this study, the Logistic and Gompertz models were fitted using a Non-Linear Least Squares (NLLS) framework to minimize the sum of squared residuals between model predictions and observed measurements; the Grey Verhulst model was fitted separately via the closed-form least-squares procedure described in Chapter A. To enhance robustness, the system automatically generated initial parameter estimates from the empirical distributions of each plant's growth data. These initial estimates served as starting points for the optimizer and were crucial for preventing convergence to suboptimal local minima, a common challenge in nonlinear regression [204].

The initialization strategy was fully automated and based on biological principles. Specifically, the asymptote was set to the maximum observed value and the inflection point to the median time point of the growth sequence. These heuristics provided stable and reproducible starting conditions for the optimization algorithm. Any model that failed to converge within the computational budget of 10,000 function evaluations was flagged as non-convergent and excluded from further comparison.

In addition to numerical convergence, modeling biological growth requires that the fitted parameters remain identifiable with the available data. This is especially important in non-destructive imaging of dense plant populations, where obtaining complete temporal profiles can be challenging. As described in Section 6.4.2.3, our dataset experiences two types of truncation that directly impact parameter identifiability.

The first type is left truncation, which occurred in Phase 1 when data collection began after the seedling establishment stage. This gap obscured the early lag phase and complicated the estimation of the lower asymptote and the onset of rapid growth.

The second type is right truncation, resulting from canopy overlap at maturity, which forced image processing to end before the stationary phase was reached. In these cases, the carrying capacity K must be inferred from partial deceleration trends, which increases the risk of making unrealistic extrapolations.

To address these challenges, we employed a Bounded Nonlinear Least Squares approach in which the parameters were constrained to biologically plausible ranges. The carrying capacity K was restricted to lie between $0.9 \times y_{\max}$ (to allow fitting through noise at the peak) and $3.0 \times y_{\max}$, which prevents the optimizer from predicting implausible saturation levels when the plateau is unobserved. The growth rate r was constrained to positive values, reflecting the irreversible nature of biomass accumulation. The parameter t_0 was allowed to take negative values (with $-\infty < t_0 < +\infty$). This is critical for Phase 1 data, as the inflection point (where maximum growth speed occurs) may have occurred before data collection began.

Together, these automated initialization and bounded optimization procedures ensured that the fitted Logistic and Gompertz models converged reliably and produced parameter estimates that accurately reflect the biological dynamics within the observable window, rather than artifacts of numerical instability or data truncation. The Grey Verhulst model, fitted using the closed-form procedure described in Chapter A, required no iterative initialization; however, its reconstructed trajectory was clipped to $[0, 3.0 \times y_{\max}]$ to enforce the same biological plausibility ceiling as the iterative models.

Bibliography

- [1] S. Uttrani, A. K. Rao, B. Kanekar, I. Vohra, and V. Dutt, *Assessment of Various Deep Reinforcement Learning Techniques in Complex Virtual Search-and-Retrieve Environments Compared to Human Performance*, pp. 139–155. Singapore: Springer Nature Singapore, 2023. xi, 22
- [2] M. Føre, K. Frank, T. Norton, E. Svendsen, J. A. Alfredsen, T. Dempster, H. Eguiraun, W. Watson, A. Stahl, L. M. Sunde, C. Schellewald, K. R. Skøien, M. O. Alver, and D. Berckmans, "Precision fish farming: A new framework to improve production in aquaculture," *Biosystems Engineering*, vol. 173, pp. 176–193, 2018. *Advances in the Engineering of Sensor-based Monitoring and Management Systems for Precision Livestock Farming*. xi, 69, 70
- [3] E. Brentarolli, S. Migliorini, D. Quaglia, and C. Tomazzoli, "Greenhouse climatic sensing through agricultural robots and recurrent neural networks," in *2023 IEEE International Workshop on Metrology for Agriculture and Forestry, MetroAgriFor*, pp. 108–113, 2023. Scopus EID: 2-s2.0-85186525655. xiv, 4, 31, 48, 57, 58, 60, 61
- [4] A. Sharifi, S. Migliorini, and D. Quaglia, "Optimizing trajectories for rechargeable agricultural robots in greenhouse climatic sensing using deep reinforcement learning with proximal policy optimization algorithm," *Future Internet*, vol. 17, no. 7, 2025. xiv, 5, 51, 57, 61
- [5] A. Sharifi, S. Migliorini, and D. Quaglia, "Optimizing the trajectory of agricultural robots in greenhouse," in *8th International Conference on Control, Automation and Diagnosis, ICCAD 2024*, 2024. xiv, 2, 5, 57, 60, 61
- [6] M. Grieves and J. Vickers, *Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems*, pp. 85–113. Cham: Springer International Publishing, 2017. 2
- [7] A. Fuller, Z. Fan, C. Day, and C. Barlow, "Digital twin: Enabling technologies, challenges and open research," *IEEE Access*, vol. 8, pp. 108952–108971, 2020. 2
- [8] A. Rasheed, O. San, and T. Kvamsdal, "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, vol. 8, pp. 21980–22012, 2020. 2
- [9] M. Bolzonella, E. Royer, A. Lima, and R. Pastres, "Predicting farmed rainbow trout weight distribution to improve feeding practice: an individual-based model approach. [version 1; peer review: 1 approved with reservations, 1 not approved]," *Open Research Europe*, vol. 2, no. 63, 2022. 2, 70
- [10] J. R. Banga and A. F. Villaverde, "Mechanistic dynamic modelling of biological systems: The road ahead," *Current Opinion in Systems Biology*, vol. 42, p. 100553, 2025. 2
- [11] J. M. Hunde, T. S. Ochoño, D. Senevirathne, D. D. Eneyew, G. T. Bitsuamlak, M. A. Capretz, and K. Grolinger, "Data-driven and physics-based modeling approaches and their integration in building digital twins: A systematic review," *Journal of Building Engineering*, vol. 114, p. 114214, 2025. 2
- [12] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021. 2
- [13] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2405–2415, 2019. 2
- [14] M. B. Jamshidi, D. T. Hoang, D. N. Nguyen, D. Niyato, and M. E. Warkiani, "Revolutionizing biological digital twins: Integrating internet of bio-nano things, convolutional neural networks, and federated learning," *Computers in Biology and Medicine*, vol. 189, p. 109970, 2025. 2
- [15] J. Chen, Y. Xu, H. Li, X. Zhao, Y. Su, C. Qi, K. Qu, and Z. Cui, "The application of digital twin technology in the development of intelligent aquaculture: Status and opportunities," *Fishes*, vol. 10, no. 8, 2025. 2

- [16] S. Revathi, N. Sivakumaran, and T. Radhakrishnan, "Design of solar-powered forced ventilation system and energy-efficient thermal comfort operation of greenhouse," *Materials Today: Proceedings*, vol. 46, pp. 9893–9900, 2021. 2, 26
- [17] A. Sharifi, S. Migliorini, and D. Quaglia, "Combining convolutional and recurrent neural networks to improve greenhouse microclimate mapping," in *2024 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*, pp. 63–68, 2024. 2, 4, 36, 38
- [18] E. Brentarolli, S. Migliorini, D. Quaglia, and C. Tomazzoli, "Mapping micro-climate in a greenhouse through a context-aware recurrent neural network," in *2023 IEEE Conference on AgriFood Electronics, CAFE*, pp. 113–117, 2023. Scopus EID: 2-s2.0-85178561996. 4, 28, 30, 38, 61
- [19] A. Sharifi, S. Migliorini, and D. Quaglia, "The power of timellm for greenhouse microclimate mapping," in *2025 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*, IEEE, 2025. 4, 32
- [20] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. 9, 10
- [21] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958. PMID: 13602029. 9
- [22] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986. 10
- [23] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014. 10
- [24] H. Jaeger, "Tutorial on training recurrent neural networks, covering bppt, rtrl, ekf and the echo state network approach," *GMD-Forschungszentrum Informationstechnik, 2002.*, vol. 5, 01 2002. 10
- [25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017. 10, 15
- [26] X. Zhou, H. Liu, C. Shi, and J. Liu, "Chapter 2 - the basics of deep learning," in *Deep Learning on Edge Computing Devices* (X. Zhou, H. Liu, C. Shi, and J. Liu, eds.), pp. 19–36, Elsevier, 2022. 10
- [27] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. 11
- [28] I. D. Mienye, T. G. Swart, and G. Obaido, "Recurrent neural networks: A comprehensive review of architectures, variants, and applications," *Information*, vol. 15, no. 9, 2024. 11, 12, 14
- [29] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," *International Journal of Forecasting*, vol. 37, no. 1, pp. 388–427, 2021. 12
- [30] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015. 12
- [31] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994. 12
- [32] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," 2013. 12
- [33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, p. 1735–1780, Nov. 1997. 12, 14
- [34] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015. 14

- [35] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: decomposition transformers with auto-correlation for long-term series forecasting," in *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, (Red Hook, NY, USA), Curran Associates Inc., 2021. 16
- [36] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 11106–11115, AAAI Press, 2021. 16
- [37] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?," *AAAI Conference on Artificial Intelligence*, 2023. 16
- [38] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan, and Q. Wen, "Time-LLM: Time series forecasting by reprogramming large language models," in *International Conference on Learning Representations (ICLR)*, 2024. 16, 32
- [39] S. S. Mousavi, M. Schukat, and E. Howley, "Deep reinforcement learning: An overview," in *Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016* (Y. Bi, S. Kapoor, and R. Bhatia, eds.), pp. 426–440, 2018. 16, 46
- [40] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018. 17, 18
- [41] DeepMind, "Deepmind ai reduces google data centre cooling bill by 40%," July 2016. Accessed: 2025-09-23. 18
- [42] Y. He, J. Zhang, M. Wei, Y. Guo, Y. Wang, and Y. Huang, "Reinforcement learning based motion planning for robotic arm of welding robot," in *Robotics and Autonomous Systems and Engineering Applications of Computational Intelligence* (J. Gu, F. Hu, H. Zhou, Z. Fei, and E. Yang, eds.), (Singapore), pp. 59–70, Springer Nature Singapore, 2024. 18
- [43] M. Ghasemi, A. H. Moosavi, and D. Ebrahimi, "A comprehensive survey of reinforcement learning: From algorithms to practical challenges," 2025. 18
- [44] Q. Technologies, "10 real-life applications of reinforcement learning in 2025," June 2025. Accessed: 2025-09-23. 18
- [45] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015. 19, 20, 46
- [46] D. Han, B. Mulyana, V. Stankovic, and S. Cheng, "A survey on deep reinforcement learning algorithms for robotic manipulation," *Sensors*, vol. 23, no. 7, 2023. 19
- [47] C. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992. 19
- [48] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013. 20
- [49] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2019. 21, 47
- [50] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. 21, 47
- [51] O. Joliet, "Hortitrans, a model for predicting and optimizing humidity and transpiration in greenhouses," *Journal of agricultural engineering research*, vol. 57, no. 1, pp. 23–37, 1994. 26
- [52] M. Van Pee, K. Janssens, D. Berckmans, R. Lemeur, and L. Marcelis, "Dynamic measurement and modelling of climate gradients around a plant for micro-environment control," 1998-01-01. 26

- [53] Y. Zhao, M. Teitel, and M. Barak, "Se—structures and environment: Vertical temperature and humidity gradients in a naturally ventilated greenhouse," *Journal of Agricultural Engineering Research*, vol. 78, no. 4, pp. 431–436, 2001. 26
- [54] J. Roy, T. Boulard, C. Kittas, and S. Wang, "Pa—precision agriculture: Convective and ventilation transfers in greenhouses, part 1: the greenhouse considered as a perfectly stirred tank," *Biosystems Engineering*, vol. 83, no. 1, pp. 1–20, 2002. 26
- [55] C. Kittas, T. Bartzanas, and A. Jaffrin, "Temperature gradients in a partially shaded large greenhouse equipped with evaporative cooling pads," *Biosystems Engineering*, vol. 85, no. 1, pp. 87–94, 2003. 26
- [56] C. Chen, "Prediction of longitudinal variations in temperature and relative humidity eor evaporative cooling greenhouses," *Agricultural engineering journal*, vol. 12, no. 3&4, pp. 143–164, 2003. 26
- [57] C. Kittas and T. Bartzanas, "Greenhouse microclimate and dehumidification effectiveness under different ventilator configurations," *Building and Environment*, vol. 42, no. 10, pp. 3774–3784, 2007. 26
- [58] C. R. Bojacá, R. Gil, and A. Cooman, "Use of geostatistical and crop growth modelling to assess the variability of greenhouse tomato yield caused by spatial temperature variations," *Computers and Electronics in Agriculture*, vol. 65, no. 2, pp. 219–227, 2009. 26
- [59] S. Piñon, E. Camacho, B. Kuchen, and M. Peña, "Constrained predictive control of a greenhouse," *Computers and Electronics in Agriculture*, vol. 49, pp. 317–329, 2005. 27
- [60] S. Speetjens, J. Stigter, and G. Van Straten, "Towards an adaptive model for greenhouse control," *Computers and Electronics in Agriculture*, vol. 67, pp. 1–8, 2009. 27
- [61] I. Hameed, "Using the extended kalman filter to improve the efficiency of greenhouse climate control," *International Journal of Innovative Computing, Information and Control*, vol. 6, pp. 2671–2680, 2010. 27
- [62] J. Lopez-Martinez, J. Blanco, and J. Perez, "Distributed network for measuring climatic parameters in heterogeneous environments: application in a greenhouse," *Computers and Electronics in Agriculture*, vol. 145, pp. 105–121, 2018. 27
- [63] T. Hamrita, "A wireless sensor network for precision irrigation in tunisia," *International Journal of Distributed Sensor Networks*, vol. 1, no. 1, pp. 67–74, 2005. 27
- [64] M. Teitel and E. Wenger, "Air exchange and ventilation efficiencies of a non-span greenhouse with one inflow and one outflow through longitudinal side openings," *Biosystems Engineering*, vol. 119, pp. 98–107, 2014. 27
- [65] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002. 27
- [66] L. Gonda and C. E. Cugnasca, "A proposal of greenhouse control using wireless sensor networks," in *World Congress on Computers in Agriculture and Natural Resources*, American Society of Agricultural and Biological Engineers, 2006. 27
- [67] A. Pawlowski *et al.*, "Monitoring and controlling the greenhouse environment," *Control Engineering Practice*, vol. 87, pp. 1–10, 2019. 27
- [68] D. Chaudhary, S. Nayse, and L. Waghmare, "Application of wireless sensor networks for greenhouse parameter control in precision agriculture," *International Journal of Wireless & Mobile Networks*, vol. 3, pp. 140–149, 02 2011. 27
- [69] M. Srbinovska, C. Gavrovski, and V. Dimcev, "Environmental parameters monitoring in precision agriculture using wireless sensor networks," *Journal of Cleaner Production*, vol. 88, pp. 297–307, 2015. 27

- [70] R. Pahuja, "An intelligent wireless sensor and actuator network system for greenhouse microenvironment control and assessment," *Journal of Biosystems Engineering*, vol. 42, no. 1, pp. 23–43, 2017. 27
- [71] P. Davis *et al.*, "Adaptive temperature control in a greenhouse using ventilator adjustments," *Control Engineering Practice*, vol. 3, pp. 1163–1170, 1995. 27
- [72] R. Piñón *et al.*, "Scheme for the temperature control of a greenhouse based on data-driven virtual sensors," *Journal of Process Control*, vol. 18, pp. 1–10, 2008. 27
- [73] Z. Ankara, T. Kammerer, A. Gramm, and A. Schütze, "Low power virtual sensor array based on a micromachined gas sensor for fast discrimination between h₂, co and relative humidity," *Sensors and Actuators B: Chemical*, vol. 100, pp. 240–245, 2004. 27
- [74] T. Wenzel, K. Burnham, M. Blundell, and R. Williams, "Kalman filter as a virtual sensor: Applied to automotive stability systems," *Transactions of the Institute of Measurement and Control*, vol. 29, pp. 95–115, 2007. 27
- [75] F. Bellas, J. Becerra, J. Santos, and R. Duro, "Applying synaptic delays for virtual sensing and actuation in mobile robots," in *International Joint Conference on Neural Networks*, pp. 144–149, 2000. 27
- [76] W. Xue, Y.-Q. Guo, and X.-D. Zhang, "Application of a bank of kalman filters and a robust kalman filter for aircraft engine sensor/actuator fault diagnosis," *International Journal of Innovative Computing, Information and Control*, vol. 4, pp. 3161–3168, 2008. 27
- [77] C. Guzmán Valdivia, J. Escobedo, H. Durán-Muñoz, J. Berumen, A. Ortiz-Hernández, and *et al.*, "Implementation of virtual sensors for monitoring temperature in greenhouses using cfd and control," *Sensors*, vol. 19, pp. 1–13, jan 2019. 27
- [78] B. Lin, B. Recke, J. K. Knudsen, and S. B. Jørgensen, "A systematic approach for soft sensor development," *Computers & Chemical Engineering*, vol. 31, no. 5, pp. 419–425, 2007. ESCAPE-15. 27
- [79] V. Prasad, M. Schley, L. Russo, and B. Bequette, "Product property and production rate control of styrene polymerization," *Journal of Process Control*, vol. 12, pp. 353–372, 2002. 27
- [80] S. Park and C. Han, "A nonlinear soft sensor based on multivariate smoothing procedure for quality estimation in distillation columns," *Computers & Chemical Engineering*, vol. 24, pp. 871–877, 2000. 27
- [81] V. Radhakrishnan and A. Mohamed, "Neural networks for the identification and control of blast furnace hot metal quality," *Journal of Process Control*, vol. 10, pp. 509–524, 2000. 27
- [82] S. Reichrath and T. Davies, "Using cfd to model the internal climate of greenhouses: Past, present and future," *Agronomie*, vol. 22, pp. 3–19, 2002. 27
- [83] G. De la Torre-Gea, G. Soto-Zarazúa, I. López-Crúz, I. Torres-Pacheco, and E. Rico-García, "Computational fluid dynamics in greenhouses: A review," *African Journal of Biotechnology*, vol. 10, pp. 17651–17662, 2011. 27
- [84] A. Sonoda, Y. Takayama, A. Sugawara, and H. Nishi, "Greenhouse heat map generation with deep neural network using limited number of temperature sensors," in *IECON 2022 – 48th Annual Conference of the IEEE Industrial Electronics Society*, pp. 1–6, 2022. 27, 28
- [85] F. Molina-Aiz, H. Fatnassi, T. Boulard, J. Roy, and D. Valera, "Comparison of finite element and finite volume methods for simulation of natural ventilation in greenhouses," *Computers and Electronics in Agriculture*, vol. 72, pp. 69–86, 2010. 27
- [86] G. Tong, D. Christopher, and B. Li, "Numerical modelling of temperature variations in a chinese solar greenhouse," *Computers and Electronics in Agriculture*, vol. 68, pp. 129–139, 2009. 27
- [87] J. Deltour, D. De Halleux, J. Nijskens, S. Coutisse, and A. Nisen, "Dynamic modelling of heat and mass transfer in greenhouses," in *Symposium Greenhouse Climate and Its Control*, pp. 119–126, 1985. 27

- [88] G. Longo and A. Gasparella, "Comparative experimental analysis and modelling of a flower greenhouse equipped with a desiccant system," *Applied Thermal Engineering*, vol. 47, pp. 54–62, 2012. 27
- [89] X. Cheng, D. Li, L. Shao, and Z. Ren, "A virtual sensor simulation system of a flower greenhouse coupled with a new temperature microclimate model using three-dimensional cfd," *Computers and Electronics in Agriculture*, vol. 181, p. 105934, 2021. 28
- [90] A. Usman, M. Rafiq, M. Saeed, A. Nauman, A. Almqvist, and M. Liwicki, "Machine learning computational fluid dynamics," in *2021 Swedish Artificial Intelligence Society Workshop (SAIS)*, pp. 1–4, IEEE, 2021. 28
- [91] E. Brentarolli, S. Locatelli, C. Nicoletto, P. Sambo, D. Quaglia, and R. Muradore, "A spatio-temporal methodology for greenhouse microclimatic mapping," *PLoS ONE*, vol. 19, p. e0310454, Sep. 2024. 28
- [92] R. Laidi, D. Djenouri, and I. Balasingham, "On predicting sensor readings with sequence modeling and reinforcement learning for energy-efficient iot applications," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 8, pp. 5140–5151, 2022. 28
- [93] C. Bettini, X. S. Wang, and S. Jajodia, "Temporal granularity," in *Encyclopedia of Database Systems* (L. Liu and M. T. Özsu, eds.), pp. 2968–2973, , 2009. 28
- [94] H. Enderton, *Elements of Set Theory*. Elsevier Science, 1977. 28
- [95] A. Sharifi, S. Migliorini, and D. Quaglia, "Combining convolutional and recurrent neural networks to improve greenhouse microclimate mapping," in *2024 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*, pp. 63–68, IEEE, 2024. 30
- [96] H. V. Dudukcu, M. Taskiran, Z. G. Cam Taskiran, and T. Yildirim, "Temporal convolutional networks with rnn approach for chaotic time series prediction," *Applied Soft Computing*, vol. 133, p. 109945, 2023. 31
- [97] J. Sánchez-Molina, F. Rodríguez, J. Moreno, J. Sánchez-Hermosilla, and A. Giménez, "Robotics in greenhouses. scoping review," *Computers and Electronics in Agriculture*, vol. 219, p. 108750, 2024. 46
- [98] C. W. Bac, E. J. van Henten, J. Hemming, and Y. Edan, "Harvesting robots for high-value crops: State-of-the-art review and challenges ahead," *Journal of Field Robotics*, vol. 31, no. 6, pp. 888–911, 2014. 46
- [99] W. Mao, H. Liu, W. Hao, F. Yang, and Z. Liu, "Development of a combined orchard harvesting robot navigation system," *Remote Sensing*, vol. 14, p. 675, 01 2022. 46
- [100] G. Sara, G. Todde, D. Pinna, and M. Caria, "Evaluating an autonomous electric robot for real farming applications," *Smart Agricultural Technology*, vol. 9, p. 100595, 2024. 46
- [101] E. J. van Henten, J. Hemming, B. A. J. van Tuijl, J. G. Kornet, J. Meuleman, J. Bontsema, and E. A. van Os, "An autonomous robot for harvesting cucumbers in greenhouses," *Autonomous Robots*, vol. 13, no. 3, pp. 241–258, 2002. 46
- [102] L. Nguyen, S. Kodagoda, R. Ranasinghe, and G. Dissanayake, "Mobile robotic sensors for environmental monitoring using gaussian markov random field," *Robotica*, vol. 39, no. 5, p. 862–884, 2021. 46
- [103] A. Viseras, D. Shutin, and L. Merino, "Robotic active information gathering for spatial field reconstruction with rapidly-exploring random trees and online learning of gaussian processes," *Sensors*, vol. 19, no. 5, 2019. 46
- [104] I. K. Waarum, A. van Hove, T. R. Krogstad, K. O. Ellefsen, and A. E. A. Blomberg, "Mixtures of gaussian processes for robotic environmental monitoring of emission sources," *Environmental Monitoring and Assessment*, vol. 197, no. 6, p. 626, 2025. 46
- [105] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the Operations Research Society of America*, vol. 2, no. 4, pp. 393–410, 1954. 46

- [106] A. Singh, A. Krause, C. Guestrin, W. Kaiser, and M. Batalin, "Efficient planning of informative paths for multiple robots," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI'07*, (San Francisco, CA, USA), p. 2204–2211, Morgan Kaufmann Publishers Inc., 2007. 46
- [107] A. Krause and C. Guestrin, "Near-optimal observation selection using submodular functions," in *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 2, AAAI'07*, p. 1650–1654, AAAI Press, 2007. 46
- [108] M. Mondal, F. Dou, J. Bi, and S. Han, "Deep q-learning-based mobile charger path planning in wireless powered communication networks," *ACM Trans. Embed. Comput. Syst.*, vol. 24, no. 6, 2025. 46
- [109] C. Chen, J. Yu, and S. Qian, "An enhanced deep q network algorithm for localized obstacle avoidance in indoor robot path planning," *Applied Sciences*, vol. 14, no. 23, 2024. 46
- [110] C.-C. Wong, K.-D. Weng, and B.-Y. Yu, "Multi-robot navigation system design based on proximal policy optimization algorithm," *Information*, vol. 15, no. 9, 2024. 47
- [111] L. Wang and G. Liu, "Research on multi-robot collaborative operation in logistics and warehousing using a3c optimized yolov5-ppo model," *Frontiers in Neurorobotics*, vol. 17, p. 1329589, 2024. 47
- [112] X. Jin and W. Zhengxiao, "Proximal policy optimization based dynamic path planning algorithm for mobile robots," *Electronics Letters*, vol. 58, 10 2021. 47
- [113] M. Segovia and J. Garcia-Alfaro, "Design, modeling and implementation of digital twins," *Sensors*, vol. 22, no. 14, 2022. 66
- [114] A. F. Villaverde, D. Pathirana, F. Fröhlich, J. Hasenauer, and J. R. Banga, "A protocol for dynamic model calibration," *Briefings in Bioinformatics*, vol. 23, p. bbab387, 10 2021. 66
- [115] G. A. Alsalloum, N. M. Al Sawaftah, K. M. Percival, and G. A. Hussein, "Digital twins of biological systems: A narrative review," *IEEE Open J. Eng. Med. Biol.*, vol. 5, pp. 670–677, July 2024. 66
- [116] F. Bravo, J. Amorim, M. B. Amirkandeh, P. Bodorik, V. Cerqueira, N. R. Gomes, J. Korus, M. Oliveira, M. Parent, J. Pimentel, D. Reilly, T. Scłodnick, J. Grant, R. Filgueira, C. Whidden, and L. Torgo, "Advancing precision aquaculture through big data analytics and machine learning in canadian fish farming," in *OCEANS 2024 - Halifax*, pp. 1–6, 2024. 66, 71
- [117] L. L. Botelho, C. Jeynes-Smith, S. Vollert, and M. Bode, "Ecosystem models cannot predict the consequences of conservation decisions," 2024. 66
- [118] S. Villon, C. Iovan, M. Mangeas, and L. Vigliola, "Confronting deep-learning and biodiversity challenges for automatic video-monitoring of marine ecosystems," *Sensors*, vol. 22, no. 2, 2022. 66
- [119] Z. Barta, "Deep learning in terrestrial conservation biology," *Biologia Futura*, vol. 74, no. 4, pp. 359–367, 2023. 66
- [120] B. Noordijk, M. L. Garcia Gomez, K. H. W. J. ten Tusscher, D. de Ridder, A. D. J. van Dijk, and R. W. Smith, "The rise of scientific machine learning: a perspective on combining mechanistic modelling with machine learning for systems biology," *Frontiers in Systems Biology*, vol. Volume 4 - 2024, 2024. 66
- [121] F. Dietrich and W. Schilders, "Scientific machine learning," *Mathematische Semesterberichte*, vol. 72, no. 2, pp. 89–115, 2025. 67
- [122] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient pathologies in physics-informed neural networks," 2020. 67, 72
- [123] Food and A. O. of the United Nations (FAO), "Global fisheries and aquaculture production reaches a new record high," 2023. Accessed: 2024-12-27. 68, 69
- [124] N. Tonachella, A. Martini, M. Martinoli, D. Pulcini, A. Romano, and F. Capoccioni, "An affordable and easy-to-use tool for automatic fish length and weight estimation in mariculture," *Scientific Reports*, vol. 12, 09 2022. 69, 70

- [125] W. A. Society, "A decadal outlook for global aquaculture," 2023. Accessed: 2024-12-27. 69
- [126] D. Stanković, C. Alain, and A. Snoj, "Rainbow trout in europe: Introduction, naturalization, and impacts," *Reviews in Fisheries Science & Aquaculture*, vol. 23, pp. 39–71, 01 2015. 69
- [127] E. D'Agaro, P. Gibertoni, and S. Esposito, "Recent trends and economic aspects in the rainbow trout (*Oncorhynchus mykiss*) sector," *Applied Sciences*, vol. 12, no. 18, p. 8773, 2022. 70
- [128] D. Brigolin, R. Pastres, P. Tomassetti, and S. Porrello, "Modelling the biomass yield and the impact of seabream mariculture in the Adriatic and Tyrrhenian Seas (Italy)," *Aquaculture International*, vol. 18, pp. 149–163, 04 2010. 70, 71
- [129] D. Brigolin, V. Meccia, C. Venier, P. Tomassetti, S. Porrello, and R. Pastres, "Modelling biogeochemical fluxes across a mediterranean fish cage farm," *Aquaculture Environment Interactions*, vol. 5, pp. 71–88, 04 2014. 71
- [130] M. Bolzonella, E. Royer, A. Lima, and R. Pastres, "Predicting farmed rainbow trout weight distribution to improve feeding practice: an individual-based model approach.," *Open Research Europe*, vol. 2, p. 63, 05 2022. 71, 72, 73, 82
- [131] H. Li, Y. Chen, W. Li, Q. Wang, Y. Duan, and T. Chen, "An adaptive method for fish growth prediction with empirical knowledge extraction," *Biosystems Engineering*, vol. 212, pp. 336–346, 2021. 71
- [132] B. Kooijman, *Dynamic Energy Budget Theory for Metabolic Organisation*. Cambridge University Press, 3 ed., 2009. 71
- [133] L. von Bertalanffy, *Quantitative laws in metabolism and growth*. London: Methuen, 1957. 71
- [134] W. Vásquez-Quispesivana, M. Guevara, and I. Pallardel, "Artificial intelligence in aquaculture: basis, applications, and future perspectives," *Scientia Agropecuaria*, vol. 13, pp. 79–96, 03 2022. 71
- [135] G. Kaur, N. Adhikari, S. Krishnapriya, S. Wawale, R. Malik, A. Zamani, J. Pérez Falcón, and J. oseí owusu, "Recent advancements in deep learning frameworks for precision fish farming opportunities, challenges, and applications," *Journal of Food Quality*, vol. 2023, 02 2023. 71
- [136] N. Zhu, X. Ji, J. Tan, Y. Jiang, and Y. Guo, "Prediction of dissolved oxygen concentration in aquatic systems based on transfer learning," *Computers and Electronics in Agriculture*, vol. 180, p. 105888, 2021. 71
- [137] M. Rana, A. Rahman, J. Dabrowski, S. Arnold, J. McCulloch, and B. Pais, "Machine learning approach to investigate the influence of water quality on aquatic livestock in freshwater ponds," *Biosystems Engineering*, vol. 208, pp. 164–175, 2021. 71
- [138] Y. Yang, H. Tai, and D. Li, "Real-time optimized prediction model for dissolved oxygen in crab aquaculture ponds using back propagation neural network," *Sensor Letters*, vol. 12, 03 2014. 71
- [139] M. Islam, M. Kashem, and J. Uddin, "Fish survival prediction in an aquatic environment using random forest model," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 10, pp. 614–622, 09 2021. 71
- [140] M. M. Rashid, A.-A. Nayan, S. Simi, J. Saha, M. Rahaman, and M. Kibria, "Iot based smart water quality prediction for biofloc aquaculture," *International Journal of Advanced Computer Science and Applications*, vol. 12, p. 7, 07 2021. 71
- [141] Y. Fu, Z. Hu, Y. Zhao, and M. Huang, "A long-term water quality prediction method based on the temporal convolutional network in smart mariculture," *Water*, vol. 13, no. 20, 2021. 71
- [142] L. C. B. d. Silva, B. D. M. Lopes, I. M. Blanquet, and C. A. F. Marques, "Gaussian distribution model for detecting dangerous operating conditions in industrial fish farming," *Applied Sciences*, vol. 11, no. 13, 2021. 71

- [143] A. Saad, A. Baikas, M. Remen, and F. O. Bjørnson, "Optimizing feeding strategies in aquaculture using machine learning: Ensuring sustainable and economically viable fish farming practices," *Procedia Computer Science*, vol. 246, pp. 4712–4721, 2024. 28th International Conference on Knowledge Based and Intelligent information and Engineering Systems (KES 2024). 71
- [144] M. Burke, D. Nikolic, P. Fabry, H. Rishi, T. Telfer, and S. Rey Planellas, "Precision farming in aquaculture: non-invasive monitoring of atlantic salmon (*salmo salar*) behaviour in response to environmental conditions in commercial sea cages for health and welfare assessment," *Frontiers in Robotics and AI*, vol. Volume 12 - 2025, 2025. 71
- [145] A. Quarteroni, P. Gervasio, and F. Regazzoni, "Combining physics-based and data-driven models: advancing the frontiers of research with scientific machine learning," *Mathematical Models and Methods in Applied Sciences*, vol. 35, p. 905–1071, Mar. 2025. 71
- [146] E. Pino-Martinez, P. Balseiro, H. F. Kvittingen, C. Pedrosa, M. Gorissen, and S. O. Handeland, "Influence of salinity on rainbow trout (*oncorhynchus mykiss*) smolt development and postsmolt performance," *Aquaculture*, vol. 587, p. 740874, 2024. 73
- [147] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," 2022. 75
- [148] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," 2017. 75
- [149] GAIN Consortium, "Report on instrumentation of gain pilot sites," Tech. Rep. Deliverable D1.1, H2020 GAIN Project, 2019. Accessed: 2025-02-10. 77
- [150] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Analytical Chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964. 79
- [151] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. New York, NY, USA: Cambridge University Press, 2007. 79
- [152] T. Sun, L. Xiao, S. T. Ata-Ul-Karim, Y. Ma, and W. Zhang, "Editorial: Leveraging phenotyping and crop modeling in smart agriculture," *Frontiers in Plant Science*, vol. 16, p. 1626622, 2025. 91
- [153] M. Janni and R. Pieruschka, "Plant phenotyping for a sustainable future," *Journal of Experimental Botany*, vol. 73, pp. 5085–5088, 09 2022. 91
- [154] Q. Li, H. Gao, X. F. Zhang, n. Jiheng, and H. Mao, "Describing lettuce growth using morphological features combined with nonlinear models," *Agronomy*, vol. 12, p. 860, 03 2022. 91, 94, 127, 149
- [155] D. Song, K. De Silva, M. D. Brooks, and M. Kamruzzaman, "Biomass prediction based on hyperspectral images of the arabidopsis canopy," *Computers and Electronics in Agriculture*, vol. 210, p. 107939, 2023. 91
- [156] S. Cárdenas-Pérez, A. Piernik, A. Ludwiczak, M. Duszyn, A. Szmjdt-Jaworska, and J. J. Chanona-Pérez, "Image and fractal analysis as a tool for evaluating salinity growth response between two *Salicornia europaea* populations," *BMC Plant Biology*, vol. 20, no. 1, p. 467, 2020. 91
- [157] H. Gupta and R. Pahuja, "Estimating morphological features of plant growth using machine vision," *International Journal of Agricultural and Environmental Information Systems*, vol. 10, pp. 30–53, 01 2019. 91
- [158] P. Kumari, A. Bhatt, V. K. Meena, S. Adhikari, N. Dhar, H. Chawda, S. Chand, P. Joshi, V. Mangal, and S. Sood, "Plant phenomics: The force behind tomorrow's crop phenotyping tools," *Journal of Plant Growth Regulation*, vol. 44, no. 5, pp. 1791–1809, 2025. 92
- [159] J. Goudriaan and H. {van Laar}, *Modelling potential crop growth processes : textbook with exercises*. No. 2 in Current issues in production ecology, Netherlands: Kluwer Academic Publishers, 1994. 93
- [160] I. Supit, A. Hooijer, C. van Diepen, and C. of the European Communities. Joint Research Centre, *System Description of the WOFOST 6.0 Crop Simulation Model Implemented in CGMS.: Theory and algorithms*. No. v. 1 in Agriculture series, European Commission, 1994. 93

- [161] D. P. Holzworth, N. I. Huth, P. G. deVoil, E. J. Zurcher, N. I. Herrmann, G. McLean, K. Chenu, E. J. van Oosterom, V. Snow, C. Murphy, A. D. Moore, H. Brown, J. P. Whish, S. Verrall, J. Fainges, L. W. Bell, A. S. Peake, P. L. Poulton, Z. Hochman, P. J. Thorburn, D. S. Gaydon, N. P. Dalgliesh, D. Rodriguez, H. Cox, S. Chapman, A. Doherty, E. Teixeira, J. Sharp, R. Cichota, I. Vogeler, F. Y. Li, E. Wang, G. L. Hammer, M. J. Robertson, J. P. Dimes, A. M. Whitbread, J. Hunt, H. van Rees, T. McClelland, P. S. Carberry, J. N. Hargreaves, N. MacLeod, C. McDonald, J. Harsdorf, S. Wedgwood, and B. A. Keating, "Apsim – evolution towards a new generation of agricultural systems simulation," *Environmental Modelling & Software*, vol. 62, pp. 327–350, 2014. 93
- [162] J. Jones, G. Hoogenboom, C. Porter, K. Boote, W. Batchelor, L. Hunt, P. Wilkens, U. Singh, A. Gijsman, and J. Ritchie, "The dssat cropping system model," *European Journal of Agronomy*, vol. 18, no. 3, pp. 235–265, 2003. Modelling Cropping Systems: Science, Software and Applications. 93
- [163] N. Brisson, C. Gary, E. Justes, R. Roche, B. Mary, D. Ripoche, D. Zimmer, J. Sierra, P. Bertuzzi, P. Burger, F. Bussi re, Y. Cabidoche, P. Cellier, P. Debaeke, J. Gaudill re, C. H nault, F. Maraux, B. Seguin, and H. Sinoquet, "An overview of the crop model stics," *European Journal of Agronomy*, vol. 18, no. 3, pp. 309–332, 2003. Modelling Cropping Systems: Science, Software and Applications. 93
- [164] Y. Guo, Y. Ma, Z. Zhan, B. Li, M. Dingkuhn, D. Luquet, and P. De Reffye, "Parameter optimization and field validation of the functional-structural model greenlab for maize," *Annals of Botany*, vol. 97, no. 2, pp. 217–230, 2006. 93
- [165] H. P. Yan, M. Z. Kang, P. de Reffye, and M. Dingkuhn, "A dynamic, architectural plant model simulating resource-dependent growth," *Annals of Botany*, vol. 93, no. 5, pp. 591–602, 2004. 93
- [166] C. FOURNIER and B. ANDRIEU, "A 3d architectural and process-based model of maize development," *Annals of Botany*, vol. 81, no. 2, pp. 233–250, 1998. 93
- [167] C. Godin and H. Sinoquet, "Functional-structural plant modelling," *New Phytologist*, vol. 166, no. 3, pp. 705–708, 2005. 93
- [168] D. Leitner, S. Klepsch, G. Bodner, and A. Schnepf, "A dynamic root system growth model based on l-systems," *Plant and Soil*, vol. 332, no. 1, pp. 177–192, 2010. 93, 95
- [169] J. A. Postma, C. Kuppe, M. R. Owen, N. Mellor, M. Griffiths, M. J. Bennett, J. P. Lynch, and M. Watt, "Opensimroot: widening the scope and application of root architectural models," *New Phytologist*, vol. 215, no. 3, pp. 1274–1286, 2017. 94, 95
- [170] C. Granier and F. Tardieu, "Is thermal time adequate for expressing the effects of temperature on sunflower leaf development?," *Plant, Cell & Environment*, vol. 21, no. 7, pp. 695–703, 1998. 94
- [171] O. K rner and H. Challa, "Process-based humidity control regime for greenhouse crops," *Computers and Electronics in Agriculture*, vol. 39, no. 3, pp. 173–192, 2003. 94
- [172] P. Berry, M. Sterling, J. Spink, C. Baker, R. Sylvester-Bradley, S. Mooney, A. Tams, and A. Ennos, "Understanding and reducing lodging in cereals," *Advances in Agronomy*, vol. 84, pp. 217–271, 2004. 94
- [173] D. Causton and J. Venus, *The Biometry of Plant Growth*. Edward Arnold, 1981. 94
- [174] R. Hunt, *Plant growth curves. The functional approach to plant growth analysis*. London: Edward Arnold Ltd., 1982. 94
- [175] C. P. Winsor, "The gompertz curve as a growth curve," *Proceedings of the National Academy of Sciences*, vol. 18, no. 1, pp. 1–8, 1932. 94
- [176] E. Tj rve and K. M. Tj rve, "A unified approach to the richards-model family for use in growth analyses: Why we need only two model forms," *Journal of Theoretical Biology*, vol. 267, no. 3, pp. 417–425, 2010. 94
- [177] F. J. RICHARDS, "A flexible growth function for empirical use," *Journal of Experimental Botany*, vol. 10, pp. 290–301, 06 1959. 94

- [178] B. Mouliá and M. Fournier, "The power and control of gravitropic movements in plants: a biomechanical and systems biology view," *Journal of Experimental Botany*, vol. 60, no. 2, pp. 461–486, 2009. 94
- [179] M. R. Golzarian, R. A. Frick, K. Rajendran, B. Berger, S. Roy, M. Tester, and D. S. Lun, "Accurate inference of shoot biomass from high-throughput images of cereal plants," *Plant Methods*, vol. 7, no. 1, p. 2, 2011. 94
- [180] O. Tackenberg, "A new method for non-destructive measurement of biomass, growth rates, vertical biomass distribution and dry matter content based on digital image analysis," *Annals of Botany*, vol. 99, pp. 777–783, 03 2007. 94
- [181] J.-M. Pape and C. Klukas, "Utilizing machine learning approaches to improve the prediction of leaf counts and individual leaf segmentation of rosette plant images," in *Proceedings of the Computer Vision Problems in Plant Phenotyping (CVPPP)* (S. A. Tsaftaris, H. Scharf, and T. Pridmore, eds.), pp. 3.1–3.12, BMVA Press, 2015. 94
- [182] F. Lopez, P. Moghaddam, D. J. McCarthy, G. Valls, H. T. Nguyen, and A. C. Smith, "Deep plant phenomics: A deep learning platform for complex plant phenotyping tasks," *Frontiers in Plant Science*, vol. 8, p. 1190, 2017. 94
- [183] S. Samiei, P. Rasti, J. Ly Vu, J. Buitink, and D. Rousseau, "Deep learning-based detection of seedling development," *Plant Methods*, vol. 16, no. 1, p. 103, 2020. 94, 95
- [184] M. P. Pound, A. P. French, J. A. Atkinson, D. M. Wells, M. J. Bennett, and T. Pridmore, "Rootnav: Navigating images of complex root architectures , *Plant Physiology*, vol. 162, pp. 1802–1814, 06 2013. 95
- [185] G. E. C. Gall, T. D. Pereira, A. Jordan, and Y. Meroz, "Fast estimation of plant growth dynamics using deep neural networks," *Plant Methods*, vol. 18, no. 1, p. 21, 2022. 95
- [186] Y. Zuo, B. Bai, Y. Wang, and X. Peng, "A review of plant phenotype research based on 3d point cloud technology," in *2024 IEEE 6th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, vol. 6, pp. 1803–1807, 2024. 95
- [187] A. Mahlein, M. Kuska, S. Thomas, D. Bohnenkamp, E. Alisaac, J. Behmann, M. Wahabzada, and K. Kersting, "Plant disease detection by hyperspectral imaging: from the lab to the field," *Advances in Animal Biosciences*, vol. 8, no. 2, pp. 238–243, 2017. 95
- [188] R. Bhatt, S. Bansal, A. Chander, R. Kaur, M. Singh, M. Kankanhalli, A. E. Saddik, and M. K. Saini, "Gromo: Plant growth modeling with multiview images," 2025. 95
- [189] A. Olafenwa, "Simplifying object segmentation with pixellib library." <https://github.com/ayoolafenwa/PixelLib>, 2020. 96
- [190] P. D. Team, "Plantcv: rgb2lab function documentation." <https://plantcv.readthedocs.io/en/stable/rgb2lab/>. Accessed: 2025-08-30. 98, 101
- [191] G. García-Mateos, J. Hernández-Hernández, D. Escarabajal-Henarejos, S. Jaén-Terrones, and J. Molina-Martínez, "Study and comparison of color models for automatic image analysis in irrigation management applications," *Agricultural Water Management*, vol. 151, pp. 158–166, 2015. New proposals in the automation and remote control of water management in agriculture: agromotic systems. 100
- [192] Wikipedia contributors, "Cielab color space." https://en.wikipedia.org/wiki/CIELAB_color_space, 2025. Accessed: 2025-08-30. 101
- [193] D. Merkel, "Docker: lightweight linux containers for consistent development and deployment," *Linux J.*, vol. 2014, Mar. 2014. 111
- [194] U.S. Environmental Protection Agency, "ph and water quality," 2023. EPA Water Quality Parameters Overview. 124

- [195] R. V. Tyson, D. D. Treadwell, and E. H. Simonne, "Opportunities and challenges to sustainability in aquaponic systems," *HortTechnology*, vol. 21, pp. 6–13, Feb. 2011. 124
- [196] I. Lohraseb, N. C. Collins, and B. Parent, "Diverging temperature responses of co₂ assimilation and plant development explain the overall effect of temperature on biomass accumulation in wheat leaves and grains," *AoB PLANTS*, vol. 9, p. plw092, 01 2017. 124
- [197] A. Sharifi, M. Tarif, S. Migliorini, D. Quaglia, and R. Pastres, "Longitudinal image-based phenotyping and water quality measurements for salicornia growth in a marine aquaponic system," 2026. Dataset. 125
- [198] F. Hosoi, K. Nakabayashi, and K. Omasa, "3-d modeling of tomato canopies using a high-resolution portable scanning lidar for extracting structural information," *Sensors*, vol. 11, no. 2, pp. 2166–2174, 2011. 125
- [199] T. T. Nguyen, D. C. Slaughter, J. N. Maloof, and N. Sinha, "Plant phenotyping using multi-view stereo vision with structured lights," in *Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping* (J. Valasek and J. A. Thomasson, eds.), vol. 9866, p. 986608, International Society for Optics and Photonics, SPIE, 2016. 125
- [200] G. Bodner, A. Nakhforoosh, T. Arnold, and D. Leitner, "Hyperspectral imaging: a novel approach for plant root phenotyping," *Plant Methods*, vol. 14, no. 1, p. 84, 2018. 125
- [201] N. Fahlgren, M. Feldman, M. Gehan, M. Wilson, C. Shyu, D. Bryant, S. Hill, C. McEntee, S. Warnasooriya, I. Kumar, T. Ficor, S. Turnipseed, K. Gilbert, T. Brutnell, J. Carrington, T. Mockler, and I. Baxter, "A versatile phenotyping system and analytics platform reveals diverse temporal responses to water availability in setaria," *Molecular Plant*, vol. 8, no. 10, pp. 1520–1535, 2015. 125
- [202] H. Motulsky and A. Christopoulos, *Fitting Models to Biological Data Using Linear and Nonlinear Regression: A practical guide to curve fitting*. Oxford University Press, 05 2004. 126
- [203] X. Jiao, H. C. Zhang, J. Zheng, Y. Yin, G. Wang, Y. Chen, J. Yu, and Y. Ge, "Comparative analysis of nonlinear growth curve models for arabidopsis thaliana rosette leaves," *Acta Physiologiae Plantarum*, vol. 40, 06 2018. 126
- [204] S. V. Archontoulis and F. E. Miguez, "Nonlinear regression models and applications in agricultural research," *Agronomy Journal*, vol. 107, no. 2, pp. 786–798, 2015. 126, 150
- [205] A. Werker and K. Jaggard, "Modelling asymmetrical growth curves that rise and then fall: Applications to foliage dynamics of sugar beet (*beta vulgaris*l.)," *Annals of Botany*, vol. 79, no. 6, pp. 657–665, 1997. 126
- [206] I. Hajirad, K. Ahmadaali, and A. Liaghat, "Crop yield and water productivity modeling using nonlinear growth functions," *Scientific Reports*, vol. 15, no. 1, p. 30087, 2025. 127
- [207] C. E. T. Paine, T. R. Marthews, D. R. Vogt, D. Purves, M. Rees, A. Hector, and L. A. Turnbull, "How to fit nonlinear plant growth models and calculate growth rates: an update for ecologists," *Methods in Ecology and Evolution*, vol. 3, no. 2, pp. 245–256, 2012. 127
- [208] F. Carini, A. Cargnelutti Filho, R. Vieira Pezzini, J. Maronez de Souza, G. Chaves, and A. Provedi, "Nonlinear models for describing lettuce growth in autumn-winter," *Ciência Rural*, vol. 50, 06 2020. 128
- [209] J. H. Friedman, "Greedy function approximation: A gradient boosting machine.,," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189 – 1232, 2001. 134
- [210] M. Zambudio Martínez, L. H. M. d. Silveira, R. Marin-Perez, and A. F. S. Gomez, "Development and comparison of artificial neural networks and gradient boosting regressors for predicting topsoil moisture using forecast data," *AI*, vol. 6, no. 2, 2025. 134
- [211] T. Alahmad, M. Neményi, and A. Nyéki, "Soil moisture content prediction using gradient boosting regressor (gbr) model: Soil-specific modeling with five depths," *Applied Sciences*, vol. 15, no. 11, 2025. 134

- [212] T. Kawano, N. Wallbridge, and C. Plummer, "Logistic models for simulating the growth of plants by defining the maximum plant size as the limit of information flow," *Plant Signaling & Behavior*, vol. 15, no. 2, p. 1709718, 2020. 147, 148
- [213] K. M. C. Tjørve and E. Tjørve, "The use of gompertz models in growth analyses, and new gompertz-model approach: An addition to the unified-richards family," *PLOS ONE*, vol. 12, no. 6, p. e0178691, 2017. 148
- [214] B. Li, S. Zhang, W. Li, and Y. Zhang, "Application progress of grey model technology in agricultural science," *Grey Systems: Theory and Application*, vol. 12, 08 2022. 148
- [215] Y. Dang, X. Ning, D. Chen, and J. Cui, "The optimization of grey verhulst model and its application," *Journal of Grey System*, vol. 27, pp. 1–12, 01 2015. 148