Digital Twins for Hydroponic Lettuce Growth

Morgan Mayborne

CMU-RI-TR-25-69

July 30, 2025

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Prof. George Kantor, *chair*
Prof. Abhisesh Silwal
Prof. Illah Nourbakhsh
Dominic Guri

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

To my parents, who established in me a lifelong curiosity, and to the many teachers that inspired my journey

Abstract

With current population trends, concerns have been raised about the capability of global food networks for coming demand, especially in high-density urban centers. Alternatives to soil-based horticulture, such as controlled environment agriculture and hydroponics, have been developed and optimized for large-scale facilities. While large facilities can supply broad markets, small-scale ventures could support specific communities and offer foods not typically produced at larger scales. Especially in schools or urban agricultural settings, automating the growth process could provide easier access to food and educational opportunities. While recent research has focused on large-scale facilities, this research provides improved automation and growth information for operators in smaller facilities.

The concept of digital twins was promising for handling biological complexities of plant growth. Modelling growth without a strict initial condition is difficult, but digital twins use a consistent stream of measured information to build a better understanding of the system. To measure growth, the measurement stream included light intensity, temperature, carbon dioxide and biomass. Initially, a dataset was collected of 1305 plant measurements, pairing RGB-D images with a mass measurement. Biomass was estimated using a custom convolutional neural network architecture that evaluated mass from an image within 1.56 g of the ground-truth. Finally, these measurements are passed to a model that defines growth, updating model parameters to reflect new information. This research evaluated possible models and chose a biological model, NiCoLet B3. After calibration, this model was able to project growth between one to four days into the future, maintaining around a 1.45 g forecasting error.

Acknowledgments

I would like to thank my advisor, Prof. George Kantor, and my supervisor, Prof. Abhisesh Silwal, for providing their time and advice. I deeply appreciate the structure they provided to my research, while allowing me to seek out what personally inspired me. I would also like to thank my other committee members, Prof. Illah Nourbakhsh and Dominic Guri, for their selfless devotion to my project. They both provided invaluable perspectives, grounding me in the significance and excitement of the work I was doing. I also express my deepest gratitude to every member of the Kantor Lab, who have given me invaluable feedback and support in my research.

I would like to thank my family for their support on this journey. I would like to thank my family back home in Milwaukee — Brook, Bret and Mason — for their undying support. Thank you for remaining patient with me through my academic path — unpredictable as it may have been. And I would like to thank to my life partner, Melissa, for continuing to pick me up whenever I need it.

Lastly, I would like to thank Carnegie Mellon University, the Robotics Institute, and all of my peers in the MSR program. My research would not have been possible without the inspiration from my classes and my peers.

Funding

This research was funded by the USDA-NIFA Climate-Smart Agriculture for Future Farms (CAFF) program, through the award 2023-68017-39402.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project Motivation

As food demand increases with global population growth, a substantial development in agricultural production has been controlled environment agriculture (CEA). CEA refers to new agricultural studies that seek to better understand what is necessary for healthy plant growth and how to optimize the local environment for improved production. The desire to optimize comes from concerns about how well traditional soil-based horticulture is suited to meet current and future demand [30]. Despite the strengths of soil-based horticulture and its widespread use for thousands of years, the main inefficiencies are land area inefficiency and the inability to control external conditions important to plant growth [26]. These two concerns have, respectively, spawned the fields of vertical farming, where plants are stacked to ensure more plants per land area, and indoor agriculture, where a closed space has controlled parameters to optimize growth.

Although designing new environments for CEA-based plant growth seems complex, greenhouses are ancient inventions and were a step toward controlled growth. The complexity arises when considering what factors contribute to growth, how to measure those factors and how to control them reliably. However, recent advances in plant science and sensor technology have significantly improved CEA's ability to monitor, regulate, and optimize food production [38]. In addition, soilless horticulture practices, such as hydroponics and aquaponics, have been important developments for CEA, as

new growth mediums are usually easier to measure and control than soil.

Many of these developments have started to demonstrably affect the global food supply, with indoor agriculture production increasing from 562 million pounds in 2009 to 880 in 2019 [8] for leafy green vegetables, supplying about 25% of the US demand [16]. Experts in the field believe that CEA producers could supply 50% of the US consumer demand for leafy greens by 2035 [26]. Internationally, the Netherlands has specifically increased funding for research and development in greenhouse technologies and precision agriculture, growing into a significant agricultural exporter on the global stage despite its small land area [34, 39].

1.1.1 Small Growers

Although the economic case for small producers building for-profit CEA facilities may not exist presently, many noncommercial applications for small, autonomous facilities can still be explored. This research is a part of the I2GROW project, a multiuniversity collaboration that seeks to build a low-cost growth environment for schools and urban neighborhood communities. Both communities benefit from the local food source, but a local autonomous greenhouse can provide additional advantages.

For school environments, a controlled agricultural environment can serve as an educational tool in various STEM disciplines. For students interested in plant science, growing plants could show students the different stages of growth from seedling to maturity, as well as different environmental variables and their effects on plant growth. For engineering students, a small CEA greenhouse could serve as a practical engineering project, giving students the ability to tinker with the platform and implement scientific hypotheses.

In urban communities, a controlled agricultural environment could work to advantageously repurpose local carbon dioxide pollution into a local food source. Since plant growth is carbon negative in aggregate and additional carbon dioxide can even improve growth rates, a carbon dioxide pump could be installed in a controlled environment to combat local carbon emissions.

1.2 Objective

This project seeks to bring the core principles of CEA technologies to small-scale growing facilities, especially with education and urban agriculture settings in mind. To achieve this, the project envisioned a greenhouse prototype to fulfill the following goals:

- Project future plant yield using measured data
- Automate measurement and operation processes, only requiring seedling installation and a single click to initialize 'smart growth' scripts
- Minimize prototype expense
- Communicate growth simply and concisely
- Provide a platform for future environmental condition controls

A robotic infrastructure, equipped with sensing and computer vision capabilities, was developed inside a custom greenhouse to automate environmental sensing and plant monitoring. This project hypothesized that plant monitoring would provide crucial information for forecasting future growth by updating simulations of individual plants, called digital twins. The status of these digital twins could then be used to communicate growth or optimize environmental conditions to maximize plant yield.

An overview of related works is discussed in Chapter 2. Chapter 3 discusses the different components of the digital twin structure, including a definition in this context, the design of the greenhouse hardware, and the software necessary for ongoing plant simulations. Chapter 4 introduces a collected custom data set, which is used to select optimal subcomponents and optimize the prediction accuracy of the network.

1.3 Thesis Contributions

We propose the following contributions.

- We define digital twins for a hydroponic production setting and produce a digital twin architecture, which simulates and forecasts future growth.
- We introduce a collected dataset of lettuce plant measurements, pairing RGB-D plant images with ground-truth mass and environmental condition measure-

ments.

- In order to measure mass during autonomous operation, we demonstrated an image analysis process for non-destructive mass evaluation, defining and training a CNN model to estimate biomass within 2 g of the actual mass.

Chapter 2

Background

2.1 Hydroponics

Hydroponics defines a broad category of soilless agricultural practices in which nutrients are provided to a plant through a water-based solution. Although different methods of hydroponics exist, the only requirement for the development of hydroponic plants is a substrate for plant seed and a circulation system to provide plants with a consistent source of nutrients. Compared to traditional horticultural practices, hydroponics has many advantages, including year-round growth in a controllable environment, land area efficiency with vertical stacking, and more efficient use of water resources. [56] Hydroponics has even been proposed as a method to combat the food demands of exponential population growth, as its advantages over soil-based horticulture are important in urban contexts. [32], [53] Today, hydroponics is a $5 billion industry and is expected to more than double in size by 2030, as food security systems respond to increasing demand. [1]

Although smart hydroponic facilities have been adopted in large commercial operations and hobbyist hydroponic products have been created for basic household use, small- and medium-scale growth opportunities have been limited by the high initial cost and skilled labor required to maintain hydroponic facilities [56]. In order for a smaller-scale hydroponic production facility to be commercially viable, further research needs to be done on smart sensing technologies, which themselves increase the initial cost. Recent works have aimed to capitalize on Internet of Things (IoT)

technologies to automate growth [44], [35], but there is still a research gap in combining automation with greenhouse environment optimization to ensure consistent and high-yield plant growth. Similar studies have been conducted to optimize greenhouse conditions for soil-based plants [4], but there is still a gap for smaller-scale hydroponic production.

2.2 Agricultural Simulation

Many different modeling techniques exist for plant growth in many different settings. Crop-scale models, such as CERES [57], exist for farmers who want to simulate aggregated plant growth throughout a field, based on field measurements. These crop-scale models can be simulated with open-source simulators like APSIM [13, 14], or DSSAT [19]. Other models focus on limited resources, such as water, and forecast growth in resource-limited environments, with examples like AquaCrop [41] and Irrigation Water Requirements (IWR) [45]. Other models focus their scope down to the individual plants and aim to quantify growth by tracking the value of an index over time. An example is the PROSAIL model [48], which measures the leaf area index of a plant and spectral lighting conditions to compute the canopy reflectance as a means of tracking growth.

Models for hydroponic production also exist, especially in the resource-limited and plant-scale modeling categories. A foundational model in the botanical literature was the TOMGRO model from 1991, which focused on modeling tomato growth as a function of lighting, carbon dioxide, and temperature [18]. This TOMGRO model is still used today for the growth of greenhouse tomatoes [43, 55]. Additionally, the HORTISIM model, which is used for greenhouse growth more generally, captures a lot of plants typically grown in hydroponic environments, like bell peppers and cucumbers. HORTISIM is also commonly used in conjunction with other greenhouse scripts, such as climate simulators, and run together to optimize a greenhouse to maximize a crop's yield [10, 24].

2.2.1 Related Work 1: NiCoLet Model

For lettuce modeling, the NiCoLet B3 model [20], which is an updated model from the original NiCoLet model by Linker et al. [31]. The model is a first-order differential equation approach to quantify the growth of lettuce plants, inspired by the approach taken in the TOMGRO model. The model aims to simulate a lettuce plant in a nitrate-limited environment by charting the carbon balance over time, given the input conditions of lighting, temperature, and carbon dioxide. Even without nitrate limitations, NiCoLEt is capable of modeling real-world plant growth in both soil-based and hydroponic facilities.

2.3 Digital Twins

A digital twin is a simulation defined by a mathematical model, or a "twin", that updates with real-time information from a physical and dynamic system. Often, these dynamic systems have inherent complexities or unknowns, embedded either in the dynamics of the physical system or in the observation of the physical system. At their core, digital twins require three elements: a physical system, a virtual model, and an update procedure [51]. Typical examples of digital twin systems are airplanes, where sensor packages may be limited for regulation compliance, and factory inventory, where the effect of a single broken machine might have a non-linear effect on production [61, 63].

2.3.1 Modeling

The model itself can take three forms, each of which has slightly different update procedures. A white-box approach describes a strictly defined mathematical model for the physical system, which updates system parameters deterministically to minimize model discrepancy. A gray-box approach might have a similar mathematical model, but includes a probabilistic update procedure that updates the probability distribution of a parameter as more measurements arrive [22]. Lastly, the black-box approach takes advantage of neural networks to encode complex relationships and uses incremental learning processes to continuously tune the model with additional data.

LSTMs, which are a form of recurrent neural networks (RNNs), are particularly useful as black-box approaches [24]. RNNs can pass persistent information between training iterations through a hidden state. Where basic RNNs have an impressive ability to recall information between short gaps in iteration, they still struggle to recall long-term dependencies, defined by a vanishing gradient problem. To address this issue, LSTMs improve the update procedure for the hidden state by including explicit gates for adding, updating, and removing information.

2.3.2 Agricultural Application

Digital twins have been applied to agricultural settings, since plant growth is an inherently complex and dynamic setting. A review of proposed digital twin architectures in agricultural settings lists a lot of non-horticulture or horticulture-supplementary digital twins present in recent literature, such as livestock management and meteorological climate analysis. This review singles out hydroponic plant growth as a particular research gap. This project will explore this research gap by building a digital twin architecture that pairs hydroponic growth with advances in modeling techniques [47].

2.3.3 Camera-based Phenotyping

Modern advances in camera technology have allowed for new plant phenotyping methods. Especially with the accessibility of stereo and LIDAR cameras, it is now possible to capture raw 3D models for individual plants, while software packages like SLAM can reconstruct high-fidelity models from a series of localized images. Given these high-fidelity models, research has been conducted to improve plant phenotyping with efficient, autonomous methods of evaluating a plant.

From reconstructed 3D point clouds, research has been done on the skeletonization of a point cloud, or simplifying the plant down to its branch structure [6, 59, 60]. For a single skeletonized point cloud, metrics of plant height and leaf characteristics can give a numerical perspective on the current state of the plant [59]. Other research pairs skeletonized point clouds over time, which is useful for defining growth or other transformations between temporal steps. [6]

Another type of post-processing on plant point clouds is volumetric analysis, fitting a convex shape around a plant. Although plant organs inside the convex shape may be obscured, the volumetric shape, paired with typical canopy density, can give a strong idea of the size and weight of the plant being measured [7, 40].

2.3.4 Related Work 2: Biomass Estimation

Lastly, multi-modal images, like RGB-D images fusing color and depth information, can be inputs into neural networks that seek to learn a specific phenotype. Neural networks, such as convolutional neural networks (CNNs) or graph neural networks (GNNs), could learn and automate the process of collecting a phenotype for a plant. A particular application is to estimate the biomass of a plant, which is typically a destructive measurement method.

Buxbaum et al. researched a method for using pre-trained classification backbones to improve the prediction accuracy of a neural network [5]. This approach feeds the depth and color channels through separate CNN backbones, concatenates the resulting latent vectors, and then creates a unified MLP network to predict the estimated biomass. This approach uses pre-trained classification networks as the CNN backbones, specifically the ResNet50, by removing their final classification layers. These networks are repurposed from their original classification application because of their deep networks, designed specifically for generalizable feature extraction and analysis.

Chapter 3

Methods

3.1 Digital Twins

3.1.1 Motivation

To observe or control a system, a model is required that uses mathematical functions to define interactions and relationships between system variables. Many models are reliant on system parameters to capture an intrinsic aspect of a system. For a model to adequately control a system, the system parameters must be calibrated to the physical system to minimize possible inconsistencies. Calibrating a system model requires measurements and can be as simple as researching universal constants, like gravitational acceleration. Complexity arises when a system is difficult to define, meaning models are difficult to construct, or when they are difficult to measure, meaning system parameters are difficult to calibrate.

Especially in the context of small growers, plant growth has complexity in both respects. Growth dynamics are physically complex, defined by molecular-scale processes that are difficult to model. The measurability of the plant adds additional complexity, as most yield measurements are limited by destructive processes. Since small growers do not often have the time or expertise to construct large datasets necessary to combat these complexities, they are limited in their ability to extrapolate future growth or change environmental conditions to optimize plant yield.

Digital twin systems are helpful for progressively building an understanding of a

system from an initial condition. Regardless of the validity of the initial condition, a consistent stream of measurements can help a model update the parameters of the system and improve its estimates. By itself, a digital twin requires a physically valid model, and it does not fix measurement complexities. However, once a model is obtained, the digital twin structure can be data efficient, updating a prior model definition to reflect the state of a physical system in real time.

3.1.2 Objective

The definition of a digital twin used in this research is a simulated model of a dynamic system paired with a consistent stream of measured data that updates the model to minimize modeling error. The measured data must be relevant to the system environment to properly tune the model. While relevant information is necessary, some digital twins are used when the full mathematical relationship between system behavior and measurable data is unknown, or when not all necessary data for deterministic model updates can be measured. Designing a digital twin requires a strong understanding of relevant measurements and their effects on the system, even when the system is underdefined by the given measurements. A general digital twin architecture, shown in Fig. X, requires three elements: a physical system to model and measure, a simulation, and a feedback structure for updating the simulation given new measurements.

This project seeks to design an architecture for digital twins that is compatible with the measurement capabilities and dynamics of plant growth. This will critically require a model that adequately defines the foundational biological processes that define growth. In addition, the digital twin architecture also requires methods for measuring plants, communicating data, and updating model parameters. Since a small growing facility allows for localized sensor measurements, this project aims to build a system that can construct digital twins for individual plants grown in an experimental greenhouse. With a live feed of localized measurements, the digital twins will update their understanding of the growth of the plant, while also providing metrics that forecast future growth.

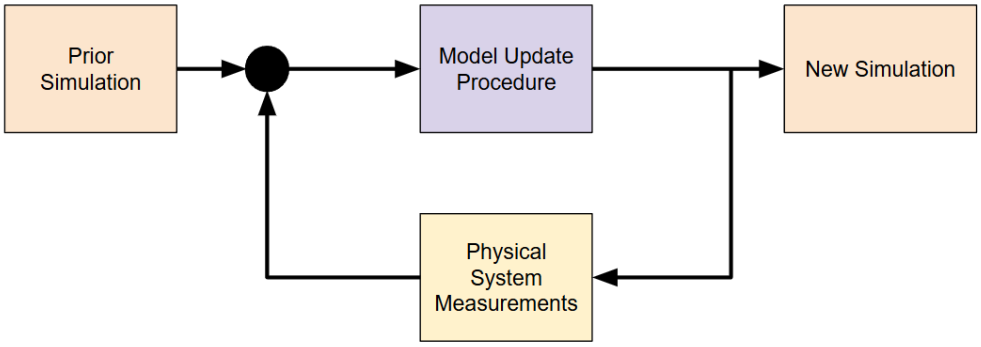

Figure 3.1: This plot shows the general architecture of a digital twin. The feedback loop of a digital twin comes from the recorded measurements updating the model's understanding of the system. The continuous feedback process should help the simulation converge to real-world behavior.

3.1.3 Growth Metric

To create digital twins in this context, a growth metric must be defined. Because digital twins are required to estimate a real system in real time, digital twins require metrics that are able to be measured and simulated. The metric was also selected to ensure accessibility in educational settings, allowing comprehension without requiring a formal scientific background.

For these reasons, fresh biomass, or the wet weight of the plant, was selected, as it was considered intuitive and was shared in common with many plant models in the literature. Fresh biomass excludes the weight of the saturated substrate used for hydroponic planting and also includes the weight of the water of the plant. Dry biomass, a more robust metric of organic matter, was not feasible for direct measurement, as the removal of water weight requires a destructive process of drying out the plant in an oven. With fresh biomass as the core growth metric, current growth will be defined by the time series of plant biomass, while models will be used to extrapolate future biomass from current growth.

3.1.4 Necessary Components

The rest of this chapter will discuss the intermediary systems necessary for the digital twin architecture. Figure 3.2 below shows a high-level pipeline for the digital twin used in this research, from the raw measured data to the updated model simulation. First, the experimental greenhouse will be specified in further detail in Section 3.2, defining the hardware, measurement processes, and communication protocols with the digital twin. In order to measure biomass in the live system to track the modeling error, a specific transformation experiment will be conducted and described in Section 3.3. Lastly, the model selection and update procedures will be discussed in Section 3.4, and an experiment will be outlined to compare different approaches to modeling and forecasting plant growth.

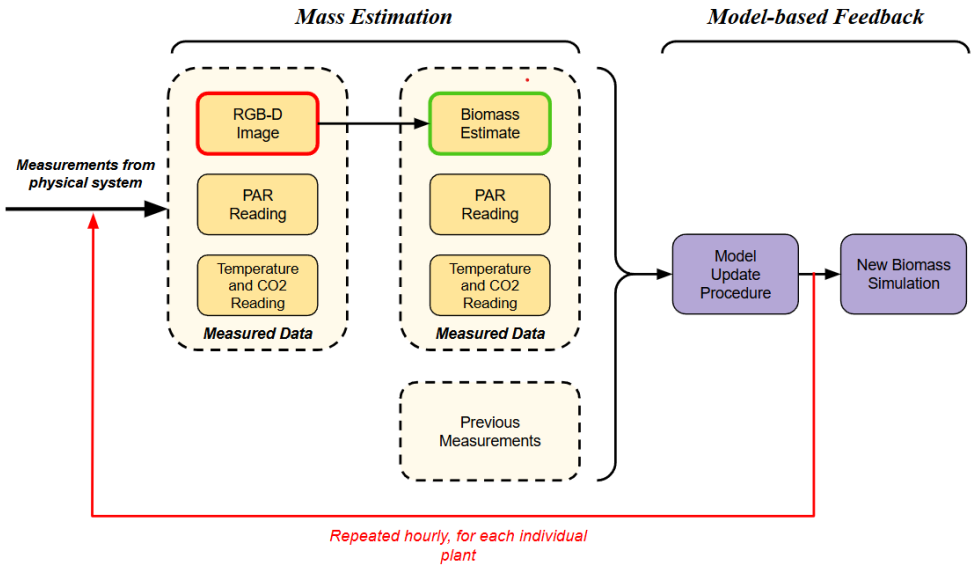

Figure 3.2: This shows a more detailed view of the digital twin feedback process in this research. 'Mass Evaluation' is highlighted a transformation is necessary to turn RGB-D images into a biomass estimate. 'Model-based Feedback' describes the final process of updating the model with new information and forecasting future growth.

3.2 Greenhouse Design

3.2.1 Motivation

To enable the digital twin experiments, a greenhouse and a computer were required to grow plants, measure growth, and communicate measurements. Additional goals for the project were to minimize equipment cost and automate daily operations to increase accessibility for growers in schools or urban agricultural settings. Lastly, the project also wanted to produce a visual interface to communicate the status of plants inside the greenhouse, using the digital twin to provide estimates of measured and projected growth.

To achieve the goals above, a custom greenhouse was developed for hydroponic plant growth, and a central computer was selected for this research, an Alienware laptop with an Intel i7 CPU and an Ubuntu 22.04 operating system, to run custom Python automation scripts. Both of these systems provided the necessary coordination to operate the robotic system autonomously and run digital twin simulations. The software architecture in Figure 3.3 displays the greater system software architecture, which will be expanded in the following sections.

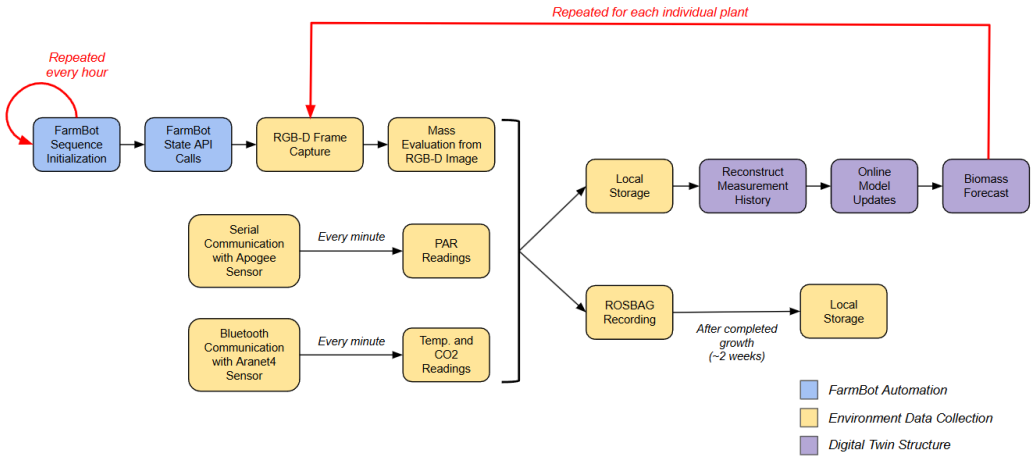

Figure 3.3: The software architecture developed in this project to support digital twin experimentation consists of three main components: automation of a gantry robot, data acquisition for digital twin analysis, and mechanisms for real-time updates and forecasting within the digital twin model.

3.2.2 Hardware and Logistics

To allow plant growth, a custom greenhouse, equipped for hydroponic plant growth, was built at Carnegie Mellon University in the fall semester of 2023. This hydroponic environment was built to grow 24 plants simultaneously and provide each of these plants with suitable air circulation and lighting conditions. The Nutrient Film Technique (NFT) was selected as the hydroponic method because of its simple hardware requirements and low cost. The greenhouse with growing plants is shown in Figure 3.4.

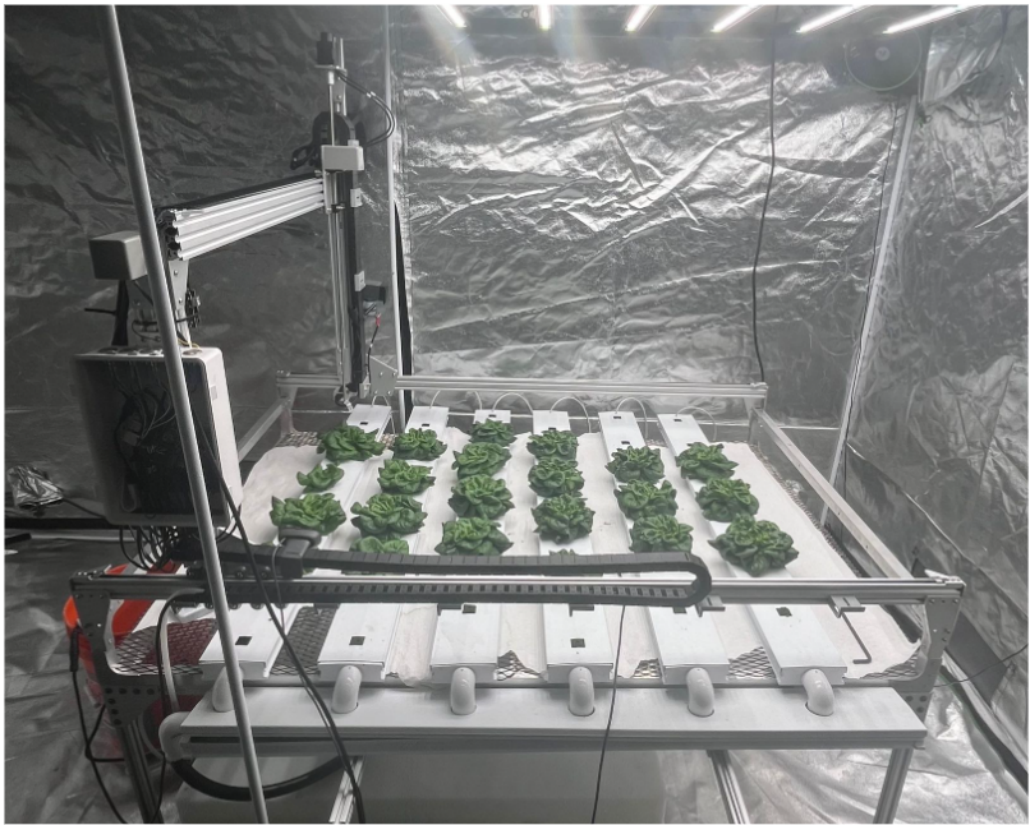

Figure 3.4: The image shows inside the custom greenhouse developed for this research.

A planting bed for the NFT system was created by forming a frame with 80-20 extruded aluminum with adjustable legs and an aluminum mesh fastened to the top face of the frame. The bed had a width of 80 cm and a length of 120 cm. NFT channels, rectangular tubing that allowed the flow of nutrient solution, are placed on

the aluminum mesh.

For the circulation of the nutrient solution, a tank is placed under the frame to store the nutrient solution, and a 370 GPH, 24 W submersible pump is placed inside. This pump is connected to six branching tubes, each equipped with a valve to regulate the flow of the nutrient solution for each of the NFT channels. The frame legs are adjusted higher on the far side of the frame to allow fluid to flow from the top of the channel to the bottom under the influence of gravity. Once it reaches the bottom of the NFT channels, the tubing returns the remaining nutrient solution to the tank.

To provide lighting for the plants, a VivoSun LED grow light was installed that could provide the necessary 200-400 $\mu\frac{mol}{m^2 s}$ of photosynthetically active radiation (PAR), which is a measure of the amount of light within a specific spectral band. The particular grow light chosen had LED lights and an application that could change the spectral composition of the lighting. To find the ideal distance between the plant bed and the lighting fixture, an initial estimate was calculated from the manufacturer's technical flux specifications, while a handheld Apogee MQ-500 Quantum PAR sensor was used to make finer adjustments after installation.

To provide adequate air circulation for the plants, fans were installed in two accessible ports in the growth tent to push external air inside the growth tent. Since the lighting unit let off a significant amount of heat, the external air was cooler than the air over the plant canopy. The fans were angled downward to direct air flow to the canopy of the plants, replacing hotter air and cooling through convection.

The process of initializing seedlings started with placing lettuce seeds in a commercial growing medium, called Oasis cubes, which were initially fully saturated with water. To grow seedlings, the Oasis cubes were placed on a tray with half an inch of nutrient solution, balanced to have an electrical conductivity in the range of 900-1100 S/cm and a pH between 5.8-6.0. This tray was placed in a second enclosed environment, shown in Figure 3.5, with a light source, which was on for 14 hours daily, and maintained a temperature of 24 degrees Celsius and 500 parts per million (ppm) of carbon dioxide. The seedlings were grown in this environment for 7-10 days, after which they were moved to the larger custom NFT environment.

Before migrating the seedlings to the custom greenhouse, the nutrient solution tank was filled with a new nutrient solution for circulation. This nutrient solution was balanced to have an electrical conductivity between 1300-1700 µS/cm and a pH

Figure 3.5: The left image shows the size and shape of the second growth environment. The second image shows inside the tent, namely, the tray for initial seedling growth.

between 5.8-6.0. At this point, the seedlings were migrated into holes in the NFT channels. Once migrated, the seedlings were able to grow to maturity without direct maintenance.

A hardware part list is described in further detail in Appendix A.2.

3.2.3 Data Measurement

Next, the project considered which measurements would best describe plant growth and environmental conditions for the update procedure of the digital twin model. Since a goal of this project was to minimize costs, this research aimed to limit the total quantity of sensors. To define the environment around the plants, many plant models [18, 20] focus on temperature, carbon dioxide, and lighting intensity as critical environmental factors, so these were considered to be the project's description of the growth environment. Second, to compute the modeling error, the estimated biomass was also measured and compared with the digital twin simulation. Estimating biomass in an autonomous environment was a difficult task, but recent literature suggested

that it was possible to estimate with a color image and a depth map of a plant. For the measurements above, a suite of three sensors was selected: a stereo camera, a multi-parameter environmental reading sensor, and a PAR sensor. These sensors provided access to the necessary readings while minimizing the need for additional sensor infrastructure.

The stereo camera used was the Intel RealSense D405, a low-cost stereo camera with strong documentation. Despite the lower cost of this device compared to other models in the D400 series, this camera met the requirements for a prototype for this project, which was producing a depth map for objects in close proximity. Additionally, the Intel RealSense D400 series shares a common package for camera communication, so scripts written for this camera will generalize to the entire D400 series of depth cameras. To use the stereo camera, the available PyPi packages for the camera enabled the development of a custom module that sent images through a wired USB connection to an external computer. The stereo camera was then mounted on the FarmBot end effector, using a custom 3D printed mounting adaptor that attached to the end effector. The USB cable was wired through the flexible cable tracks on the FarmBot to avoid interference during operation. Every hour for each plant, an RGB-D image is post-processed by the stereo camera and sent to the central computer where it is saved locally, labelled with a timestamp.

Temperature and carbon dioxide were collected through a wireless SAF Aranet4 Home sensor. Through an open-source PyPi package, the sensor was also able to communicate with an external computer over Bluetooth. While Bluetooth capability made the sensor more expensive than comparable sensors, this wireless communication was justified by the minimization of wired connections and greater flexibility in sensor placement. Every minute, the central computer requests and saves the environmental data from the Aranet4 sensor.

Lighting measurements were performed through an Apogee Quantum SQ420 PAR sensor, which was able to communicate to an external computer via a wired USB connection. Although sensor readings are typically communicated through proprietary software with an expensive license, available Python documentation from the manufacturer allowed this project to develop a custom script for serial communication of sensor readings. Inside this growth tent, the sensor was securely fastened to the plant bed, and the wired USB cable was routed underneath the plant

bed to the central computer. Since the PAR sensor and the corresponding plant location are fixed, the lighting conditions were extrapolated to the coordinates of each plant, using manual PAR measurements to initially fit a 2-D curve to the lighting conditions in the plant bed. Every minute, the central computer requests and saves lighting measurements from the PAR sensor.

In order for the digital twin to read the history of measurements, the project chose to save information locally. The required information to save locally was the environmental conditions and the RGB-D images. Local saving relied on standard methods for reading and writing files to ensure reliable data access during and after the program is executed. RGB-D images were exported from NumPy arrays into NPY files into a local directory and were about 2.4 MB on average. Environmental measurements were appended to a local CSV file, with individual lines containing a timestamp and the corresponding sensor readings.

3.2.4 Automation Hardware

To enable autonomous measurements for biomass readings, the camera needs to read an overhead perspective of each plant at a fixed interval. To achieve this, an off-the-shelf gantry robot, the FarmBot Genesis v1.7, was placed on top of the planting bed. FarmBot is an open-source robotics platform for gardening and autonomous scheduling capabilities, and this project chose FarmBot for its accessible software and customizability. In particular, this FarmBot gantry has a customizable end effector to enable external sensing capabilities, as well as three controllable degrees of freedom for simple access to all plants on the bed.

The FarmBot is composed of a frame, a chassis, and an end effector, all shown in Figure 3.6. The frame surrounds the entire plant bed and contains a lateral actuation system, which can move the chassis from left to right relative to the frame. This lateral actuation is a belt-driven linear actuation system with hard stops at the frame limits. The chassis contains the central computer, which is a Raspberry Pi 4 with a custom operating system and shown in red in the center image in Figure 3.6, and contains two actuation methods that can move the end effector in longitudinally or up/down relative to the chassis. Longitudinal actuation, shown in red in the right image in Figure 3.6, is similarly belt driven with hard rails at either end of the

bed, while Z-direction actuation, also shown in blue in the right image in Figure 3.6, is instead an overhead stepper motor driving a lead screw. Although Z-direction actuation also has a hard stop, the hard stop is adjustable to manually set a safe operation height. Importantly, each of the actuating motors has incremental encoders, which are translated into position estimates through FarmBot for each axis.

Lastly, external cabling is required for internet access, through an Ethernet cord to the Raspberry Pi, and connection to an external power supply. Additional wires necessary for external sensors can be run through flexible cable tracks for safe wiring without pinching cables or impeding motion.

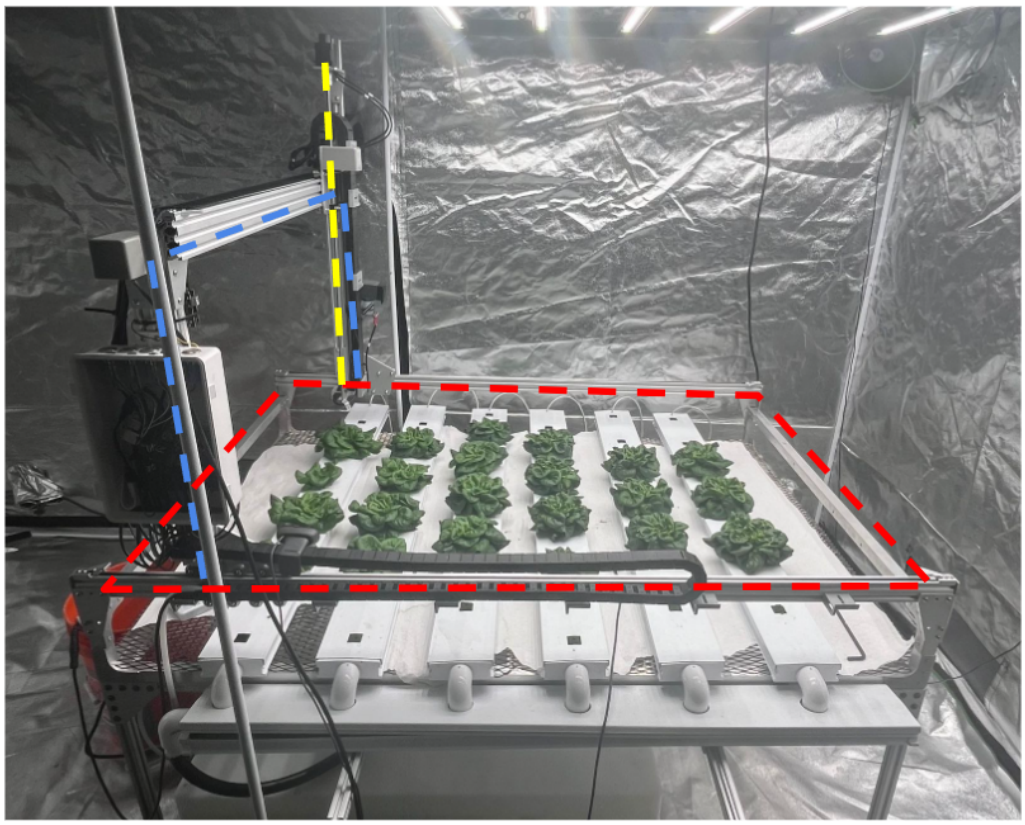

Figure 3.6: This image labels the major components of the FarmBot. The frame is outlined and red and contains the rail for X-actuation. The chassis is labelled in blue and contains the FarmBot computer and Y- and Z- Z-actuation systems. The end effector, labelled in yellow, extends out from the chassis.

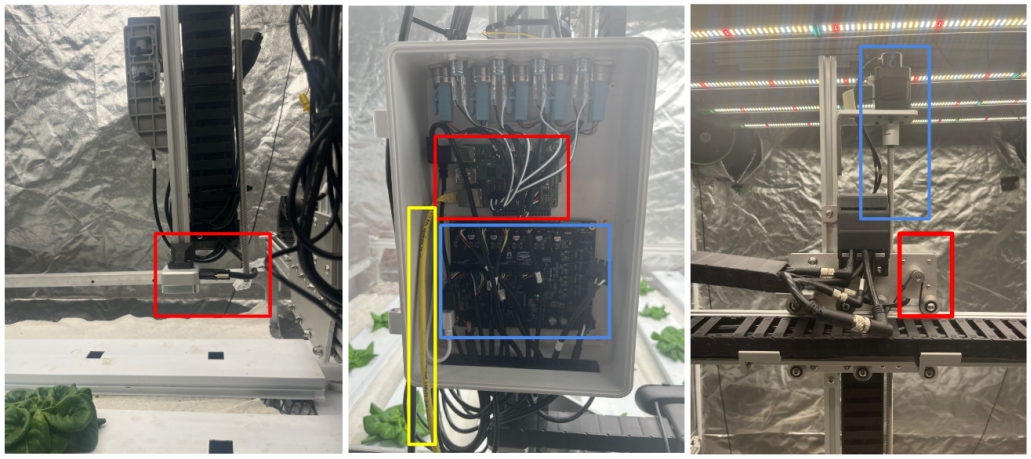

Figure 3.7: The left image shows the end effector of the FarmBot. The middle shows the FarmBot computer (red), an Ethernet connection (yellow), and the motor controller board (blue). The right image shows the belt-driven Y-actuation system (red) and the lead-screw Z-actuation system (blue).

3.2.5 Automation Scheduling

To coordinate hourly plant imaging, a FarmBot scheduling script needed to be capable of visiting each plant and taking an RGB-D image at a constant interval, allowing the mass evaluation module to estimate growth at a constant frequency. This task required multi-threading coordination between a FarmBot controller and a stereo camera controller.

Although FarmBot software is open source, it does require communication with a cloud-based web application to initiate actions. To request action on the gantry, a request is sent as an API request through the cloud infrastructure, which then responds by initiating the action from the FarmBot's Raspberry Pi. To coordinate sequences of actions, sequences can be saved in a user's account as a custom-formatted JSON file, which can be generated through a block programming interface on the FarmBot website. For this research, a 'raster scan' sequence was saved, traveling along each of the NFT channels and waiting at each plant for an image to be taken.

These initial steps simplified the FarmBot initialization process in the web interface, but additional steps were required to enable full control from an external computer. FarmBot supports API requests for initiating action sequences and reading state information for the robot, which can be done through a Python script connected

to a FarmBot account. A custom script was developed to automate these sequence initialization commands at a user-defined time interval.

In order to collect RGB-D images at each plant location, the Python control script sets up a secondary thread that sends API calls to receive the current state of the robot, particularly the coordinate position of the robot. If the robot is waiting at the hard-coded locations of a plant, a Python module for stereo camera control, customized from Intel RealSense's documentation files, is executed that captures an RGB-D frame.

To support the project's goal of simple initialization, ROS 2 middleware was chosen to manage script execution and data storage. Specifically for this project, the Humble Hawksbill ROS2 release was used with the Ubuntu 22.04 operating system of the central computer, but the custom ROS package developed for this project was written to generalize to other ROS 2 releases. Two ROS packages were written for this project. The main package was developed in Python to run all the custom programs simultaneously and record the measured information in a backup ROSBAG. The second package was written in C++ to define custom message types that are used to publish ROSBAG information. The automation scripts were initialized by executing a ROS 2 launch function, which executed all relevant scripts and started a ROSBAG recording.

Additional items were scheduled externally due to product limitations. The VivoSun software application was used to automate the lighting fixture to be on for 14 hours and turn off for the remaining 10 hours, matching the ideal diurnal cycle for plant growth [33]. The spectral composition was chosen as VivoSun's default 'vegetative' spectrum, which included strong concentrations of blue and red light. The VivoSun fans was also automated through the VivoSun application, scheduling fan circulation for 50 minutes of every hour while the lighting system was on.

3.2.6 Visual Interface

In parallel with the scheduling and data collection scripts, the central computer displays a visual interface. Since the greenhouse is sealed off for ventilation and lighting, it was necessary to develop a visual interface to communicate the current status of the plants, the status of the robot, and the greenhouse environmental

conditions. For these reasons, the project opted to show four items to communicate these objectives: (1) a mosaic of the most recent plant images, (2) a video feed directed at the growing plants, (3) a 2-D grid displaying both robot and plant positions, and (4) a recent history of environmental condition measurements. An example of the visual interface is shown in Figure 3.8.

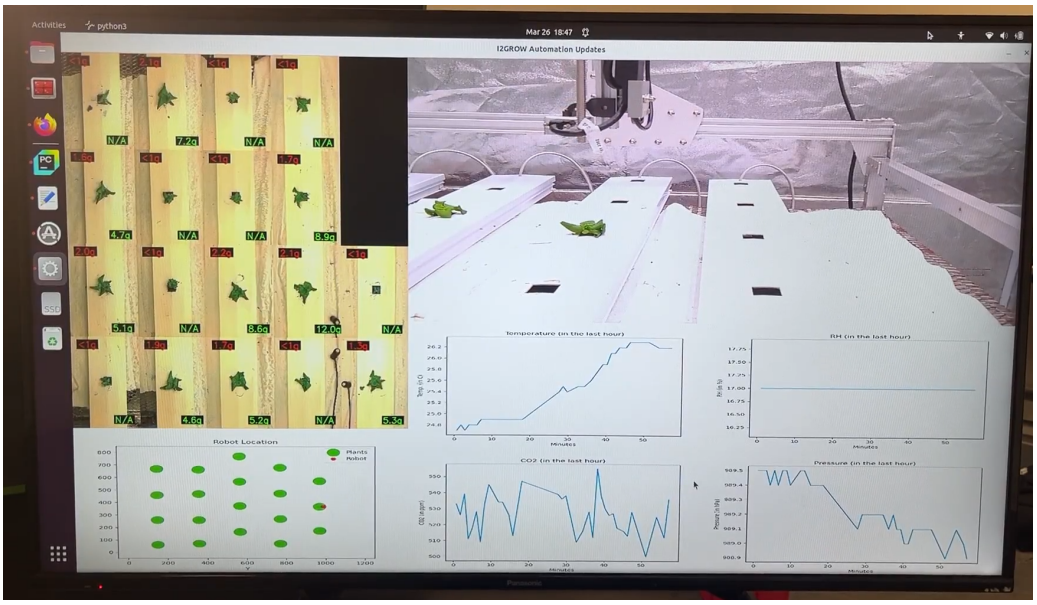

Figure 3.8: The visual interface for the custom greenhouse. The top left shows an image mosaic for all measured plants in the greenhouse. The top right image shows a live video feed inside the tent. The bottom right shows a 2-D coordinate frame with plant locations and current robot position. The bottom right image shows a measurement history of relevant data for the digital twin.

The critical component of the visual interface is the mosaic, the recent set of plant images alongside quick summary metrics. To intuitively organize the mosaic, the images are combined into a grid-like format, with the image locations on the mosaic corresponding to their relative locations on the plant bed. Each image shows the most recent color image taken by the stereo camera, as well as two metrics to communicate the growth of the plant. Shown in the image mosaic on the top left of Figure 3.8, the red number above each plant image displays the estimated current mass of the plant, derived from the mass estimation module. The green metric shows its predicted mass in 4 days, estimated by the digital twin infrastructure from the measurement history of the plant. These metrics aim to concisely inform the user of the current and future

yield of the plants inside. Lastly, these numbers are updated as new images arrive, providing a simple representation of digital twin forecast updates.

The last three components attempt to visualize the current operations and conditions inside the growth tent. First, to visually locate the robot, the end effector position is displayed on a two-dimensional grid, which also shows the different plant positions. Second, a live video is placed on a rail of the robot, which moves along the FarmBot chassis and provides another visual perspective on the plants growing inside. Lastly, recent environmental conditions are displayed with a line graph that shows how temperature, pressure, carbon dioxide, and relative humidity have changed in the last hour.

3.3 Mass Evaluation

3.3.1 Motivation

The goal of the digital twin is to minimize the modeling error given the estimated biomass and the measured environmental conditions. As mentioned in the previous section, it is necessary to estimate the current biomass of a plant to calculate the modeling error between the physical and the simulated growth. Although mass is simple to measure by hand, measuring plant biomass is a complex challenge under the cost and autonomy constraints of this project.

An initial idea might be to place load cells, an inexpensive mass measurement sensor, throughout the greenhouse to determine the mass of a specific plant. While it would be possible to measure each plant's weight with load cells, building a large load cell array that is wired over the large plant bed to reach the base of every plant would be complex, expensive and likely not robust to environmental changes. Alternatively, picking up plants and measuring them with a hanging scale or placing them on a commercial scale might be considered. Without breaking autonomy requirements, this would require a complex pick-and-place machine to enable repeatable measurements, but such a machine would likely not fit any reasonable definition of 'low cost.'

Another option seemed to be image analysis, specifically image regression on plant images. Cameras seem to fit well with project requirements as they tend to be less expensive, robust to environmental conditions, and simple to automate. A downside

relative to the previous options is storage requirements, but the storage required for hourly images for an entire plant growth cycle did not exceed 2 GB, a standard amount found on cheap flash drives. Image analysis was ultimately chosen as the ideal method for evaluating plant biomass for this project due to its cost-effectiveness and repeatability.

3.3.2 Problem Definition

Image analysis in this context requires some initial work to map the image elements to a mass estimate. Given a training dataset, different algorithms can learn the relationship between an image and the mass of the plant. For this analysis, a function that defines the relationship will be predefined and an algorithm will learn the weights necessary to best estimate the biomass. These functions differ in complexity, from linear regression to complex neural networks. There may be higher-level methods for learning functions and not just weights, but these were not explored in this research.

It is also necessary to define the format of images before proceeding. For the sake of simplicity, one image per plant was taken at each evaluation. The viewpoint of the image was chosen as a bird's-eye view, like the example in Figure 3.9, to maximize the amount of the plant visible in a single frame. Additionally, to capture spatial information that is not included in a color image, RGB-D images were chosen as inputs for the image analysis, allowing for a depth map to be evaluated alongside the color image.

Visual observation of plant growth revealed that younger plants grew by expanding their canopy surface area. However, this canopy growth would saturate at a mass of about 30 g, instead increasing leaf density inside a constant canopy size. In Figure X, example images are shown from a fixed height of plants at different stages of development to visualize canopy area growth. Once canopy growth saturates, it becomes difficult to generalize image analysis to plants after the saturation threshold. While potentially feasible, improving the generalization ability of the model was left out of the scope of this research. This decision was partially inspired by the Buxbaum et al. (2022) paper, which also only focuses on premature plants in a similar application. [5]

Figure 3.9: Example image showing bird's-eye view.

3.3.3 Considered Architectures

A naive function, labeled Architecture A, might count the green pixels in an RGB image and conduct a linear regression analysis to estimate the mass from the pixel count. This approach effectively encodes mass as a function of the canopy area, which can be difficult with larger plants that do not expand the canopy area as quickly. This approach will be used as the baseline for comparison to analyze the practicality of more complex methods.

Another approach is to use convolutional neural networks (CNNs) to define a more complex function that can receive an RGB-D input image and output an estimated mass for an image. Different CNN architectures define different functions that can encode information from different parts of the image. The ideal architecture requires balancing complexity with various trade-offs in training time, memory requirements, and the risk of overfitting. While training time was less important for this project, memory requirements and overfitting were key points of consideration in the architecture design. This research has defined three CNN architectures with different amounts of complexity in order to contrast their relative performances and discuss potential trade-offs.

Architecture B is relatively naive, passing the image through a series of three convolution layers. This approach aims to estimate the mass by simply extracting

features. A visualization of this architecture is shown in Figure 3.10.

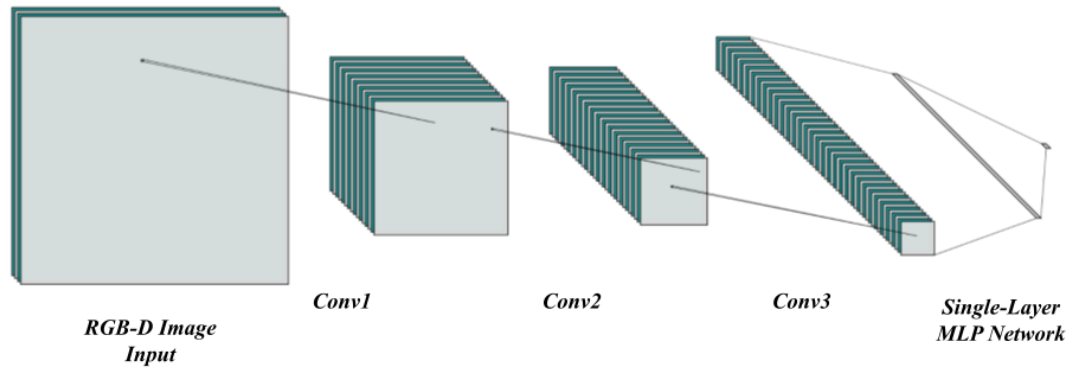

Figure 3.10: Architecture B: a CNN with three convolution layers.

Architecture C will leverage current research into deep CNN architectures for complex feature extraction. One prominent example is the foundational model Residual Neural Network, or ResNet, [12] which is commonly used in computer vision applications in classification contexts. Although typically used for classification purposes, the initial convolution can be adjusted to take in a four-channel input, and the output classification layer can be replaced with two fully connected layers to produce a single output. A visualization of this architecture is shown in Figure 3.11.

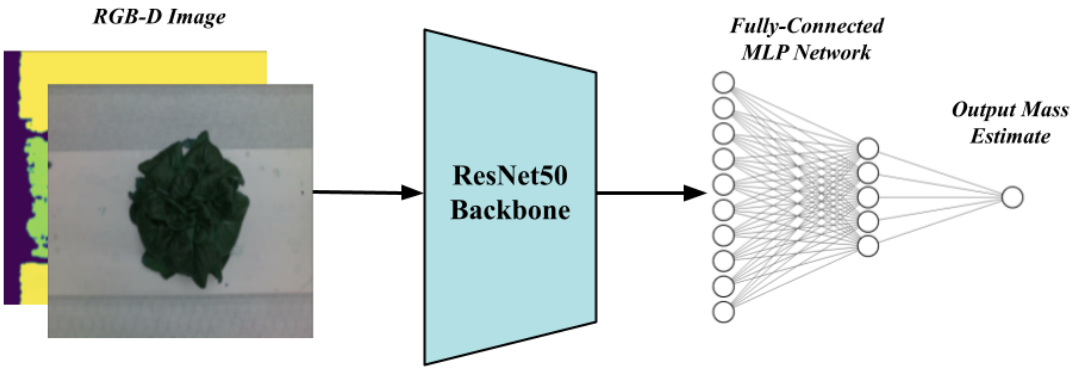

Figure 3.11: Architecture C: a CNN with a ResNet50 backbone.

Lastly, architecture D, inspired by Buxbaum et al., separates the color and depth images, funnels them through separate subnetworks, and combines latent vectors into

a unified network. Both the color and depth subnetworks are similar in design to the second CNN architecture, passing an image through a ResNet backbone. Unlike the previous architecture, the subnetworks produce a final latent vector, which is concatenated and passed through a basic MLP network. This MLP network feeds the concatenated vector through two fully connected layers to produce a final mass estimate. A visualization of this architecture is shown in Figure 3.12.

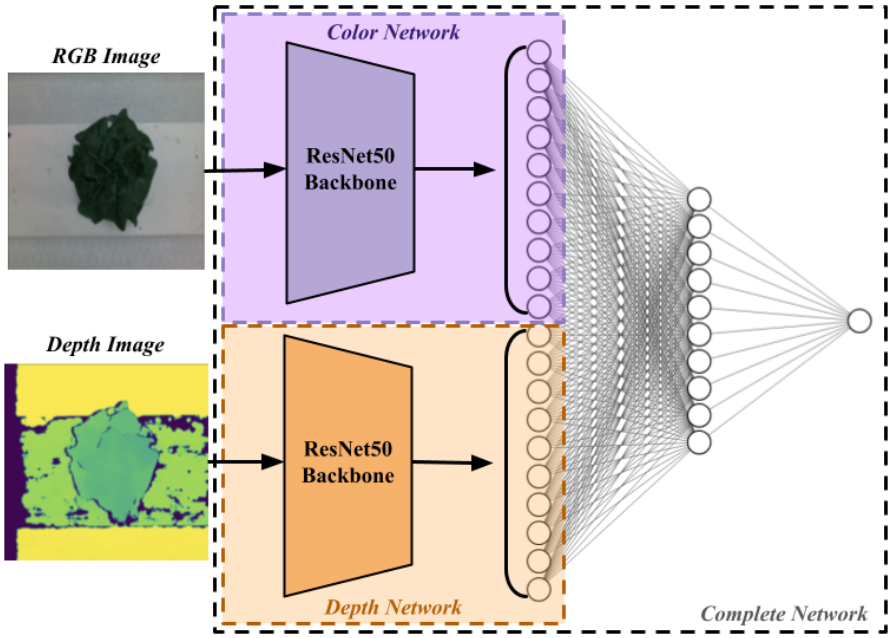

Figure 3.12: Architecture D: Buxbaum et al. inpsired CNN with ResNet backbones.

Two methods of training architecture D will be attempted.

- **Training Method (1):** The first approach will train the entire network architecture from scratch.
- **Training Method (2):** The second approach will train the color and depth subnetworks to separately evaluate the mass before completely training the entire architecture, initialized with the trained subnetwork weights.

Once architectures A-D are trained and evaluated, ablation studies will be performed to tune the model's hyperparameters. After the image analysis method is optimized, it will be used to send a live mass estimate to the simulation side of the digital twin. With a history of these mass estimates, the digital twin can better localize the current state of the plant to improve the accuracy of future mass forecasts.

3.3.4 Training Procedures

In order to compare these different approaches, the above architectures will be constructed in a Python environment using the PyTorch PyPI library. A training set of images, consisting of RGB-D images paired with ground-truth estimates, will be used to train the models. The network architectures will be compared by their average error performance, used to define the predictive capacity of the architecture, on a test set. From the architecture with the best performance, ablation studies will be performed to adjust the hyperparameters of the selected model, including analyses of possible augmentations, learning rates, and other model changes to improve accuracy.

3.4 Modelling and Forecasting

3.4.1 Motivation

With the mass evaluation method, the digital twin has all the necessary components to measure the processing system in the digital twin. To complete the architecture, there still needs to be a procedure in place to update a model that tracks the behavior of the system and to project growth. This section will define and compare different models for this context, determining optimal update procedures, and defining how future biomass projection will be accomplished.

The different model options defined in this section will be numerical extrapolation, botanical-based modeling, and neural-network-based modeling, each of which aims to capture different levels of model complexity and system knowledge.

3.4.2 Problem Definition

Given new measurements, a digital twin in this setting should be able to update its previous understanding of the current growth of the plant to better reflect new information. To accomplish this task, the digital twin architecture requires adjustable model parameters to minimize modeling error, so each model defined in this section will define its adjustable system parameters. To evaluate the modeling error, a time series of biomass growth will be defined for each plant, labeled 'growth trajectory' in a training set. Using available growth trajectories, the current modeling error will be

calculated by finding the average error between a measured mass and a forecasted mass, and each model will optimize its system parameters to minimize the modeling error. Model updates must minimize error over the entire batch of plants grown, since the model should be valid for all plants growing simultaneously.

Each model will also have different methods for forecasting future masses, which will be detailed in their relevant sections. While forecasting inputs vary by the model type, all forecasting methods share a common output, which is a projected mass after a certain 'prediction horizon.' This prediction horizon defines a constant time interval between the current measurement and a forecast date. The different models will be compared in their prediction accuracy for a variety of prediction horizons, and the model with the best performance will be selected for further use in the digital twin architecture.

Lastly, a training procedure will be defined for model calibration for a specific growth environment. These data-informed calibration steps will help initialize the model to begin forecasting immediately, avoiding uncalibrated initial conditions that lack information about the specific environment.

3.4.3 Exponential Model

Definition

The shape of a plant's biomass growth for a lettuce plant was assumed to roughly follow a sigmoid curve under ideal growing conditions. This sigmoid curve represents an initial exponential growth, followed by growth saturation in the mature stages of plant growth. Although long-term growth of the plant saturates, this project only considered the first 2-4 weeks of growth before maturity, so a short-term exponential growth shape was assumed. This exponential growth was defined by Equation 3.1.

$$M_{FM} = Ae^{Bt} \tag{3.1}$$

Adjustable Parameters

In this model definition, initial biomass A and exponential growth rate B are adjustable parameters to improve the modeling error of the digital twin.

Update Procedure

The modeling error is minimized by fitting an exponential curve to a window of previously measured biomass data. This exponential fit is conducted through a linear least-squares regression in the logistic domain.

Initial Calibration

An initial calibration method was not defined for the exponential model, as the exponential fitting process was quick and deterministic.

Biomass Projection

Future masses are calculated by extrapolating a mass from the fitted exponential curve. The input provided for the extrapolation process is a window of daily mass measurements, and the predicted mass is on a future date after the defined prediction horizon.

3.4.4 Botanical-Based Model

Definition

As mentioned previously, the NiCoLet B3 model [20] simplifies lettuce growth into a first-order differential equation model, using a biological approach to estimate growth by the carbon balance of the plant and the concentration of nitrate in the nutrient solution. This section will describe the mathematical behavior of the NiCoLet B3 model, and the following equations will mathematically describe the specific model dynamics. For brevity, the variable names and default values not mentioned in this section are contained in Appendix A.3.

Plants gain biomass from the process of cell multiplication, spending carbon-based sugars to build new cells. In order to model this process as a first-order differential equation, the rate of change for the carbon balance was defined. The carbon balance is defined as a two-element vector, estimating the mass of carbon in the vacuole, M_{cv}, and in the structure of plant cells, M_{cs}. The rate of change for this vector is defined by the processes above, which describe the inflows and outflows for both elements.

Additionally, the model contains three control parameters: temperature T, lighting I, and carbon dioxide C, which affect the intensity of the different processes.

The balance of carbon is the core element of this model and describes how plants acquire, consume, or internally transport carbon to facilitate growth. There are four core processes that alter the carbon balance: photosynthetic assimilation, growth, maintenance respiration, and growth respiration. Photosynthetic assimilation is the primary method for acquiring carbon, using the energy of light to convert carbon dioxide into a usable, carbon-based sugar. While photosynthesis obtains carbon for the plant, growth defines the internal transfer of this carbon from an initial storage area inside the plant cell, called the vacuole, into the specific structure of plant cells. In the model, the growth process describes this transfer of carbon, which includes redistributing carbon to new plant cells formed by multiplication of cells. Both forms of respiration, maintenance and growth, are processes for consuming the initial carbon-based sugar from the vacuole to produce usable energy. Maintenance respiration describes the production of energy to maintain the current set of plant cells, while growth respiration describes the energy produced for cell multiplication.

For photosynthetic assimilation, the rate of carbon acquired by the system is the product of the maximum photosynthesis rate, canopy cover, and the photosynthesis inhibition factor. The maximum rate of photosynthesis is determined by the amount of light and carbon dioxide, which are the drivers of the photosynthesis chemical reaction. The canopy cover, the percentage of the ground visible from above, is defined by an exponential saturation function where the rate is controlled by a leaf area closure parameter a. The photosynthesis inhibition factor is determined by the amount of carbon in the vacuole, inhibiting the flow of carbon from photosynthesis to the vacuoles if they are already saturated with carbon. Equations 3.2-3.6 show the full definition for photosynthetic assimilation F_{cav}.

$$C_{cv}(M_{cs}, M_{cv}) = \frac{M_{cv}}{\lambda \cdot M_{cs}} \tag{3.2}$$

$$p(I, C) = \frac{\epsilon \cdot I \cdot \sigma \cdot (C - C^*)}{\epsilon \cdot I + \sigma \cdot (C - C^*)} \tag{3.3}$$

$$f(M_{cs}) = 1 - e^{-aM_{cs}} \tag{3.4}$$

$$h_p(M_{cs}, M_{cv}) = \left(1 + \left(\frac{(1 - b_p) \cdot \Pi_v}{\Pi_v \cdot \gamma \cdot C_{cv}(M_{cs}, M_{cv})}\right)^{s_p}\right)^{-1} \tag{3.5}$$

$$F_{cav} = p(I, C) \cdot f(M_{cs}) \cdot h_p(M_{cs}, M_{cv}) \tag{3.6}$$

Maintenance respiration is defined as a function of the current carbon mass in the vacuole by a specific maintenance respiration variable. This variable is the product of a maintenance respiration coefficient, denoted as k, and a factor that changes exponentially based on the difference between the current temperature and a set point temperature. Equations 3.5-3.6 show the full definition for maintenance respiration F_{cm}.

$$e(T) = K \cdot e^{c(T - T^*)} \tag{3.7}$$

$$F_{cm} = e(T) \cdot M_{cs} \tag{3.8}$$

For growth, it is a product of the maximum growth rate, the canopy cover, and the growth inhibition factor. The maximum growth rate is simply the specific maintenance respiration rate, multiplied by a growth rate coefficient v. The growth inhibition factor is a factor of the amount of carbon in the vacuole, slowing growth if too little carbon is available in the vacuole to build additional plant cells. Growth respiration is simply the growth rate multiplied by a growth respiration loss factor, denoted as θ. Equations 3.9-3.12 show the full definition for growth F_{cvs} and growth respiration F_{cg}.

$$g(T) = v \cdot e(T) \tag{3.9}$$

$$h_g(M_{cs}, M_{cv}) = \left(1 + \left(\frac{b_g \cdot \Pi_v}{\gamma \cdot C_{cv}(M_{cs}, M_{cv})}\right)^{s_g}\right)^{-1} \tag{3.10}$$

$$F_{cvs} = g(T) \cdot f(M_{cs}) \cdot h_g(M_{cs}, M_{cv}) \tag{3.11}$$

$$F_{cg} = \theta \cdot F_{cvs} \tag{3.12}$$

Finally, the carbon balance rate of change is defined by Equations 3.13 and 3.14 below. For the vacuole carbon state, photosynthetic assimilation adds carbon, whereas every other process removes carbon. For the carbon state of the structure, the growth process adds carbon, while maintenance respiration removes carbon.

$$\dot{M}_{cv} = F_{cav} - h_g(M_{cs}, M_{cv}) \cdot F_{cm} - F_{cg} - F_{cvs} \tag{3.13}$$

$$\dot{M}_{cs} = F_{cvs} - (1 - h_g(M_{cs}, M_{cv})) \cdot F_{cm} \tag{3.14}$$

Equations 3.15 and 3.16 are used to convert the state vector of the model, in the form of a carbon mass, into an output vector. This output vector is a two-element vector with fresh and dry biomass metrics, M_{FM} and M_{DM}. Both biomass metrics are mass density, but they describe the mass density for a constant plot of ground, not for the canopy area of the plants.

$$M_{DM} = \eta_{OMC} \cdot (M_{cv} + M_{cs}) + \eta_{MMN} \cdot (\frac{\lambda \cdot \Pi_v}{\beta} \cdot M_{cs} - \frac{\gamma}{\beta} \cdot M_{cv}) \tag{3.15}$$

$$M_{FM} = 1000 \cdot \gamma \cdot M_{cs} + M_{DM} \tag{3.16}$$

Adjustable Parameters

From a sensitivity analysis performed on the original NiCoLet model by Lopez-Cruz et al., only three parameters were found to have significantly affected the behavior of the model [28]. Those parameters were the maintenance respiration coefficient k, the growth rate coefficient v, and the leaf area closure parameter a.

All of these parameters play significant roles in the carbon balance equations, and tuning them can calibrate the model to a new environment or plant species. Looking at each parameter individually:

- k controls the maintenance respiration, scaling the baseline amount of carbon

consumption necessary for survival.

- v scales the maximum growth rate of the plant. Increasing this metric would mean rapid exponential growth, but also rapidly increasing carbon costs for growth.
- a controls how fast the canopy cover of the plant increases or spreads out, which controls the amount of usable light for photosynthesis.

Update Procedure

To update the selected parameters, an optimization process will be tested that minimizes the residual between the simulated and ground-truth trajectories, given the environmental conditions of growth. The residual between the two trajectories was derived by calculating the absolute value of the error at each measured point in the ground-truth trajectory and averaging these errors.

Importantly, for modeling error, the difference between the ground-truth measurements and the NiCoLet model was in the units of mass measurement. The NiCoLet model is designed to simulate the mass density of plants in a larger plant bed, with units $\frac{kg}{m^2}$. The ground-truth measurements were in units of grams, so a conversion method was necessary for comparison. The NiCoLet model strictly defines the area used for mass density as the ground area, defined by the density of the plant over the bed. For example, plants grown 10 cm apart would receive a ground area of 100 cm^2. To convert the ground-truth trajectory from individual mass to mass density, each plant was given a ground area of 7 inches, which was derived from the recommended spacing distance for plant species from the seed wholesaler. The mass density was converted to the mass of the individual plant by multiplying the output by the fixed ground area.

Since the parameters chosen were highly sensitive and accurate gradients were expensive, a non-gradient optimization solver, Nelder-Mead, was chosen for residual minimization. All parameter optimization was done in Python using the scikit-learn library.

Initial Calibration

To initialize the calibration of the model, the baseline values for the selected parameters were taken from the original NiCoLet B3 model. These parameters are as follows:

- Maintenance Respiration Coefficient (k) = 4e-7
- Growth Rate Coefficient (v) = 22.1
- Leaf Area Closure (a) = 0.2

Since these values were intended to best describe growth in the environment of the original article, this project will carry out a new calibration to adjust the parameters to the new lettuce species and environment.

Parameter calibration will use the same Nelder-Mead optimization to minimize modeling error in the training data. To better explore possible optima, three categories of optimization were performed: one-dimensional searches, multidimensional searches, and sequential one-dimensional searches. These methods were meant to intentionally seek out new local optima at the risk of overfitting a specific parameter to the training data. Based on the performance in the calibration process, the most successful optimization method will be carried over to live digital twin simulation, using this method as the basis of model updates.

Biomass Projection

In order to predict future masses, a simulation will be initialized with lighting, temperature, and carbon dioxide readings. A future mass will be predicted by extrapolating the simulation to a certain prediction horizon.

3.4.5 Neural Network-based Model

Definition

Where the NiCoLet model is informed by biological processes but remains only a first-order ordinary differential equation, a neural network approach was considered in order to model some system complexities not captured in the biological model. A specific challenge for any neural network in this application is the limited amount of data, or a low-data regime problem. Since plants grow slowly and measurement opportunities

are infrequent, a neural network must be able to learn complex relationships reliably, requiring data-efficient training.

To test the capability of neural networks in a data-limited setting, three models were considered: a basic multilevel perceptron (MLP), a long short-term memory (LSTM) model, and a transformer-based model. The input for the model was a time series window, formatted as a two-column array with a time series of lighting and the biomass in the first and second columns, respectively. These window measurements correspond to the previous daily measurements in the chronological order in which they were taken. Other environmental readings could be added to this array in the future, but conditions, except lighting, were held constant in the dataset.

$$\text{Input} = \begin{bmatrix} l_1 & m_1 \\ \vdots & \vdots \\ l_n & m_n \end{bmatrix}$$

While MLP was a simple baseline neural network, it was not chosen as it does not encode the temporal dependencies necessary to predict future values. Learning temporal dependencies, or a generalizable understanding of how the plants grow for arbitrary windows, is crucial for this application. An MLP architecture does not have mechanisms in place to learn these growth dynamics in a data-efficient manner and would not be the best fit for this low-data regime context.

A transformer-based approach was tested, but it was also not chosen due to its data efficiency. Transformer models are known to struggle in low-data regimes relative to other neural networks because of their high parameter density. While a transformer-based architecture would likely outperform other models given a high-data context, its potential performance might not be fulfilled in a low-data context, where it is likely to underperform alternative neural network architectures without additional data [25].

The chosen method was an LSTM model. In a low-data regime, the ability to recall temporal dependency information through robust memory structures utilizes data efficiently, and by focusing most on pattern recognition, LSTMs are able to generalize well and avoid overfitting. Ultimately, the LSTM model was chosen for its strong data efficiency.

To test LSTM networks in this context, a neural network architecture was designed.

The LSTM architecture has four total layers, with two initial LSTM layers followed by two fully-connected MLP layers. The input array is first processed by the first LSTM layer with 64 output dimensions, followed by a second LSTM layer with 32 output dimensions. The resulting 32-element latent vector is passed through the MLP layers to produce a single scalar value, representing a predicted mass after a predefined prediction horizon.

In order to build the LSTM-based mass prediction pipeline, an LSTM neural network, based on the architecture shown in Fig. X, was constructed in a Python environment with the TensorFlow.Keras PyPI library. LSTM layers were created using the predefined LSTM class defined by the Keras.Layers library, with the default tanh activation function in the LSTM unit input gate.

Adjustable Parameters

Since the LSTM network is a "black-box" approach, the specific behavior of different parameters was not immediately known, and for the sake of simplicity, all parameters of the LSTM model were available for adjustment.

Update Procedure

In order to build a training set for model training, input growth windows were built from individual plant growth time series. To format the input windows and pair them with a corresponding future mass, the growth trajectories needed to be sliced into smaller windows. Given the length of the window and the prediction horizon, a training set was prepared by sliding a window through the trajectory. A window was included in the training set only if it contained measured data within the window itself and at the specified horizon point in the future.

In training, the model attempts to learn the temporal dependencies necessary for forecasting by minimizing the mean squared error (MSE) between actual and forecast mass estimates. To update the entire model, the LSTM network will use an incremental learning technique to reinitialize the network and add measured data to the training set.

Initial Calibration

Before incremental learning, the LSTM model is trained offline to project future mass on the collected data, using the same process as the online update procedure. To optimize the architecture's initial prediction accuracy, ablation studies were performed to improve the model's hyperparameters. These studies explored varying the length of the input sequence, adjusting the prediction horizon, and other hyperparameters within the LSTM block.

After completing offline training, the model weights are stored locally and used to initialize live parameter updates during the operation of the digital twin.

Biomass Projection

Given an input window and trained model weights, the LSTM model will be trained to project a future mass on a specific prediction horizon.

Chapter 4

Experiments

4.1 Image Dataset

4.1.1 Motivation

The two experiments defined in Chapter Chapter 3 required training data to evaluate model performance.

The first experiment, detailed in Section section 4.2, will train different model architectures for mass evaluation, which is a crucial measurement for digital twin updates. Since all of the mass evaluation methods require some initialization on real data before use, a training dataset needs to be collected that includes RGBD images paired with ground-truth mass measurements.

The second set of experiments, detailed in Section section 4.3, is for the model-based forecasting component of the digital twin architecture. Unlike mass evaluation training, training data is composed of growth trajectories, or time series of growth data, to calculate the modeling error for the growth of a plant. For this experiment, the collected dataset needs to include full time series for an individual plant's growth, tracking biomass over an entire growth period of two to three weeks.

Lastly, both training and testing sets are required to be separated from the mass evaluation training set, This is required since models will be trained and evaluated on data that passes through the mass evaluation module, to approximate performance with real-world evaluation error.

In summary, a dataset of RGBD images and plant masses needs to be collected to train and evaluate each of the experiments defined in Chapter Chapter 3, and the data composition needs to reflect task requirements, based on their context in the digital twin architecture.

4.1.2 Composition

To construct the dataset, Butterhead lettuce plants were grown, as specified in Chapter Chapter 3, periodically over the months between August and December 2024. Plants underwent measurement once a day for two weeks on average, where an RGB-D image and fresh biomass were measured. All images maintained a resolution of 640 x 480 and were saved to the common directory. For ground-truth mass measurements, each plant was lifted out of its channel hole and placed in an off-the-shelf kitchen scale for measurement, which had a resolution of 1 gram. After subtracting the saturated mass of the medium used for hydroponic growth, the isolated fresh biomass of the plant was noted and saved on a CSV file, alongside the timestamp for the image taken.

Over the course of the four months, 1308 plants were measured. All of these plants have a labeled date, fresh biomass, RGB-D image, and a lighting condition. The timestamps for an individual plant's growth were also noted, in order to build a time series of each plant's growth. In total, 125 individual plants were measured, with a median lifespan of 14 days. The spread of the biomass measurements is pictured below in Figure 4.1, skewing towards lighter masses. This skew reflects the exponential nature of plant growth, where plants spend more time in early growth stages before rapidly accumulating mass. The average mass in the dataset is around 10 g.

Within the dataset, there is an additional variable: the distance of the stereo camera to the plant bed. The camera distance was varied to allow for future mass estimation models to generalize to different distances, but this project restricted the scope to a fixed distance. Within the dataset, there is a subset of 1095 images that were taken from a fixed distance of 175 mm.

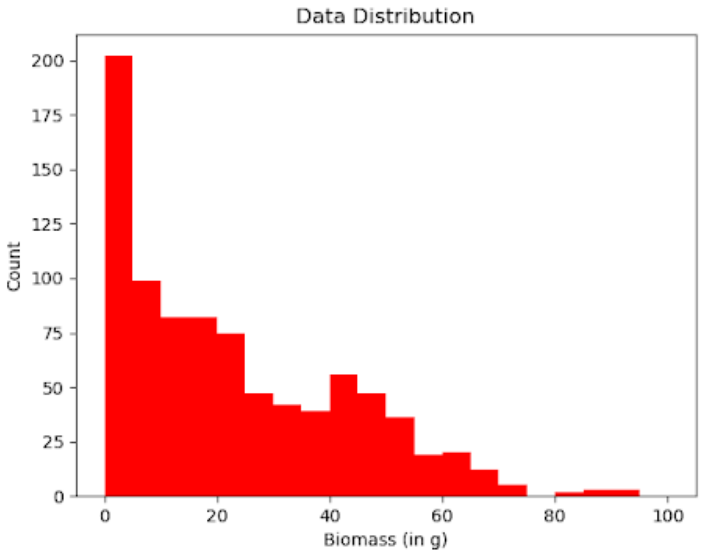

Figure 4.1: Distribution of mass in the custom RGB-D dataset.

4.1.3 Task-based Data Allocation

Using only the sub-set of 1095 images at a fixed distance, two items needed to be trained: a mass evaluation module and digital twin models, specifically a NiCoLet model and an LSTM-based neural network.

Most of these measurements, 823 mass-image pairs, belonged to a time series of 8 or more measurements detailing the growth of an individual plant. Growth trajectories of individual plants were necessary for the model-based forecasting experiments, which require biomass measurement histories to effectively minimize modeling error. The remaining 272 measurements were automatically assigned to the mass evaluation training set, which could train on lone mass-image pairs.

Beyond this initial allocation, many of the feasible mass evaluation models are training intensive, so more data than the initial 272 measurements needs to be allotted to this task. Since the remaining data split fairly evenly into three subsets with an equal quantity of mass-image pairs, around 275 each, and growth trajectories, around 25 each, three subsets were created and subsequently allotted to different tasks.

Since the models occur ‘downstream’ from the mass evaluation system, the two training tasks required separate subsets of the broader dataset. Thus, a subset was

allotted as additional data for the mass evaluation training and another was allotted for the model calibration required for the model-based forecasting task. Lastly, a final subset was kept aside as a universal testing dataset, since reusing the data for multiple tasks was not an issue. A visualization of the data organization is shown below in Figure 4.2.

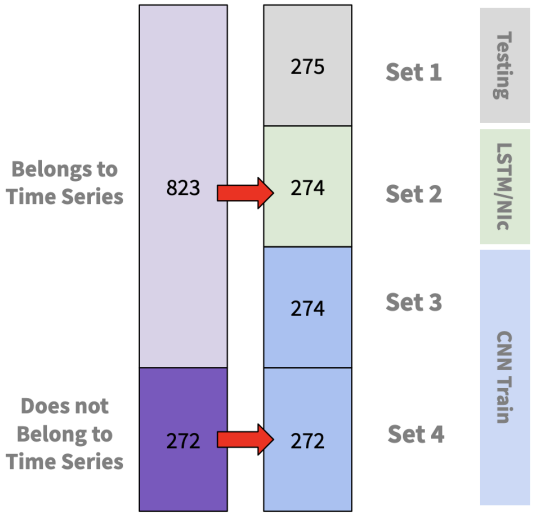

Figure 4.2: Data organization for training individual models, using equal-length folds for neat splitting.

4.2 Mass Evaluation

4.2.1 Problem Definition

As mentioned in Chapter Chapter 3, a mass evaluation measurement system was required for digital twin model updates, and this project chose to study image analysis as a non-destructive and autonomous method for biomass evaluation. This research explored different architectures that could define the relationship between RGB-D images and mass, each with different amounts of complexity. Using the subset of data allotted for mass evaluation, the image analysis architectures were trained to evaluate the mass in a given image. This chapter will explore the trade-offs for these different functions, namely the risk of underfitting or overfitting the training set. Lastly, a

series of ablation studies will explore which pre-processing steps and hyperparameters are optimal.

To define the performance of any given function for the evaluation of the mass, a few different metrics were used. The first metric is the average error, defined by the root mean squared error (RMSE), and was used to define the predictive capacity of the function over the entire test set, which was the primary metric used for comparison, since it directly quantifies the evaluation error.

Additional metrics used were the coefficient of determination, R^2, for an actual vs. evaluated scatter plot and classification metrics. The R^2 score analyzes whether a method has consistency over the entire range of masses or whether evaluation bias is introduced in a subset, such as underprediction at high masses. The different classification metrics define the model's ability to forecast masses that are within a specific threshold of the actual mass. The listed accuracy, precision, recall, and F1 metrics for a model were initially defined with a 1g threshold. Although this may be considered a strict classification, it was defined in this way to evaluate 'near-perfect' guesses.

4.2.2 Architecture Comparison

Architecture A is a naive linear function that defines the mass as directly proportional to the number of green pixels in an image. 'Green pixels' were defined by the range of HSV values between (35, 35, 35) and (85, 255, 255). A linear regression analysis was performed on the full set of 823 training data points available, learning the function:

$$m = (2.03 \cdot 10^{-4}) \cdot p + 1.43$$

where the mass m and the number of green pixels p are given. Applying this function to the testing subset, the average error was found to be 2.40 g.

The next three architectures are defined by CNNs and trained on 300 images in the training set, the rest being assigned to a validation set. All models were initially trained with a mean squared error (MSE) loss function.

Architecture B was defined by a simple CNN architecture, shown in Figure 3.10. It consists of three sequential convolution layers and a final MLP layer, which outputs a single scalar value. This CNN was trained and tested, producing an average error of 2.21 g in the test subset.

Architecture C was defined by a more complex ResNet50 network, shown in Figure 3.11. In this architecture, the publicly available ResNet50 architecture was altered to fit the desired input and output for this application. This was done by altering the first convolution layer to receive a four-channel input and replacing the final classification layer with a two-layer MLP network, which produced a scalar output value. This approach was tested, producing an average error of 2.22 g.

Architecture D tested was an architecture inspired by the Buxbaum et al. approach. Rather than altering ResNet50 to receive a four-channel input, the color and depth images were instead passed through separate subnetworks, with final latent vectors in the subnetworks concatenated and passed through a final MLP network. This approach aims to learn information from both image modes separately before combining that learned information into a final network. A visual diagram of the model architecture can be seen in Figure 3.12. This model was trained with two methods:

- **Training Method (1):** From scratch, which produced an average error of 2.03g
- **Training Method (2):** Initializing training with pre-trained subnetworks, which produced an average error of 1.91g.

Architecture	RMSE	R^2	Acc.	Prec.	Rec.	F1
Pixel Regression [A]	2.40	0.808	0.149	0.100	0.106	0.087
Basic CNN [B]	2.21	0.910	0.194	0.113	0.094	0.098
ResNet-based CNN [C]	2.22	0.815	0.179	0.082	0.097	0.078
Buxbaum, Training (1) [D(1)]	2.03	0.833	0.244	0.119	0.151	0.124
Buxbaum, Training (2) [D(2)]	1.91	0.875	0.214	0.104	0.100	0.096

Table 4.1: Results for training architectures A-D, as well as comparing two training methods for architecture D.

From the results seen in Table 4.13, architecture D performed the best. Both approaches to training the full architecture performed better in different metrics, but training method (2) performed better in the primary RMSE metric.

Architectures A and B performed relatively well compared to their more complex counterparts, proving that these two approaches might fit in a project that is more

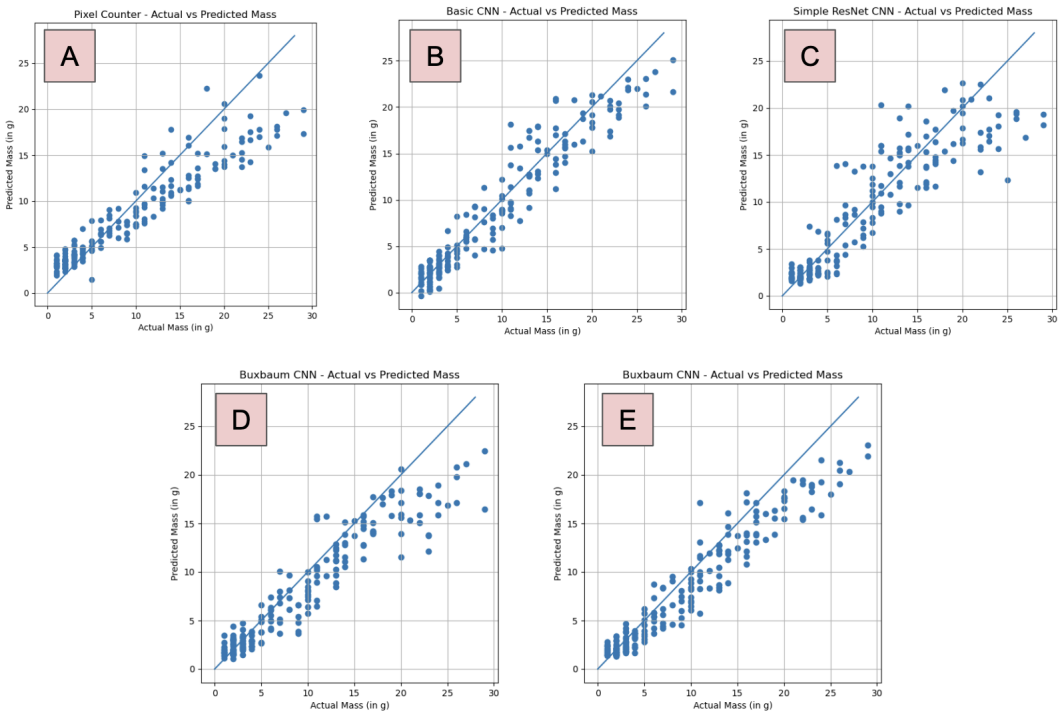

Figure 4.3: A comparison of all actual vs. predicted scatter plots for the trained mass evaluation models. Plots A-C correspond to architectures A-C, while Plot D and E are for training method (1) and (2), respectively, for architecture D.

time or memory constrained. Without those constraints for this project, these architectures were abandoned in favor of architecture D.

This method will be refined with the ablation studies below that seek to optimize the performance of architecture D by testing alternative hyperparameters and introducing pre-processing steps.

4.2.3 Ablation Studies

First, it was hypothesized that applying a color segmentation to separate the plant from the background would remove possible background noise for smoother model training. To apply this color segmentation, a range of green HSV values was provided, and nongreen pixels were made black and given a nominally large depth distance. A morphological closing process was applied to close the holes and remove noisy background patches.

Figure 4.4: Example of converting a sample image on the left into a color-segmented image on the right.

In this study, shown in Table 4.2, the ResNet50 backbone was trained in the same way as before, but both the training and the test set images were color segmented or not. The study showed quite clearly that the use of color segmentation improved the accuracy of the final model. Color segmentation will be used in the following studies to maintain this improved performance.

Method	RMSE	R^2	Acc.	Prec.	Rec.	F1
No Segmentation	1.91	0.875	0.214	0.104	0.100	0.096
Added Segmentation	1.56	0.915	0.264	0.156	0.156	0.145

Table 4.2: Results for ablation study on color segmentation

It was also hypothesized that recent advancements to pre-trained CNN architectures could improve the performance of the model, substituting for the ResNet50 backbone in the current architecture. Two candidate foundational models were selected: EfficientNetv2 [50] and DenseNet [15], which emerged after the 2022 Buxbaum et al. paper and performed better than ResNet50 in classification metrics with similar parameter counts. The results between the substituted backbones are shown in Table 4.3. The DenseNet backbone performed better than EfficientNetv2 in both RMSE and R^2, and while it performed worse in classification metrics, DenseNet was chosen as the ideal backbone for the model.

Then, an analysis was performed on different types of augmentation to increase the image diversity in the training set. Inspired by the augmentations done in the

CNN Backbones	RMSE	R^2	Acc.	Prec.	Rec.	F1
ResNet50	1.56	0.915	0.264	0.156	0.156	0.145
EfficientNetv2-S	1.48	0.920	0.313	0.217	0.217	0.211
DenseNet121	1.45	0.929	0.254	0.142	0.127	0.126

Table 4.3: Results for ablation study on Architecture D CNN backbones.

Buxbaum et al. (2022) paper, a number of image augmentations were attempted in the preprocessing phase. These augmentations performed an operation and added a new image to the training set. Applied augmentations were:

- **Image Flipping:** Images were either flipped vertically, horizontally, or on both axes, each with a 10% chance.
- **Random Cropping:** Images were randomly cropped and resized
- **Random Rotation:** Images were randomly rotated up to 180 degrees
- **Random Grayscale:** Images had a 10% chance of being converted to grayscale
- **Random Color Jitter:** Images had their brightness randomly varied between 50% and 150%, along with random changes to pixel hue, saturation, and value (HSV).

Each of the augmentation operations is shown visually in Figure 4.5.

The first augmentation study trained with different geometric augmentations, specifically random flip, rotate, and crop augmentations. Several possible augmentation combinations were tested, and Table 4.4 lists the results. Compared with no augmentations, none of the geometric augmentations performed better, likely due to overfitting. Crop augmentations performed particularly poorly, possibly due to resizing the apparent plant canopy size, confusing the embedded relationship between canopy size and mass.

The next augmentation type studied was color augmentation, defined as random grayscale or color jitter operations. The results are shown in Table 4.5. Although the average error and the coefficient of determination did not improve in any of these combinations, certain combinations improved classification metrics, defining when an evaluation method has more 'near perfect' estimates. With these improvements in mind, the combination of 10% grayscale probability and color jitter augmentations was chosen as optimal augmentations for model training.

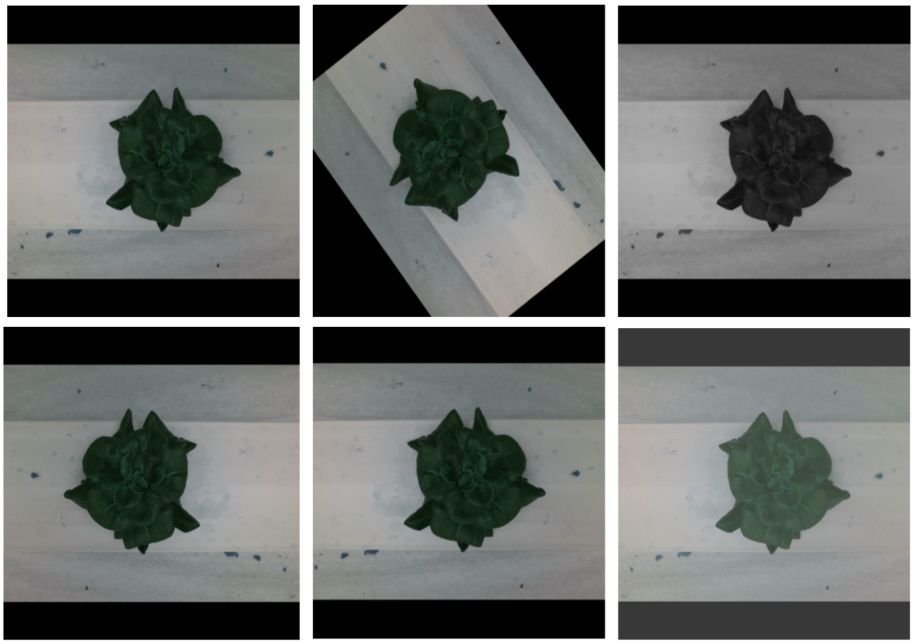

Figure 4.5: Image mosaic of example augmentations done to a sample image. Top, Left: Original; Top, Center: Rotation; Top, Right: Greyscale; Bottom, Left: Flip; Bottom, Center: Crop; Bottom, Right: Color Jitter

Flip	Rotate	Crop	RMSE	R^2	Acc.	Prec.	Rec.	F1
0	0	0	1.45	0.929	0.254	0.142	0.127	0.126
1	0	0	1.48	0.927	0.259	0.145	0.172	0.137
0	1	0	1.56	0.908	0.313	0.166	0.167	0.162
0	0	1	1.66	0.900	0.244	0.140	0.147	0.136
1	1	0	1.47	0.925	0.259	0.134	0.126	0.126
1	0	1	1.63	0.911	0.259	0.135	0.149	0.134
0	1	1	1.63	0.902	0.254	0.155	0.155	0.146
1	1	1	1.71	0.900	0.219	0.103	0.119	0.107

Table 4.4: Results for ablation study on geometric augmentations.

Lastly, an ablation study was conducted for different learning rates and loss functions of the model. For learning rates, the initial learning rate of 5e-6 was compared to both 5e-5 and 1e-5 to observe if faster learning rates produced better results. Slower rates were also tested, but often became trapped in local minima with high RMSE. The MSE loss function was also compared with the mean average percentage error (MAPE) to observe if the metric might be less sensitive to outlier

Grey	Color Jitter	RMSE	R^2	Acc.	Prec.	Rec.	F1
0	0	1.45	0.929	0.294	0.142	0.127	0.126
1	0	1.47	0.925	0.294	0.199	0.228	0.194
0	1	1.54	0.920	0.264	0.162	0.162	0.150
0	0	1.45	0.925	0.313	0.223	0.243	0.204

Table 4.5: Results for ablation study on color augmentations.

images in the data set. The results are shown in Table 4.6, and it was ultimately determined that a 5e-6 learning rate and an MSE loss function were optimal for the application.

Loss Fn.	Learn. Rate	RMSE	R^2	Acc.	Prec.	Rec.	F1
MSE	0.00005	1.49	0.923	0.259	0.160	0.177	0.140
MSE	0.00001	1.48	0.925	0.259	0.148	0.151	0.140
MSE	0.000005	1.45	0.929	0.254	0.142	0.127	0.126
MAPE	0.00005	1.51	0.920	0.308	0.198	0.194	0.177
MAPE	0.00001	1.49	0.915	0.333	0.212	0.208	0.198
MAPE	0.000005	2.21	0.902	0.249	0.134	0.145	0.126

Table 4.6: Results for ablation study on loss and learning rate.

4.2.4 Final Performance

After running the ablation studies above, the best neural network structure tested was:

- **Architecture:** Buxbaum-inspired CNN, using training method (2)
- **Architecture Backbone:** Customized, pre-trained DenseNet CNN
- **Preprocessing:** Color-segmentation step
- **Augmentations:** Random greyscale and random color jitter
- **Learning Rate:** 5e-6
- **Loss Function:** MSE

The final performance metrics for this network were the following:

- **RMSE** = 1.45 g
- $\mathbf{R^2}$ = 0.925

- **Accuracy** = 0.313
- **Precision** = 0.223
- **Recall** = 0.243
- **F1** = 0.204

Figure 4.6 shows the actual vs. predicted plot for this finalized architecture. The plot shows a spread of predictions without significant prediction biases at any mass, while maintaining a fairly uniform error percentage throughout the range of masses.

This trained CNN will be used in Section 4.3 to provide simulated mass estimates.

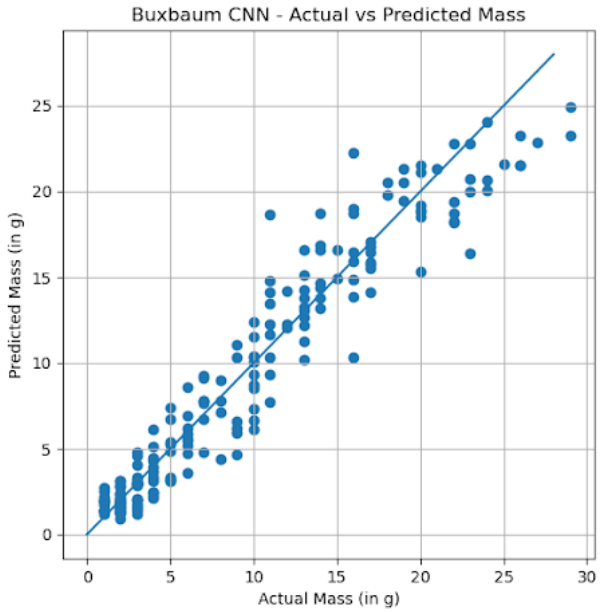

Figure 4.6: Actual vs. predicted scatter plot for the finalized CNN architecture.

4.3 Modelling and Forecasting

4.3.1 Problem Definition

This section will define the experiments used to compare the forecasting methods using the collected data set. To train and test each of the models below, a single

subset of the collected data set was used as the training set, and another subset was used as the testing set, with each set totaling about 25 growth time series. Since these methods utilize the trained CNN mass evaluation system to approximate real-world data, the training set is distinct from the data subset used to train the CNN in order to ensure independent training. Once the resulting models are trained, the prediction accuracy will be compared to select an optimal forecasting method for the digital twin architecture.

4.3.2 Exponential Model

This process did not require any initial calibration since an exponential model was simply fit to a window of data and extrapolated to forecast a future mass. Using the windows available in the test subset, Table 4.13 shows the performance of the exponential model at different prediction horizons. Performance metrics show an expected decrease in accuracy as the prediction horizon increases.

4.3.3 NiCoLet Model

Calibration Process

To initialize the Nelder-Mead optimization, the initial guess for the parameter set was borrowed from NiCoLet model literature [20]:

- Maintenance Respiration Coefficient (K) = 4e-7
- Growth Rate Coefficient (V) = 22.1
- Leaf Area Closure (A) = 0.2

To explore the solution space, three categories of optimization were performed on the NiCoLet parameters.

- Single-Parameter Optimization: This type allows for only a one-dimensional search. For example, a 'K' minimization would only require updating the maintenance respiration coefficient (K) to minimize the simulation error.
- Multi-Parameter Optimization: This type allows for multiparameter searches. For example, a 'KV' minimization would update both the maintenance respiration coefficient (K) and the growth rate coefficient (V) to minimize simulation

error.

- Sequential Single-Paramter Optimization: This type searches the space uniquely by performing a series of single-parameter optimizations. Once the first single-parameter optimization is complete, the next optimization uses that optimum as a baseline and continues with the optimization of a different parameter. For example, a 'K-V' minimization would denote optimizing the maintenance respiration coefficient (K) first and then the growth rate coefficient (V).

To come up with a robust set of possible parameter sets, each of the above optimization methods, totaling 11 variations, was applied to 10 randomly shuffled subsets within the training set. This resulted in 110 possible optimal parameter sets, which were subsequently tested on the test set to approximate the parameter optimizations that performed best.

Calibration Result

From this analysis, the values that performed best in the test set had the following parameter values:

- Maintenance Respiration Coefficient (K) = 4e-7
- Growth Rate Coefficient (V) = 22.8
- Leaf Area Closure (A) = 0.2

The average residuals for each of the optimization types are shown in Table 4.7, grouped by both the specific parameter optimization and the general categories of optimization. The most optimal parameter sets for minimizing residuals in the test set came from single-parameter optimizations, with other optimization types seeming to overfit to the training data set. In practice, single-parameter optimizations tended to remain close to the initial guess, while other optimization parameters found optimum parameters that deviated much farther.

Among the single-parameter optimizations, the growth rate coefficient optimization was the most successful at minimizing error, both on average and in an individual parameter set. Since all single-parameter optimizations performed similarly, the project chose this category for online updates, selecting the best parameter set based on the modeling error for each batch of plants.

Lastly, the calibrated parameter values were compared with the initial parameter

	Average Error
Single-Parameter	1.81g
Multi-Parameter	1.82g
Seq. Single-Parameter	2.38g

Table 4.7: The average modeling error for all optimization types tested on the evaluation set.

set. A future mass was forecasted at different prediction horizons, and their prediction accuracies were compared. The prediction accuracy was measured by the absolute error between a ground-truth mass and a predicted mass. Table 4.8 shows the relative performance of the baseline NiCoLet model to the calibrated NiCoLet model, showing that calibration improved the performance. Since this calibrated parameter set performed better than the initial parameter set, the calibrated parameter set was instead used as the initial guess for a brand-new digital twin, updated with new growth measurements.

	1-Day Horizon	2-Day Horizon	3-Day Horizon	4-Day Horizon
Baseline NiCoLet	2.41g	2.50g	2.42g	2.75g
Calibrated NiCoLet	1.88g	2.05g	1.89g	2.22g

Table 4.8: Comparison table between the initial NiCoLet model parameters and the calibrated set collected from offline optimization.

4.3.4 LSTM Neural Network

Training Process

The input growth windows were built from individual plant growth time series, or growth trajectories, in the pre-assigned subset of the collected dataset, comprised of data from 25 growth trajectories. The number of individual training inputs varied with the prediction horizon, or how many days in the future to generate an estimate, and the window size, or how many daily measurements are included in the input array.

The corresponding RGB-D images for each measurement were also noted and used by the trained CNN module for mass evaluation. Since the live version of this network

would be subject to noise from the CNN evaluation error, training on CNN-evaluated trajectories was necessary to make the LSTM robust to noisy trajectories and was made available in the LSTM training set. In summary, the training set consisted of data from ground-truth growth, using true mass measurements, and growth evaluated by CNN, using CNN evaluations of images. Lastly, the LSTM test set was only made up of data from CNN-evaluated growth to better approximate its real-world viability.

The input windows were then constructed and paired with a future biomass value to project. To handle measurement gaps and maximize training data, an exponential curve was fit to the entire trajectory to fill missing measurements, as seen in Figure 4.7. For windows composed of CNN-evaluated data, CNN evaluations were used for the window, while the ground-truth mass measurement was used at the horizon point.

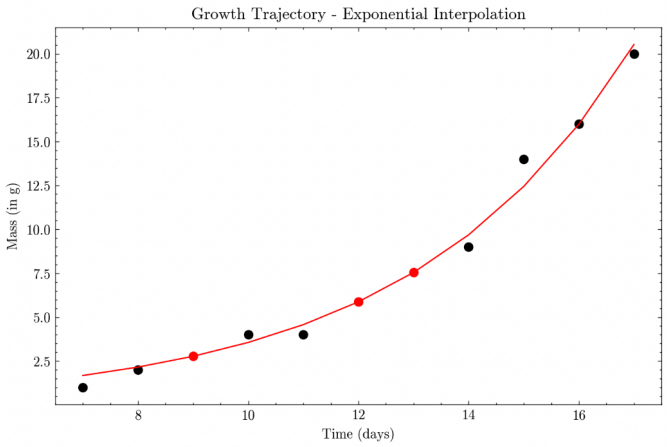

Figure 4.7: This image shows how missing data was interpolated by fitting an exponential curve and estimating the mass at a given day.

An initial LSTM network was trained as a comparison point for future analysis. The hyperparameters used in this training were:

- Training Set Composition: Ground-truth mass measurements
- Window Size: 4 Days
- Learning Rate: 5e-5
- Loss Function: MSE
- Dropout Rate: 0%

The following plots are the result of training the LSTM network to forecast a

1-day horizon. Figure 4.8 shows the actual vs. predicted scatter plot and the MSE loss curve from network training, respectively. The average error on the test set was 2.22 g, and the R^2 for the actual vs. predicted scatter plot was 0.826. The training process took 114 seconds.

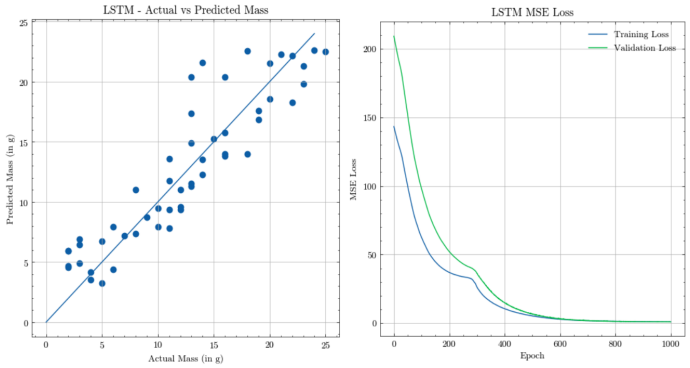

Figure 4.8: These are performance plots for the initial LSTM architecture. The left image shows the actual vs. predicted scatter plot for mass-image pairs in the test set. The right image shows the training curve for the LSTM network

Ablation Studies

This project aimed to study the model hyperparameters to evaluate their effects on the accuracy of the model prediction. The window size was left as a hyperparameter to observe trade-offs in providing more data at the cost of possible overfitting. Second, varying the prediction horizon was tested to observe how the prediction accuracy changes with an increase in the horizon length. The next analysis was on the composition of the training set to observe whether composting the training set with only the ground-truth information, only CNN evaluations, or a combination of both was optimal for training the LSTM. Additionally, typical neural network hyperparameters such as dropout rate, learning rate, and loss function were adjusted to observe how they affected prediction accuracy. To study optimal conditions and hyperparameters for LSTM training, ablation studies were conducted for each of the above analyses.

Table 4.9 shows the results of varying the composition of the training set with three options: only ground-truth training data, only CNN-evaluated training data, or

a hybrid training set that included data from both sets of trajectories. The window size was kept constant at a length of 4 days, but the prediction horizon was varied between 1, 2, 3, and 4 days.

The results show that the best composition of the training set comes from mixing CNN-evaluated data with ground-truth data. Training on CNN-evaluated data might make the model robust to noise from the CNN, while training on ground-truth data might enable the model to better understand the underlying temporal dependencies. While both of these are valuable, the combination of both might balance both approaches and provide a robust prediction.

	1-Day Horizon	2-Day Horizon	3-Day Horizon	4-Day Horizon
Only CNN Evaluations	2.19g	2.36g	2.35g	2.80g
Only Ground-Truth	2.31g	2.72g	2.62g	2.99g
Mixture Composition	2.10g	2.41g	2.21g	2.59g

Table 4.9: Average error comparison for different training set compositions.

Table 4.10 shows the results of varying the custom window size and prediction horizon parameters, with each cell showing the average error for the forecast measurements. The minimum window size was set to 2, the minimum necessary to perfectly fit an exponential curve.

In terms of window size, the table shows that there was no improvement in increasing the window size beyond 2 days. It is hypothesized from this study that only 2 days are necessary to define future growth, and any additional measurements in the window only contribute to overfitting. This result seems indicative of a system defined by a second-order Markovian process under constant environmental conditions. Although this study lacks a robust dataset to determine whether the system can be modeled as a Markov process, future analysis could enhance these findings and potentially provide better plant modeling with a Markov process design.

Additionally, while there was expected to be a clear increase in average error was expected as the prediction horizon increased, this expectation was not reflected in the testing for the 2-day window. This may have been due to the smaller sample size at larger horizon lengths, as well as less CNN-evaluation noise in the randomly selected test set. Although the expected trend is not observed in this study, further statistical analysis with a larger data set may determine whether it holds more broadly.

	1-Day Horizon	2-Day Horizon	3-Day Horizon	4-Day Horizon
2-Day Window	1.86g	2.27g	2.19g	2.21g
4-Day Window	2.10g	2.41g	2.21g	2.59g
6-Day Window	2.16g	2.44g	2.84g	3.57g

Table 4.10: Average error comparison for different window lengths.

Table 4.11 shows the results of changing the loss function. An alternative loss function, Mean Absolute Percentage Error (MAPE), was tested to observe if it would decrease relative error at the expense of minor average error gains. The window size was kept constant at a length of 2 days and the prediction horizon was varied between 1, 2, 3, and 4 days. Lastly, since comparing both loss functions on average error would likely prefer the MSE loss, the R2 value of the actual vs. predicted plot was added to observe the linear fit of the actual vs. predicted scatter plot. A lower R2 would communicate that a prediction method struggled to maintain consistent prediction over the entire range of actual mass values, signaling a plateau in a certain segment.

The results of the study are shown in Table X, where the MSE loss performed better in both categories. From this study, MSE was chosen as the optimal loss function for training the LSTM architecture due to the precision and consistency of the prediction.

	1-Day Horizon		2-Day Horizon		3-Day Horizon		4-Day Horizon	
	Error	R^2	Error	R^2	Error	R^2	Error	R^2
MSE	1.86g	0.880	2.27g	0.840	2.19g	0.800	2.21g	0.603
MAPE	2.06g	0.851	2.60g	0.785	2.46g	0.765	2.32g	0.529

Table 4.11: Average error and coefficient of determination metrics for different loss functions for LSTM training.

Table 4.12 shows the results of varying learning rates and dropout percentages. Two learning rates were tested, one that was less, 1e-5, and another that was greater, 1e-4, than the original 5e-5 learning rate. Additionally, dropout was tested at a moderate value, 15%, and a high value, 30%. Window size was kept constant at 2 days, and the prediction horizon was varied between 1, 2, 3, and 4 days.

The results, shown in Table 4.12, show that the learning rate of 5e-5 was most appropriate for the training context. Additionally, adding dropout mechanisms to the

model training significantly hindered the model performance, so dropout was set to 0

LR	Dropout	1-Day Horizon	2-Day Horizon	3-Day Horizon	4-Day Horizon
5e-5	0%	1.86g	2.27g	2.19g	2.21g
1e-4	0%	1.91g	2.31g	2.24g	2.23g
1e-5	0%	1.92g	2.36g	2.33g	2.44g
5e-5	15%	2.69g	3.20g	3.17g	3.29g
5e-5	30%	3.46g	4.26g	3.74g	3.78g

Table 4.12: Average error comparison for different combination of adjusted learning rates and network dropout percentages.

Training Result

From the studies above, the LSTM architecture was finalized with these chosen hyperparameters:

- Training Set Composition: Mixture of ground-truth and CNN-evaluated data
- Window Size: 2 Days
- Learning Rate: 5e-5
- Loss Function: MSE
- Dropout Rate: 0%

The following plots are the result of training the LSTM network to forecast a 1-day horizon. Figure 4.9 shows the actual vs. predicted scatter plot and the MSE loss curve from network training, respectively. In improving the training process, the average error decreased from 2.22 g to 2.06 g, and R^2 for the actual vs. predicted scatter plot improved from 0.826 to 0.851. The training process took slightly longer at 129 seconds. These results suggest that the new LSTM training has improved its accuracy and consistency for future forecasting.

4.3.5 Model Comparison

With all models calibrated to the training set, this project is able to compare the prediction accuracy between the tested forecasting models and select an optimal model for use in the digital twin architecture. In Table 4.13, a comparison between

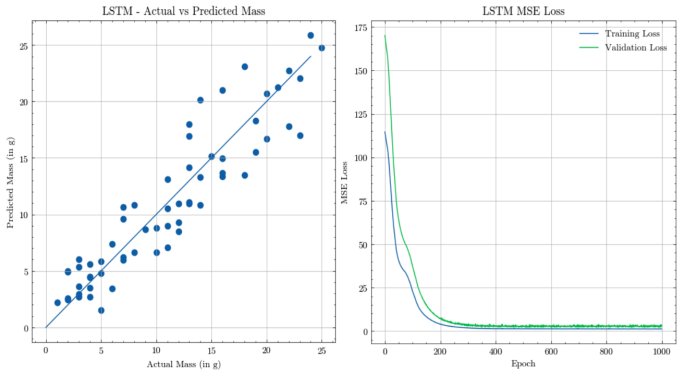

Figure 4.9: These are performance plots for the finalized LSTM architecture. The left image shows the actual vs. predicted scatter plot for mass-image pairs in the test set. The right image shows the training curve for the LSTM network.

four prediction methods is shown. Each row corresponds to a different modeling method, and the performance metrics compared are the prediction errors across different prediction horizons.

The calibrated NiCoLet model performed the best, with approximately 2 g of average error for each selected prediction horizon. Since this method was based on a scientific mathematical model, it makes sense that the model is robust to different prediction horizons if it encodes the underlying behaviors driving growth. The 2-day LSTM model had comparable performance, but showed a clear decrease in prediction accuracy as the prediction horizon was extended. Both the NiCoLet and the LSTM methods produced better forecasts than the baseline exponential model, which particularly had trouble with large prediction horizons.

Since the calibrated NiCoLet model performed best relative to other modeling options, this project used the NiCoLet model in the broader digital twin architecture.

	1-Day Hor.	2-Day Hor.	3-Day Hor.	4-Day Hor.
Exponential	2.20g	2.41g	2.61g	4.44g
Base NiCoLet	2.41g	2.50g	2.42g	2.75g
Calibrated NiCoLet	1.88g	2.05g	1.89g	2.22g
LSTM	1.86g	2.27g	2.19g	2.21g

Table 4.13: Full comparison table for the prediction accuracies of different modeling methods, using average error as a metric of accuracy.

Chapter 5

Conclusions

Particularly for small growers, hydroponic plant growth is a high-cost venture that requires skilled maintenance labor to properly start. While large-scale growers may always have the advantage of economies of scale, there are different, non-commercial benefits to enabling efficient, small-scale hydroponic production. This project developed a prototypical greenhouse for lettuce growth in schools and urban agricultural settings, with the aim of minimizing cost, automating plant management, and communicating current growth with concise and understandable metrics. Since plant growth dynamics are complex and difficult to measure, this project chose to use digital twins for this setting and produced a running simulation that adapted to a continuous feed of measured data.

In this paper, a practical digital twin architecture was outlined that could (1) measure live data, (2) communicate data from the physical system to a live simulation, and (3) maintain a simulation of the NiCoLet model capable of adapting to the incoming measurements. In order to train and test subcomponents for deployment, a dataset of ground-truth mass-image pairs was collected and allotted to different training tasks. First, a CNN was trained to estimate biomass autonomously from RGB-D images, using a refined architecture inspired by the approach taken by Buxbaum et al. Second, an update procedure was established that minimized modeling error between a previously calibrated model and the live measurement history for growing plants. By automating the cyclic process of measuring data and updating the models to reflect new data, a twin digital simulation was created that could evaluate and

forecast lettuce growth.

5.1 Future Work

This project developed a baseline digital twin architecture that can be used to project growth in a greenhouse environment. A limitation of this project was proper environmental controls, which were restricted by proprietary software for scheduling the lighting and fan fixtures used in the custom greenhouse. In retrospect, a different set of environmental condition hardware would have been selected to allow fine environmental controls, allowing for edge case testing and the optimization of conditions to maximize yield. Additional data collection and analysis are required to make the architecture robust to a set of possible conditions, instead of just the constant conditions assumed in this project.

Additionally, this digital twin pipeline is implemented specifically for lettuce plants, but further work could adapt this architecture for different hydroponic plants. As mentioned in Chapter X, different plants have available mathematical models, particularly the TOMGRO model for tomato growth, that could be easily swapped with the NiCoLet model chosen here.

Lastly, more research could improve the human-robot interaction aspect of the greenhouse design to adapt this project for educational purposes. It might be useful to have conversations with local schools, farmers, and nonprofit organizations and adapt the visual interface based on their feedback. Before deploying this robot to the target audiences mentioned in this paper, more research would need to be done to improve the educational value of the visual interface.

Appendix A

Appendix

A.1 CNN on Different Image Distances

Although CNN was trained on images taken a fixed distance from the plant bed, around 200 images were taken at different distances from the plant. These images were taken for a generalization test to check whether the CNN could generalize its predictions to images taken at different distances. Despite being taken at different distances, the image resolution and the camera used remained the same.

Although the images taken at different distances would not be suitable inputs for the trained CNN model directly, it was hypothesized that these new images could undergo an affine transformation to either zoom in or out the images. The depth image would then be separately scaled by multiplying the old depth by the ratio between the new and old distances. Lastly, a cubic interpolation method would then be used to ensure the right image resolution, and the image would receive zero padding if the process zoomed out the image. For this preprocessing, the intrinsic camera matrix was required, which was collected from the calibration function inside the Intel RealSense Python developer package.

To test the evaluation accuracy of this preprocessing, two groups of images were compared: a nonuniform distance set and a uniform distance set. The uniform distance set of images was images at the specified distance that the CNN was trained on, which was 176 mm. As previously described, this set had an RMSE of 1.56 g. These images were in a subset that was not utilized for CNN training and were unseen

by CNN during training. The nonuniform distance set were all images of distances other than the 176 mm used for training, including distances that required both zooming in and out. Despite a lesser performance than the uniform distance set, the 2.332 g RMSE for the nonuniform distance set was relatively promising given that the model was not trained on the specific heights in the new testing set. The CNN showed some generalizability to images taken at different distances, although there was some noticeable additional error.

A.2 Custom Greenhouse Part List

Table A.1 describes the hardware used for the custom greenhouse.

Function	Specific Product
FarmBot Gantry	FarmBot Genesis v1.7 (800 x 1200 frame)
Lighting Fixture	VivoSun VSFL6450 645W Grow Light
Lighting Measurement (handheld)	Apogee MQ-500
Lighting Measurement (autonomous)	Apogee SQ-420X
Environmental Readings Sensor	Aranet4 Home
Stereo Camera	Intel RealSence D405
Growth Medium	Oasis Horticubes 1”
Lettuce Species	Salvanova Butterhead
Fertilizer	Hort Americas Hydroponic Nutrients
pH and EC Meter	UIUZMAR PH EC Meter
Grow Tent Frame	YieldLab 96”x48”x78” Indoor Grow Tent
Planting Bed Frame	80/20 Extruded Aluminum
Visual Interface Webcam	Logitech Brio 101
Seedling Growth Tent	AC Infinity Growth Tent Kit [5x5]
VivoSun Circulation	VivoSun AeroWave Fans [2-Pack]
VivoSun Scheduler	VivoSun GrowHub E42A

Table A.1: This is the hardware part list for the custom greenhouse installed at Carnegie Mellon University.

A.3 NiCoLet B3 Model Definitions

Tables A.2, A.3, and A.4 describe the NiCoLet B3 parameters used in this research.

Name	Initial Value
Fresh Biomass (M_{FM})	0.001 kg/m^2[ground]
Dry Biomass (M_{DM})	0.02 kg/m^2[ground]
Vacuole Carbon (M_{cv})	0.0023 kg/m^2[ground]
Structure Carbon (M_{cs})	0.022 kg/m^2[ground]
PAR (I)	N/A [units: W/m^2[ground]]
Carbon Dioxide (C)	550 ppm
Temperature (T)	23 $°C$
Maintenance Respiration Coefficient (k)	4e-7 s^{-1}
Growth Rate Coefficient (v)	22.1 mol[C] mol^{-2}[ground]
Leaf Area Closure Parameter (a)	0.2 mol^{-1}[C] m^2[ground]

Table A.2: These are the core parameters for defining the state, model inputs, and the core system parameter studied in this research.

Name	Initial Value
Apparent Light Use Efficiency (ϵ)	0.2 mol[C] mol^{-1}[PAP]
Carbon Dioxide Transport Coefficient (σ)	0.2 ms^{-1}
Temperature Effect Parameter (c)	0.0693 $°C^{-1}$
Growth Respiration Loss Factor (θ)	0.3 mol[C] mol^{-2}[ground]
Osmotic Carbon Equivalence Coefficient (γ)	0.61 m$^3 \cdot$ Pa $\cdot$ mol^{-1}[C]
Carbon Concentration Calculation Parameter (λ)	8.3e-4 m$^3 \cdot$ mol^{-1}[C]
Slope Parameter for Photosynthesis Inhibition (s_p)	10
Slope Parameter for Growth Inhibition (s_g)	10
Threshold Parameter for Photosynthesis Inhibition (b_p)	0.2
Threshold Parameter for Growth Inhibition (b_g)	0.8
Carbon Dioxide Inhibition Point (C^*)	0.0011 mol^{-1}[C] $\cdot$ mol^3
Reference Temperature (T^*)	20 °C
Osmotic Vacuole Pressure (Π_v)	580 Pa
Regression Parameter of C/N Ratio in Vacuole (β)	6.0 m$^3 \cdot$ Pa $\cdot$ mol^{-1}[C]
Organic Matter in kg per mol[C] (η_{OMC})	0.03 kg[DM]mol^{-1}[C]
Vacuole Minerals in kg per mol[N] (η_{MMN})	0.148 kg[DM]mol^{-1}[N]

Table A.3: This table details the names and values used for the other system parameters.

Function Symbol	Function Name
Vacuole Carbon Concentration	$C_{cv}(M_{cs}, M_{cv})$
Uninhibited Photosynthesis Rate	$p(I, C)$
Photosynthesis Inhibition Function	$h_p(M_{cs}, M_{cv})$
Source Switching Function	$h_g(M_{cs}, M_{cv})$
Canopy Closure Function	$f(M_{cs})$
Specific Maintenance Respiration	$e(T)$
Maximum Growth Rate	$g(T)$

Table A.4: This table details the names and symbols for the functions used.

Bibliography

[1] Hydroponics market size, share trends analysis report by type, by crop type (tomatoes, lettuce, peppers, cucumbers, herbs, and others), by crop area, by region, and segment forecasts, 2024 - 2030. https://www.grandviewresearch.com/industry-analysis/hydroponics-market. Accessed: 2025-07-09. 2.1

[2] Conversion - ppfd to watts. https://www.apogeeinstruments.com/conversion-ppfd-to-watts/. Accessed: 2025-06-01.

[3] *The NiCoLet Lettuce Model: A Theme with Variations*, number 654, 08 2004. International Society for Horticultural Science (ISHS), Leuven, Belgium. ISBN 2406-6168. doi: 10.17660/ActaHortic.2004.654.7. URL https://doi.org/10.17660/ActaHortic.2004.654.7.

[4] Cindy Boonen. On-line measurement and modelling of dynamic plant responses to variations of the microenvironment. 2005. 2.1

[5] Nicolas Buxbaum, Johann Heinrich Lieth, and Mason Earles. Non-destructive plant biomass monitoring with high spatio-temporal resolution via proximal rgb-d imagery and end-to-end deep learning. *Frontiers in Plant Science*, 13: 758818, 2022. 2.3.4, 3.3.2

[6] Nived Chebrolu, Federico Magistri, Thomas Läbe, and Cyrill Stachniss. Registration of spatio-temporal point clouds of plants for phenotyping. *PloS one*, 16 (2):e0247243, 2021. 2.3.3

[7] Fernando A Auat Cheein and Jose Guivant. Slam-based incremental convex hull processing approach for treetop volume estimation. *Computers and Electronics in Agriculture*, 102:19–30, 2014. 2.3.3

[8] Wilma V. Davis and Catharine Weber. Growth in greenhouses: Controlled environment agriculture production, operations on the rise. https://www.ers.usda.gov/data-products/charts-of-note/chart-detail?chartId=109422. Accessed: 2025-06-01. 1.1

[9] Evan H. Delucia, Thomas W. Sasek, and Boyd R. Strain. Photosynthetic inhibition after long-term exposure to elevated levels of atmospheric carbon dioxide. *Photosynthesis Research*, 7(2):175–184, 1985. doi: 10.1007/BF00037008.

URL https://doi.org/10.1007/BF00037008.

[10] H Gijzen, E Heuvelink, H Challa, LFM Marcelis, E Dayan, S Cohen, and M Fuchs. Hortisim: A model for greenhouse crops and greenhouse climate. *II Modelling Plant Growth, Environmental Control and Farm Management in Protected Cultivation 456*, pages 441–450, 1997. 2.2

[11] W.T. Godbey. Chapter 5 - cell growth. In W.T. Godbey, editor, *Biotechnology and its Applications (Second Edition)*, pages 117–150. Academic Press, second edition edition, 2022. ISBN 978-0-12-817726-6. doi: https://doi.org/10.1016/B978-0-12-817726-6.00005-8. URL https://www.sciencedirect.com/science/article/pii/B9780128177266000058.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3.3.3

[13] Dean Holzworth, Neil I Huth, Justin Fainges, Hamish Brown, Eric Zurcher, Rogerio Cichota, Shaun Verrall, Neville I Herrmann, Bangyou Zheng, and V Snow. Apsim next generation: Overcoming challenges in modernising a farming systems model. *Environmental Modelling & Software*, 103:43–51, 2018. 2.2

[14] Dean P Holzworth, Neil I Huth, Peter G deVoil, Eric J Zurcher, Neville I Herrmann, Greg McLean, Karine Chenu, Erik J van Oosterom, Val Snow, Chris Murphy, et al. Apsim–evolution towards a new generation of agricultural systems simulation. *Environmental Modelling & Software*, 62:327–350, 2014. 2.2

[15] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 4.2.3

[16] IUNU. 2020 state of indoor farming report. https://iunu.com/resources/2020-state-of-indoor-farming-report. Accessed: 2025-06-01. 1.1

[17] Wenqing Jin. *Improving light use efficiency of lettuce in vertical farms by far-red radiation and dynamic light intensities*. PhD thesis, Wageningen University, 2022.

[18] James W Jones, Ehud Dayan, LH Allen, Herman Van Keulen, and Hugo Challa. A dynamic tomato growth and yield model (tomgro). *Transactions of the ASAE*, 34(2):663–0672, 1991. 2.2, 3.2.3

[19] J.W Jones, G Hoogenboom, C.H Porter, K.J Boote, W.D Batchelor, L.A Hunt, P.W Wilkens, U Singh, A.J Gijsman, and J.T Ritchie. The dssat cropping system model. *European Journal of Agronomy*, 18(3):235–265, 2003. ISSN 1161-0301. doi: https://doi.org/10.1016/S1161-0301(02)00107-7. URL https://www.sciencedirect.com/science/article/pii/S1161030102001077. Modelling

Cropping Systems: Science, Software and Applications. 2.2

[20] A. Juárez-Maldonado, K. De-Alba-Romenus, M. I. Ramírez-Sosa M., A. Benavides-Mendoza, and V. Robledo-Torres. An experimental validation of nicolet b3 mathematical model for lettuce growth in the southeast region of coahuila méxico by dynamic simulation. In *2010 7th International Conference on Electrical Engineering Computing Science and Automatic Control*, pages 128–133, 2010. doi: 10.1109/ICEEE.2010.5608663. 2.2.1, 3.2.3, 3.4.4, 4.3.3

[21] Antonio Juárez Maldonado, Karim De Alba Romenus, América Morales, and Marco I Ramiez-Sosa Moran. Dynamic behavior analysis of the nicolet b3 model. In *World Automation Congress 2012*, pages 1–6, 2012.

[22] Michael G Kapteyn, Jacob VR Pretorius, and Karen E Willcox. A probabilistic graphical model foundation for enabling predictive digital twins at scale. *Nature Computational Science*, 1(5):337–347, 2021. 2.3.1

[23] L. Karthikeyan, Ila Chawla, and Ashok K. Mishra. A review of remote sensing applications in agriculture for food security: Crop growth and yield, irrigation, and crop losses. *Journal of Hydrology*, 586:124905, 2020. ISSN 0022-1694. doi: https://doi.org/10.1016/j.jhydrol.2020.124905. URL https://www.sciencedirect.com/science/article/pii/S0022169420303656.

[24] Zhixian Lin, Shanye Wang, Rongmei Fu, Kuan-Chong Ting, and Tao Lin. *Data-Driven Modeling for Crop Growth in Plant Factories*, pages 101–129. Springer International Publishing, Cham, 2022. ISBN 978-3-031-03834-1. doi: 10.1007/978-3-031-03834-1_5. URL https://doi.org/10.1007/978-3-031-03834-1_5. 2.2, 2.3.1

[25] Gabriel Lindenmaier, Sean Papay, and Sebastian Padó. Efficient language modeling for low-resource settings with hybrid rnn-transformer architectures, 2025. URL https://arxiv.org/abs/2502.00617. 3.4.5

[26] Kai-Shu Ling and James Altland. Vertical farming – no longer a futuristic concept. https://www.ars.usda.gov/oc/utm/vertical-farming-no-longer-a-futuristic-concept/. Accessed: 2025-06-01. 1.1

[27] R. Linker, I. Seginer, and F. Buwalda. Description and calibration of a dynamic model for lettuce grown in a nitrate-limiting environment. *Mathematical and Computer Modelling*, 40(9):1009–1024, 2004. ISSN 0895-7177. doi: https://doi.org/10.1016/j.mcm.2004.12.001. URL https://www.sciencedirect.com/science/article/pii/S0895717704002997.

[28] I. L. López-Cruz, A. Ramírez-Arias, and A. Rojano-Aguilar. Sensitivity analysis of a dynamic growth model for greenhouse-grown lettuce (lactuca sativa l.). 38: 613–624, 2004. ISSN 1405-3195. 3.4.4

[29] Shaochun Ma, Tao Lin, Enrong Mao, Zhenghe Song, and Kuan-Chong Ting. *Sensing, data managing, and control technologies for agricultural systems*. Springer Nature, 2022.

[30] M Manida. The future of food and agriculture trends and challenges. *Agriculture & Food E-Newsletter*, 4(2):27–29, 2022. 1.1

[31] Jennifer Mathieu, Raphael Linker, Lanfang Levine, Louis Albright, A.J. Both, Roger Spanswick, Raymond Wheeler, Eileen Wheeler, David deVilliers, and Robert Langhans. Evaluation of the nicolet model for simulation of short-term hydroponic lettuce growth and nitrate uptake. *Biosystems Engineering*, 95(3): 323–337, 2006. ISSN 1537-5110. doi: https://doi.org/10.1016/j.biosystemseng.2006.07.006. URL https://www.sciencedirect.com/science/article/pii/S1537511006002431. 2.2.1

[32] Hupenyu Allan Mupambwa, Morris Fanadzo, Ernest Dube, and Nothando Dunjana. *Hydroponics in Household Vegetable Food Production*, pages 329–337. Springer Nature Singapore, Singapore, 2023. ISBN 978-981-99-2427-1. doi: 10.1007/978-981-99-2427-1_18. URL https://doi.org/10.1007/978-981-99-2427-1_18. 2.1

[33] Kazunari Nozue and Julin N. Maloof. Diurnal regulation of plant growth. *Plant, Cell & Environment*, 29(3):396–408, 2006. doi: https://doi.org/10.1111/j.1365-3040.2005.01489.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-3040.2005.01489.x. 3.2.5

[34] Government of the Netherlands. Agriculture and horticulture. https://www.government.nl/topics/agriculture/agriculture-and-horticulture. Accessed: 2025-07-01. 1.1

[35] Vaibhav Palande, Adam Zaheer, and Kiran George. Fully automated hydroponic system for indoor plant growth. *Procedia Computer Science*, 129:482–488, 2018. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2018.03.028. URL https://www.sciencedirect.com/science/article/pii/S1877050918302473. 2017 INTERNATIONAL CONFERENCE ON IDENTIFICATION,INFORMATION AND KNOWLEDGEIN THE INTERNET OF THINGS. 2.1

[36] L. Poulet, G.D. Massa, R.C. Morrow, C.M. Bourget, R.M. Wheeler, and C.A. Mitchell. Significant reduction in energy for plant-growth lighting in space using targeted led lighting and spectral manipulation. *Life Sciences in Space Research*, 2:43–53, 2014. ISSN 2214-5524. doi: https://doi.org/10.1016/j.lssr.2014.06.002. URL https://www.sciencedirect.com/science/article/pii/S2214552414000327.

[37] Yuda Prasetia, Aji Gautama Putrada, and Andrian Rakhmatsyah. Evaluation of iot-based grow light automation on hydroponic plant growth. *J. Ilm. Tek.*

Elektro Komput. dan Inform, 7(2):314, 2021.

[38] S Ragaveena, A Shirly Edward, and U Surendran. Smart controlled environment agriculture methods: A holistic review. *Reviews in Environmental Science and Bio/Technology*, 20(4):887–913, 2021. 1.1

[39] Laura Reiley. Netherlands is the second-largest exporter of agricultural products. https://www.washingtonpost.com/business/interactive/2022/netherlands-agriculture-technology/. Accessed: 2025-07-01. 1.1

[40] Martin Rutzinger, Arun Kumar Pratihast, S Oude Elberink, and George Vosselman. Detection and modelling of 3d trees from mobile laser scanning data. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci*, 38:520–525, 2010. 2.3.3

[41] Mahmoud Sabzian, Ali Rahimikhoob, Mahmoud Mashal, Sasan Aliniaeifard, and Tahmine Dehghani. Comparison of water productivity and crop performance in hydroponic and soil cultivation using aquacrop software* a case study of lettuce cultivation in pakdasht, iran. *Irrigation and Drainage*, 70(5):1261–1272, 2021. 2.2

[42] Ramin Shamshiri, Fatemeh Kalantari, KC Ting, Kelly R Thorp, Ibrahim A Hameed, Cornelia Weltzien, Desa Ahmad, and Zahra Mojgan Shad. Advances in greenhouse automation and controlled environment agriculture: A transition to plant factories and urban agriculture. *International Journal of Agricultural and Biological Engineering*, 11(1):1–22, 2018.

[43] Redmond Ramin Shamshiri, James W Jones, Kelly R Thorp, Desa Ahmad, Hasfalina Che Man, and Sima Taheri. Review of optimum temperature, humidity, and vapour pressure deficit for microclimate evaluation and control in greenhouse cultivation of tomato: a review. *International agrophysics*, 32(2):287–302, 2018. 2.2

[44] Anurag Shrivastava, Chinmaya Kumar Nayak, R. Dilip, Soumya Ranjan Samal, Sandeep Rout, and Shaikh Mohd Ashfaque. Automatic robotic system design and development for vertical hydroponic farming using iot and big data analysis. *Materials Today: Proceedings*, 80:3546–3553, 2023. ISSN 2214-7853. doi: https://doi.org/10.1016/j.matpr.2021.07.294. URL https://www.sciencedirect.com/science/article/pii/S2214785321051531. SI:5 NANO 2021. 2.1

[45] Pascalle C. Smith, Pierluigi Calanca, and Jürg Fuhrer. A simple scheme for modeling irrigation water requirements at the regional scale applied to an alpine river catchment. *Water*, 4(4):869–886, 2012. ISSN 2073-4441. doi: 10.3390/w4040869. URL https://www.mdpi.com/2073-4441/4/4/869. 2.2

[46] T R Sreedevi and M.B Santosh Kumar. Digital twin in smart farming: A categorical literature review and exploring possibilities in hydroponics. In *2020 Advanced Computing and Communication Technologies for High Performance*

Applications (ACCTHPA), pages 120–124, 2020. doi: 10.1109/ACCTHPA49271.2020.9213235.

[47] TR Sreedevi and MB Santosh Kumar. Digital twin in smart farming: A categorical literature review and exploring possibilities in hydroponics. *2020 advanced computing and communication technologies for high performance applications (ACCTHPA)*, pages 120–124, 2020. 2.3.2

[48] Jia Sun, Lunche Wang, Shuo Shi, Zhenhai Li, Jian Yang, Wei Gong, Shaoqiang Wang, and Torbern Tagesson. Leaf pigment retrieval using the prosail model: Influence of uncertainty in prior canopy-structure information. *The Crop Journal*, 10(5):1251–1263, 2022. ISSN 2214-5141. doi: https://doi.org/10.1016/j.cj.2022.04.003. URL https://www.sciencedirect.com/science/article/pii/S2214514122000873. Crop phenotyping studies with application to crop monitoring. 2.2

[49] Chuyun Tan, Shanhong Zhang, Yu Guo, and Yang Wang. Analysis and evaluation of a dynamic model for greenhouse lettuce growth. *Spanish Journal of Agricultural Research*, 20(4):e0904, Oct. 2022. doi: 10.5424/sjar/2022204-18658. URL https://sjar.revistas.csic.es/index.php/sjar/article/view/18658.

[50] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, pages 10096–10106. PMLR, 2021. 4.2.3

[51] Fei Tao, He Zhang, Ang Liu, and A. Y. C. Nee. Digital twin in industry: State-of-the-art. *IEEE Transactions on Industrial Informatics*, 15(4):2405–2415, 2019. doi: 10.1109/TII.2018.2873186. 2.3

[52] Battsetseg Tuvdendorj, Bingfang Wu, Hongwei Zeng, Gantsetseg Batdelger, and Lkhagvadorj Nanzad. Determination of appropriate remote sensing indices for spring wheat yield estimation in mongolia. *Remote Sensing*, 11(21):2568, 2019.

[53] SH Van Delden, M SharathKumar, M Butturini, LJA Graamans, E Heuvelink, M Kacira, E Kaiser, RS Klamer, L Klerkx, G Kootstra, et al. Current status and future challenges in implementing and upscaling vertical farming systems. *Nature Food*, 2(12):944–956, 2021. 2.1

[54] Marc W van Iersel. Optimizing led lighting in controlled environment agriculture. *Light emitting diodes for agriculture: smart lighting*, pages 59–80, 2017.

[55] Bram HE Vanthoor. *A model-based greenhouse design method*. Wageningen University and Research, 2011. 2.2

[56] Roberto S. Velazquez-Gonzalez, Adrian L. Garcia-Garcia, Elsa Ventura-Zapata, Jose Dolores Oscar Barceinas-Sanchez, and Julio C. Sosa-Savedra. A review on hydroponics and the technologies associated for medium- and small-scale operations. *Agriculture*, 12(5), 2022. ISSN 2077-0472. doi: 10.3390/agriculture12050646.

URL https://www.mdpi.com/2077-0472/12/5/646. 2.1

[57] B.A. Wielicki, B.R. Barkstrom, B.A. Baum, T.P. Charlock, R.N. Green, D.P. Kratz, R.B. Lee, P. Minnis, G.L. Smith, Takmeng Wong, D.F. Young, R.D. Cess, J.A. Coakley, D.A.H. Crommelynck, L. Donner, R. Kandel, M.D. King, A.J. Miller, V. Ramanathan, D.A. Randall, L.L. Stowe, and R.M. Welch. Clouds and the earth's radiant energy system (ceres): algorithm overview. *IEEE Transactions on Geoscience and Remote Sensing*, 36(4):1127–1141, 1998. doi: 10.1109/36.701020. 2.2

[58] DH Willits. Cooling fan-ventilated greenhouses: a modelling study. *Biosystems engineering*, 84(3):315–329, 2003.

[59] Sheng Wu, Weiliang Wen, Boxiang Xiao, Xinyu Guo, Jianjun Du, Chuanyu Wang, and Yongjian Wang. An accurate skeleton extraction approach from 3d point clouds of maize plants. *Frontiers in plant science*, 10:248, 2019. 2.3.3

[60] Sheng Wu, Weiliang Wen, Boxiang Xiao, Xinyu Guo, Jianjun Du, Chuanyu Wang, and Yongjian Wang. An accurate skeleton extraction approach from 3d point clouds of maize plants. *Frontiers in Plant Science*, Volume 10 - 2019, 2019. ISSN 1664-462X. doi: 10.3389/fpls.2019.00248. URL https://www.frontiersin.org/journals/plant-science/articles/10.3389/fpls.2019.00248. 2.3.3

[61] Minglan Xiong and Huawei Wang. Digital twin applications in aviation industry: A review. *The International Journal of Advanced Manufacturing Technology*, 121(9):5677–5692, 2022. 2.3

[62] Dan Xu, Jingjing Chen, Ba Li, and Juncheng Ma. Improving lettuce fresh weight estimation accuracy through rgb-d fusion. *Agronomy*, 13(10), 2023. ISSN 2073-4395. doi: 10.3390/agronomy13102617. URL https://www.mdpi.com/2073-4395/13/10/2617.

[63] Emre Yildiz, Charles Møller, and Arne Bilberg. Virtual factory: Digital twin based integrated factory simulations. *Procedia CIRP*, 93:216–221, 2020. ISSN 2212-8271. doi: https://doi.org/10.1016/j.procir.2020.04.043. URL https://www.sciencedirect.com/science/article/pii/S2212827120306077. 53rd CIRP Conference on Manufacturing Systems 2020. 2.3