Canadian Journal of
Plant Science

OPEN ACCESS | Research Article

Fusarium head blight detection, spikelet estimation, and severity assessment in wheat using 3D convolutional neural networks

Oumaima Hamila[a], **Christopher J. Henry**[a,b], **Oscar I. Molina**[c], **Christopher P. Bidinosti**[a,d], **and Maria Antonia Henriquez**[c]

[a]Department of Applied Computer Science, The University of Winnipeg, Winnipeg, MB, Canada; [b]Department of Computer Science, The University of Manitoba, Winnipeg, MB, Canada; [c]Morden Research and Development Centre, Agriculture and Agri-Food Canada, Morden, MB, Canada; [d]Department of Physics, The University of Winnipeg, Winnipeg, MB, Canada

Corresponding authors: **Christopher J. Henry** (email: christopher.henry@umanitoba.ca); **Maria Antonia Henriquez** (email: mariaantonia.henriquez@agr.gc.ca)

Abstract

Fusarium head blight (FHB) is one of the most significant diseases affecting wheat and other small-grain cereals worldwide. Developing FHB-resistant cultivars is critical but requires field and greenhouse disease assessment, which are typically laborious and time consuming. In this work, we developed automated applications based on three-dimensional (3D) convolutional neural networks (CNNs) that detect FHB symptoms expressed in wheat, estimate the total number of spikelets versus the total number of infected spikelets on a wheat head, and subsequently calculate FHB severity index. Such tools are an important step toward the creation of automated and efficient phenotyping methods. The data used to generate the results are 3D point clouds consisting of four colour channels—red, green, blue (RGB), and near-infrared (NIR)—collected using a multispectral 3D scanner. Our 3D CNN models for FHB detection achieved 100% accuracy. The influence of the multispectral information on performance was evaluated; the results showed the dominance of the RGB channels over both the NIR (720 nm peak wavelength) and the NIR plus RGB channels combined. Our best 3D CNN models for estimation of total and infected number of spikelets achieved mean absolute errors (MAEs) of 1.13 and 1.56, respectively. Our best 3D CNN models for FHB severity estimation achieved 8.6 MAE. A linear regression analysis between the visual FHB severity assessment and the FHB severity predicted by our 3D CNN showed a significant correlation.

Key words: fusarium head blight (FHB), wheat, FHB severity detection, 3D convolutional neural network (3D CNN), multispectral point cloud, assessment

1. Introduction

Fusarium head blight (FHB) is a devastating fungal disease caused by a variety of species within the *Fusarium* genus that affect cereal crops like wheat, barely, and oats (Ghimire et al. 2020; Khan et al. 2020; Chin et al. 2023). The fungus infects the spikes[1] during the flowering stage (Sakuma et al. 2019), causing premature grain shrivelling and bleaching, leading to a significant loss in yield quality and quantity. Moreover, trichothecene mycotoxins may accumulate in the infected kernels, which can cause acute toxicity to both humans and animals (Ferrigo et al. 2016). Disease management strategies such as crop rotation, tillage, variety selection, and fungicide use help reduce the impact of FHB. Developing wheat cultivars with resistance to FHB is a high priority (Dhariwal et al. 2020), but still relies on time-consuming, subjective human assessment. This is further compounded by the fact that FHB is notoriously difficult to screen effectively in a field situation because FHB resistance is under the control of a few major genes that work in concert with an unknown number of minor genes (Anderson et al. 2001; Buerstmayr and Buerstmayr 2015). In addition, numerous factors can influence disease development in the grain, including cultural practices, spike morphology, canopy density, plant height, rainfall, relative humidity, temperature, and host plant genetic resistance (Buerstmayr et al. 2020). The considerable role of environmental factors coupled with the quantitative nature of the genetics for resistance creates serious challenges for phenotyping, which make it both highly time- and resource-intensive (Anderson 2007). To effectively evaluate breeding materials, screening must be done in replicated nurseries in

[1] A wheat head (also known as a spike) consists of a number of spikelets, and a spikelet consists of florets that could develop into 1–3 grains.

multiple environments, because even with inoculated fusarium screening nurseries, susceptible lines can elude fusarium infection.

The aforementioned issues necessitate the creation of automated tools that can replace the arduous tasks of visually identifying and characterizing FHB symptoms for thousands of wheat plants in a growth chamber or test plot. Advances in imaging technology and remote sensing (Teke et al. 2013; Lu et al. 2020)allow the acquisition of large datasets of plant and crop images (Lu and Young 2020) that can be analyzed via machine learning (ML) techniques (Jordan and Mitchell 2015). ML is quickly becoming an important part of agricultural and plant science research (Harfouche et al. 2023), with a wide variety of applications (Ferentinos 2018; Liu et al. 2020; Singh et al. 2020). Several groups have begun exploring the detection of FHB in wheat via ML, primarily using red, green, blue (RGB) images (Qiu et al. 2019; Zhang et al. 2019, 2022; Gu et al. 2021; Gao et al. 2022*a*, 2022*b*; Mao et al. 2023) but also with hyperspectral images (Barbedo et al. 2015; Huang et al. 2020). These are still early days, however, and given the challenges and limitations associated with each approach (including data quality, a reliance on augmented data, and significant preprocessing), this remains an open problem with much yet to explore. For example, to the best of our knowledge, there are currently no studies in the literature that automate the counting of spikelets or the determination of FHB severity in wheat using three-dimensional (3D) data such as point clouds (PCs). The use of 3D data is growing in popularity (Vázquez-Arellano et al. 2016), owing to the increased amount of information such as object shape and volume (Ma et al. 2003) available for ML models to make decisions. In particular, PCs, which are collections of points in 3D space along with associated colour information (e.g., measured RGB values), can provide much more detailed representations of an object's edges, surfaces, and textures than standard two-dimensional (2D) images. The use of PCs in ML and deep learning applications is established (Charles et al. 2017; Klokov and Lempitsky 2017; Li et al. 2018; Shi et al. 2019) but still not as broadly practiced as it could be due mostly to data scarcity and the added effort in utilizing 3D data.

We address both these challenges in this study, creating a bespoke labelled dataset of wheat PCs and developing efficient models for data preprocessing and FHB detection and characterization. The publicly available dataset (called the University of Winnipeg–Morden Research and Development Centre 3D WHEAT Dataset or UW-MRDC 3D WHEAT Dataset for short (Hamila et al. 2023*a*)) was acquired with a commercial multispectral scanner (Phenospex PlantEye F500), which produces a PC with RGB and near-infrared (NIR) intensities at each point. The dataset consists of two separate collections (called Dataset I and Dataset II); both comprising water-controlled (WC) (i.e., healthy) and FHB-infected PCs of wheat, but each used for different applications as discussed further in the following section. In regard to algorithm development, the main contributions of this work include (i) graphics processing unit (GPU)-enabled preprocessing to shift, scale, and reorder the PC data; (ii) a 3D CNN model for FHB detection; and (iii) two different approaches to determine FHB severity in a wheat head using 3D CNN models that either count spikelets (total versus infected) or perform a direct estimation.

2. Materials and methods

2.1. Methodology overview

The overall procedure that was designed and developed to conduct this study is shown in Fig. 1. It consists of three major systems: dataset creation, data preprocessing, and model creation for FHB detection and severity estimation in wheat using 3D CNNs. The dataset creation process consists of the three steps depicted in Fig. 1: data preparation (A); data acquisition (B); and data naming and labelling (C). The resulting dataset, called UW-MRDC 3D WHEAT (Hamila et al. 2023*a*), consists of two separate collections (Dataset I and Dataset II) that are used for different parts of this study. Following dataset creation is data preprocessing (D), during which data samples that are multispectral PCs were transformed into multispectral 3D images, whose representation is compatible with CNNs. Finally, following data preprocessing is model creation, in which 3D CNNs were developed and trained to automate the tasks of FHB detection (E); total number of spikelets estimation (F); total number of infected spikelets estimation (G); and severity assessment in wheat (H). For the development of the (E) and (F) applications, Dataset I was used, whereas Dataset II was used for (G) and (H).

The order in which the 3D CNN models are applied is depicted in Fig. 2. The pipeline begins with the 3D CNN for FHB detection, which predicts whether the input PC sample represents diseased or healthy wheat. If the sample is healthy, the input will be processed by the 3D CNN for the total number of spikelets estimation. Otherwise, the sample will be processed either through Method 1 or Method 2. In Method 1, the sample will be processed by both the 3D CNN for total number of spikelets estimation and the 3D CNN for total number of infected spikelets estimation; then, the FHB severity percentage will be the ratio of these two estimations. Whereas in Method 2, the sample will be directly processed through the 3D CNN for FHB assessment, which will directly estimate the FHB severity percentage of the sample.

Two methods for severity assessment were created due to our incremental ML model development process. At the onset, it was not certain that an ML regression model could directly produce a severity assessment with low mean absolute error (MAE). As a result, the first experiment was to determine whether a model could reasonably count the number of spikelets on a wheat head. Once successful, the next logical step was to determine whether a model could be developed to count only the number of infected spikelets. Together, the output of these two models was used to produce a severity assessment. This success led us to investigate next whether a model could then produce a severity assessment directly, thereby skipping the counting models all together. For completeness, we have presented the results of all these approaches.

Fig. 1. Methodology overview of this study that consists of three major components: (i) dataset creation (A, B, and C), (ii) data preprocessing with CUDA (D), and (iii) creation of detection and estimation three-dimensional (3D) convolutional neural network (CNN) models (E, F, G, and H). These methods were created to achieve fusarium head blight (FHB) detection (E), spikelet estimation (F and G), and severity assessment (H) in wheat using 3D CNN models and multispectral point cloud (PC) data. RGB, red, green, blue; NIR, near-infrared; WC, water controlled.

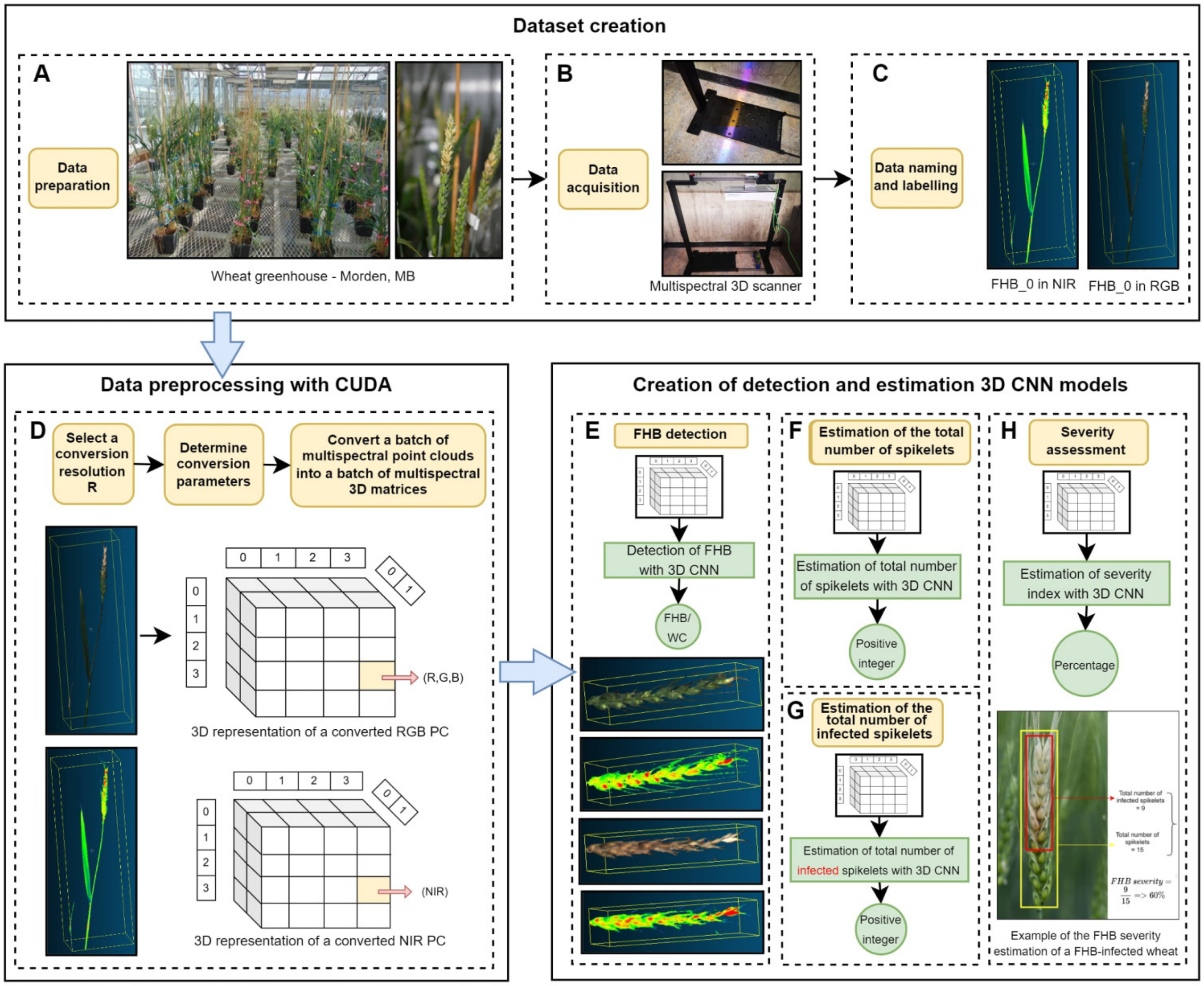

Dataset creation
A
Data preparation
Wheat greenhouse - Morden, MB
B
Data acquisition
Multispectral 3D scanner
C
Data naming and labelling
FHB_0 in NIR
FHB_0 in RGB
Data preprocessing with CUDA
D
Select a conversion resolution R
Determine conversion parameters
Convert a batch of multispectral point clouds into a batch of multispectral 3D matrices
0 1 2 3
(R,G,B)
3D representation of a converted RGB PC
(NIR)
3D representation of a converted NIR PC
Creation of detection and estimation 3D CNN models
E
FHB detection
Detection of FHB with 3D CNN
FHB/ WC
F
Estimation of the total number of spikelets
Estimation of total number of spikelets with 3D CNN
Positive integer
G
Estimation of the total number of infected spikelets
Estimation of total number of infected spikelets with 3D CNN
Positive integer
H
Severity assessment
Estimation of severity index with 3D CNN
Percentage
Total number of infected spikelets = 9
Total number of spikelets = 15
FHB severity = 9/15 => 60%
Example of the FHB severity estimation of a FHB-infected wheat

Fig. 2. Pipeline overview of the three-dimensional convolutional neural network models for fusarium head blight (FHB) detection and severity estimation.

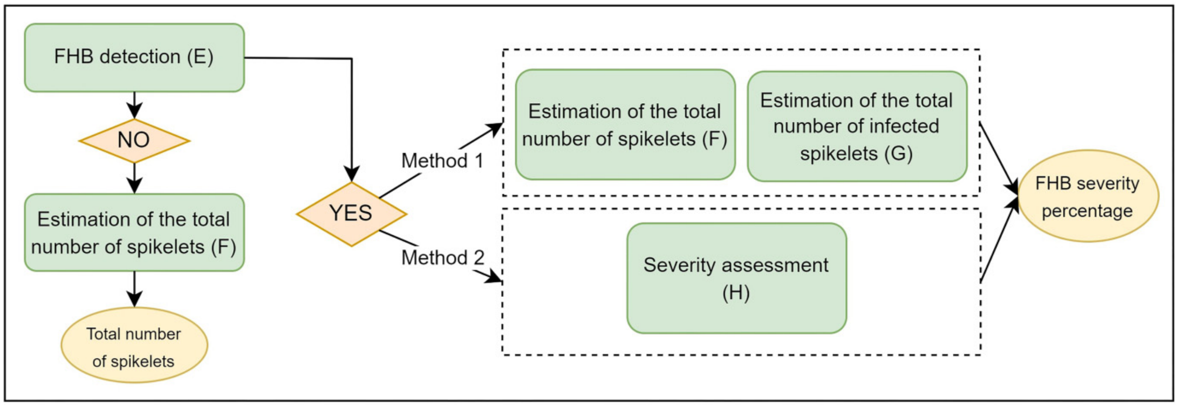
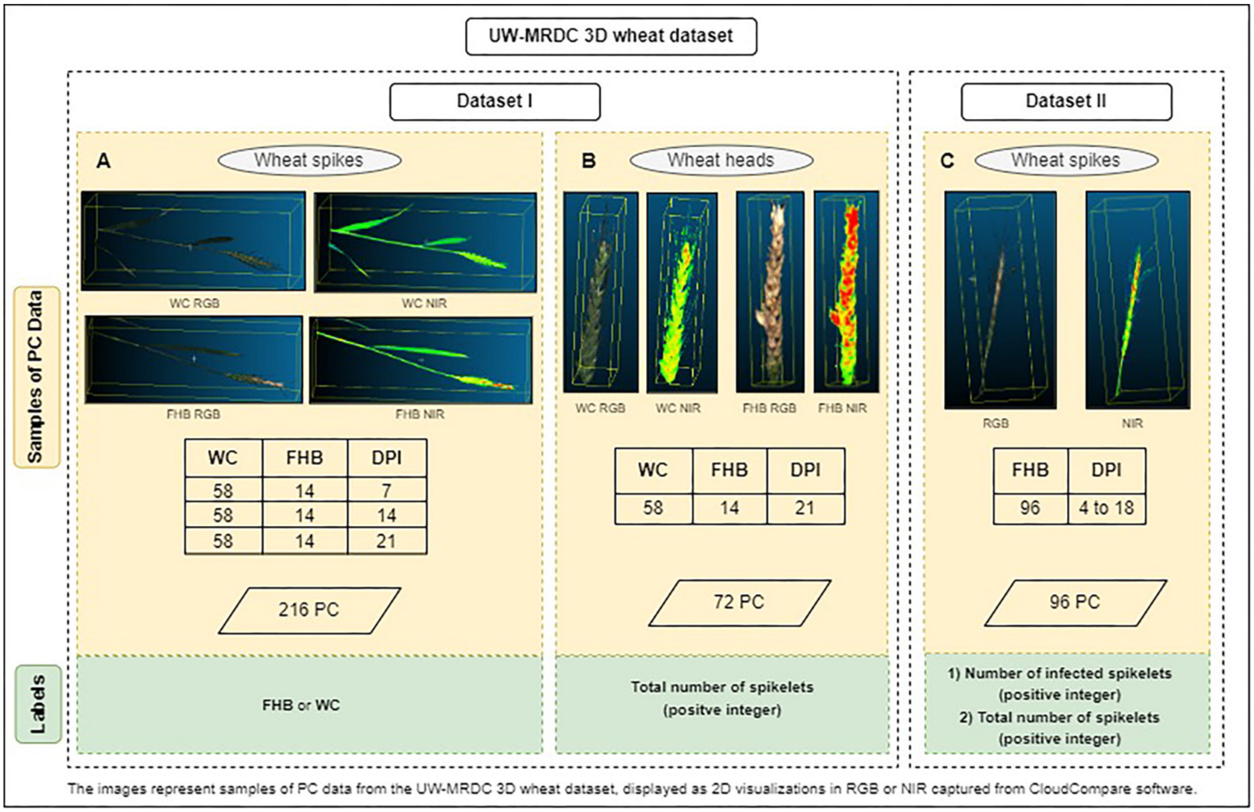

FHB detection (E)
NO
Estimation of the total number of spikelets (F)
Total number of spikelets
YES
Method 1
Method 2
Estimation of the total number of spikelets (F)
Estimation of the total number of infected spikelets (G)
Severity assessment (H)
FHB severity percentage

Fig. 3. Overview of the content and specifications of the University of Winnipeg–Morden Research and Development Centre 3D WHEAT Dataset (UW-MRDC 3D WHEAT dataset) consisting of Dataset I and II. Dataset I consists of two collections of point cloud (PC) data: wheat spikes (A) and wheat heads (B), and Dataset II consists of one collection of PC data: wheat spikes (C). The tables give the number of water-controlled (WC) and fusarium head blight (FHB) PCs at each DPI or range of DPIs (far right), with the total number given in the boxes below. RGB, red, green, blue; NIR, near-infrared; DPI, days post-inoculation.



WC	FHB	DPI
58	14	7
58	14	14
58	14	21

WC	FHB	DPI
58	14	21

FHB	DPI
96	4 to 18

The images represent samples of PC data from the UW-MRDC 3D wheat dataset, displayed as 2D visualizations in RGB or NIR captured from CloudCompare software.

2.2. UW-MRDC 3D WHEAT dataset creation

2.2.1. Dataset overview

Figure 3 illustrates the content of the UW-MRDC 3D WHEAT dataset broken into the two collections Dataset I and Dataset II. The main difference between the two collections is the methods used during the data preparation phase, which is described in detail in Section 2.2.2. Dataset I is further divided into images of wheat spikes versus wheat heads, as shown in Fig. 3, for the purpose of different applications as described below.

Dataset I (A) represents wheat spikes. It was created to achieve the task of FHB detection in wheat. The dataset was acquired by scanning 72 wheat plants, of which 14 were inoculated and 58 were kept WC, at three different growth stages. All 72 plants were captured at 7, 14, and 21 days post-inoculation (DPI), representing the growth stages of 7 days after Zadoks 65 (7 days after anthesis), Zadoks 73 (early milk), and Zadoks 83 (early dough) (Zadoks et al. 1974), respectively. Plants were scanned on different dates to capture the development of disease symptoms over time. Early FHB symptoms were recorded when at least one wheat spikelet turned yellow or pinkish and became distinguishable from the other green spikelets, and, as time went by, the disease kept developing and more spikelets got infected in a wheat head. The final Dataset I for wheat spikes consists of 216 labelled PCs, where each PC is labelled either FHB or WC.

Dataset I (B) represents wheat heads. It was created to achieve the task of estimating the total number of spikelets; therefore, the scans were focused only on the wheat spikes (also called wheat heads). The data were acquired by cutting the wheat stems of the 72 wheat plants at 21 DPI and then scanning the remaining heads. Thus, the dataset consists of 72 PCs, each labelled by a positive integer in the range [7, 22].

Dataset II (C) for wheat spikes was created to achieve the tasks of estimating the total number of infected spikelets and the FHB severity. Therefore, only FHB-diseased wheat with visible symptoms was captured at different DPIs ranging from 4 to 18 DPI, as shown in Table (C) in Fig. 3. The final dataset consists of 96 PCs, each of which is labelled with two positive integer values, the first of which indicates the total number of spikelets and ranges between [13, 21] and the second of which reflects the number of infected spikelets in a wheat head and ranges between [2, 15].

2.2.2. Sample preparation

FHB Dataset I and Dataset II were created by delivering a macroconidia suspension into bilateral florets of a wheat spike and counting the number of spikelets blighted after a determined time under controlled-environment conditions. Furthermore, a large number of *Fg* isolates were used as discussed below. This type of trial is very labour demanding, time consuming, and non-trivial to run.

The plant material used for Dataset I included the Canada Western Red Spring wheat cultivar 5602HR and CDC Teal. The 5602HR cultivar is moderately resistant to FHB, and CDC Teal is susceptible. The plant material used for Dataset II included only the wheat cultivar 5602HR. Planting and inoculation methods are identical to those described by Nilsen et al. (2020). A 3-acetyldeoxynivalenol (3-ADON) producing isolate of *Fg* (Henriquez Spring Wheat (HSW)-15-39), obtained from the HSW collection of *Fusarium* isolates, was used for Dataset I. For Dataset II, ten 3-ADON and nine 15-ADON producing isolates of *Fg* were used in this study. All isolates from the HSW culture collection are available upon request under a material transfer agreement.

Seeds were sown in 3.5 in. pots with a mixture of 50% Sunshine soilless #5 mix (manufactured by Sun Gro Horticulture) and 50% soil, plus 6 g of slow-release Osmocote 14-14-14 fertilizer (manufactured by the Scotts Company). Plants were grown in controlled-environment cabinets with 16 h of light at 22 °C and 8 h of darkness at 15 °C. The bilateral florets of a spikelet positioned at the fifth spikelet in the upper part of a spike were inoculated at 50% anthesis with a 10 μL of *Fg* macroconidia suspension (5×10^4 macroconidia/mL), which was performed between the lemma and palea using a micropipette. Control plants were treated with sterile water. Inoculated plants were covered with a plastic bag for 48 h to promote infection. FHB severity was calculated by counting the number of spikelets showing disease symptoms within each spike at 7, 14, and 21 DPI.

2.2.3. Data acquisition and preprocessing

All the wheat plants in this work were scanned using the Phenospex PlantEye F500 multispectral scanner, which provides 3D representations of plants via PCs. A wheat plant container is placed beneath the PlantEye, which, once activated, begins scanning laterally and emitting multispectral light beams onto the plant. The reflections of those beams are acquired to form a 3D representation of the wheat plant that include intensities of the four different colour bands (RGB and NIR). Multispectral information and 3D representation are then combined into a single PC. The spectral range of the PlantEye scanner is ∼450–750 nm with peak wavelengths for its four LEDs (blue, green, red, and NIR) at 460, 545, 635, and 720 nm, respectively.

The structure of the data output from the Phenospex is not immediately compatible for use with 3D CNNs. Preprocessing is required to shift, scale, and reorder the data first. The details of these steps—along with the strategies we employed for their efficient execution on a GPU—are given in Appendix A. A brief description of the pertinent components of the CUDA parallel computing platform is also included there.

2.3. Model development

This section presents the approach to develop all the models that were used to generate the reported results. In each case, the development process followed the same approach. As a result, we discuss each component of the model development process below and highlight individual differences where necessary. Specifically, four types of models were created to automate the following tasks: detection of FHB (DFHB), estimation of the total number of spikelets (ETNS), estimation of the total number of infected spikelets (ETNIS), and FHB severity assessment (FHBSA). Notice that the models for ETNS and ETNIS are used to calculate a severity percentage, while the last model (FHBSA) outputs this value directly. All of these models were developed to accept 3D PCs, transformed into 3D images (see Section 2.2.3 and Appendix A), as input. Here, the focus is on developing automated solutions for FHB detection and severity estimation, but the overall methodology can be generalized to other classification and regression problems. A nice primer on the use of artificial intelligence for plant digital phonemics is given by Harfouche et al. (2023). Finally, the following subsections detail the methods that were used for the development of the four reported models.

2.3.1. Monitored grid search

In our study, 3D CNNs were developed from scratch, as opposed to starting from off-the-shelf models such as Resnet (He et al. 2016) or DenseNet (Huang et al. 2017). A grid search over the number of layers and the number of neurons per layer was conducted. The objective of the grid search was to find the optimal 3D CNN architecture that produces the highest accuracy on the task at hand. The layers employed to build the models were 3D convolution layers, 3D max pooling layers, and densely-connected (or dense) layers, and the search space (denoted by curled braces) used to determine the optimal number of layers and neurons was the following:

- {3,4,5,6}: Search space of the number of 3D convolution and 3D max pooling layers. The last layer of 3D convolution before the densely-connected layers is not followed by a 3D max pooling layer.
- {1,2,3,4,5,6}: Search space of the number of densely-connected layers.
- {128,64,32,16,8}: Search space of the number of neurons per layer. The last densely-connected layer always has one neuron.

However, with these sets of variables, the number of possible combinations is 380 835 000 networks, which is too large to search exhaustively. Thus, a monitored grid search was employed as an alternative to training only a small number of 3D CNN models. The monitored grid search worked by randomly generating a batch of 20 networks at a time, such that a 5-fold cross-validation (CV) (Refaeilzadeh et al. 2009) was performed on each network in the batch, and then the top three networks that achieved the highest average CV accuracy out of the 20 networks were retrained on the training set and evaluated on the test set.

2.3.2. Datasets characteristics

Three different dataset versions were used to train the CNN networks presented here. The datasets were obtained by converting PCs into 3D images using different resolution factors for each task. Specifically, DFHB used a resolution factor of $R = 1$ and the remaining tasks used $R = 1.5$. Moreover, the datasets differed by their voxel information, which are defined as follows:

- The 3D wheat-plant images in RGB (3DWP_RGB): In this dataset, the voxels of the 3D images contained RGB colour information (three channels).
- The 3D wheat-plant images in NIR (3DWP_NIR): In this dataset, the voxels of the 3D images contained NIR colour information (one channel).
- The 3D wheat-plant images in RGB + NIR (3DWP_RGB_NIR): In this dataset, the voxels of the 3D images contained the RGB + NIR colour information (four channels).

3D images within the datasets had different sizes, such that the width, height, and depth values corresponding to the 3D images dimensions were within [25, 237], [85, 378], and [14, 384], respectively. Since CNNs require input samples of a fixed size, resizing all the 3D images within the datasets to the same size was required. The easiest option was to resize every 3D image to the maximum size, which corresponded to 237 × 378 × 384 voxels (vx). However, this method raised the volume of the data tremendously, such that each resized 3D image contained 34 401 024 vx. Training the models on big volume data consumes too much time and resources. Thus, a smaller fixed size was determined such that, any batch of resized 3D images could fit in the GPU memory along with any of the aforementioned model parameters.

For DFHB, it was important to keep as many relevant voxels as possible when resizing since the real height of a scanned wheat plant is an important feature for the model. Therefore, a fixed size was determined by fixing the height to 300 and by calculating the width and depth via the average aspect ratios of the images in the datasets. As a result, the data samples were all rescaled to a fixed size equal to 75 × 300 × 95 vx while maintaining their respective original aspect ratios. This means that a 3D image was rescaled to the highest possible size that preserved the initial height–width proportion, preserved the height–depth proportion, and was contained within the 75 × 300 × 95 vx envelope. The rescaled images were then zero-padded to 75 × 300 × 95 vx. Similarly, the images were scaled to 161 × 51 × 93 for ETNS and 227 × 70 × 111 for ETNIS and FHBSA.

2.3.3. Labels

The labels for DFHB were binary values indicating the presence or absence of FHB; the labels for ETNS were integers that varied between 7 and 22, indicating the total number of spikelets; the labels for ETNIS were integers between 2 and 15, giving the total number of infected spikelets; and the labels for FHBSA were real values ranging from 0.923 (92.3%) to 0.111 (11.1%) that were the FHB severity assessments.

2.3.4. 5-fold cross validation

Prior to training the 20 models, the data samples were divided into training and test sets. For DFHB and ETNS, 90% of the data were assigned to the training set with the remainder placed into the test set. For tasks ETNIS and FHBSA, an 80/20 split was used. For all experiments, we ensured that the training set and the test set had the same class distribution with respect to FHB and WC classes. Next, the training samples were further split into five folds that had the same class distribution as the training set to perform CV, such that each fold consisted of 20% of the training data. Then, 5-fold CV was applied by training the models on four training folds and validating them on the validation fold. The top three model architectures that achieved the highest average CV accuracy were retrained on the entire training set and evaluated on the test set, where "average CV accuracy" refers to the average accuracy value achieved by the network trained on each fold of the five CV folds.

2.3.5. Model architectures

The architectures of each of the 20 DFHB and five ETNS models that were constructed by the monitored grid search are presented in Tables 1 and 2, respectively. For ETNIS and FHBSA, a hundred 3D CNN models were trained in total, and, due to this large number of tested models, only the architectures of the top three models will be discussed. These are depicted in Tables 3 and 4 for ETNIS and FHBSA, respectively.

The number of convolutional neurons refers to the number of neurons per 3D convolutional layer and the number of fully-connected neurons refers to the number of neurons per fully-connected layer. Even though the architectures of the models were randomly generated through the monitored grid search, only architectures with a descending order of the number of neurons per both convolution layers and fully connected layers were considered valid candidate models. In other words, given a layer l with a number of neurons equal to n_l, the number of neurons n_{l+1} in the subsequent layer $l + 1$ had to be less than or equal to the number of neurons in layer l, i.e., $n_{l+1} \leq n_l$. The choice of decreasing the number of neurons throughout the layers created lighter models with a relatively small and condensed number of parameters. Every 3D convolutional layer was followed by a 3D max pooling layer except for the last 3D convolutional layer.

Table 1. Overall architectures of the 20 three-dimensional convolutional neural network models created from the monitored grid search for the detection of fusarium head blight-disease symptoms in wheat.

Model	Number of convolutional neurons	Number of fully-connected neurons
1	16,8,8	128,64,8,8
2	64,64,64,32,8	128,32,8
3	32,32,8,8	128,64,32
4	64,64,16	128,128,32
5	32,16,16,8	32,16
6	64,64,64,16	16
7	64,16,16,16	32,64,16
8	32,32,32,32,16	128,64,32,16
9	32,32,32,16,16	64,32,16,8
10	32,32,32,8,8	64,32,16
11	32,32,32,32,16	128,64
12	16,8,8,32,64	32
13	64,64,8,8,8	32,16
14	64,32,8	128,32,16,8
15	64,64,32	128,16,8
16	64,16,8,8	16,8
17	32,32,32,16	128
18	32,32	8,8,8
19	32,32,16,16,8	32
20	64,32,32	128

Note: "Number of convolutional neurons" is the number of neurons per convolutional layer, and "Number of fully-connected neurons" is the number of neurons per fully-connected layer.

Table 2. Overall architectures of the five three-dimensional convolutional neural network models generated by the monitored grid search for the estimation of the total number of spikelets.

Model	Number of convolutional neurons	Number of fully-connected neurons
1	32,16	128,64,8
2	32,32,8	128,16
3	32,16,16,16	64,8
4	32,16,8,8,8	32,32,16
5	32,32,32,32	128

Note: "Number of convolutional neurons" is the number of neurons per convolutional layer, and "Number of fully-connected neurons" is the number of neurons per fully-connected layer.

Table 3. Overall architectures of the top three three-dimensional convolutional neural network models generated by the monitored grid search for the estimation of the number of infected spikelets.

Model	Number of convolutional neurons	Number of fully-connected neurons	Optimizer
1	32,32,32,16	32,8	Adam
2	32,32,32,16	32,8	RMSprop
3	64,32,32,32	32	Adam

Note: "Number of convolutional neurons" is the number of neurons per convolutional layer, and "Number of fully-connected neurons" is the number of neurons per fully-connected layer.

Table 4. Overall architectures of the top three three-dimensional convolutional neural network models generated by the monitored grid search for the estimation of the fusarium head blight severity of infected wheat.

Model	Number of convolutional neurons	Number of fully-connected neurons
1	32,32,32,16	64,32,8
2	32,32,32,32	64,32,8
3	32,32,32,32	32

Note: "Number of convolutional neurons" is the number of neurons per convolutional layer, and "Number of fully-connected neurons" is the number of neurons per fully-connected layer.

For DFHB, the activation function in all the layers was the rectified linear unit (ReLU) (Nair and Hinton 2010), except for the output layer, where the activation was a sigmoid function. By default, the last fully-connected layer had one neuron since the networks were solving a detection problem. For ETNS, the only difference was the use of a ReLU activation function in all the networks' output layers. For ETNIS and FHBSA, every 3D convolutional layer is followed by a 3D max pooling layer except for the last 3D convolutional layer, and the last fully-connected layer has one neuron. For ETNIS, as shown by Table 3, Models 1 and 2 have identical 3D CNN architectures, with the optimizer being the only distinction, such that Model 1 used an Adam optimizer whereas Model 2 used a root mean square propagation (RMSProp) optimizer (Dauphin et al. 2015).

For DFHB, the top three 3D CNN models with the highest average CV accuracy on the 3DWP_RGB dataset were Models 8, 10, and 11. However, Models 8, 9, and 11 and Models 3, 5, and 9 were the top three models that achieved the highest average CV accuracy on the 3DWP_RGB_NIR dataset and the 3DWP_NIR dataset, respectively. With respect to ETNS, Model 5 was the best-performing model, since it achieved the best average CV MAE. For ETNIS, Model 1 achieved the best average MAE among the top three best-performing models. Its architecture consisted of four blocks, each consisting of a 3D convolutional layer and a 3D max pooling layer, where the kernel size of each convolutional layer was equal to (3 × 3 × 3). Following the convolutional layers was a flattening layer, followed by three densely connected layers where the number of neurons per dense layer was equal to 32, 8, and 1, respectively. The activation function in all the layers was a ReLU function. For FHBSA, Model 1 achieved the best average MAE among the top three best-performing models. Its architecture consisted of four blocks, each consisting of a 3D convolutional layer and a 3D max pooling layer, where the number of neurons in each convolutional layer was 32 except for the last layer, where the number of neurons was equal to 16, and the kernel size in all the convolutional layers was equal to (3 × 3 × 3). Following the convolutional layers was a flattening layer, followed by four densely connected layers where the number of neurons per dense layer was equal to 64, 32, 8, and 1, respectively. The activation function in all the layers was the ReLU function except for the output layer, where it used the sigmoid activation.

2.3.6. Model training

For DFHB model training, the RMSProp optimization algorithm was used to update each network's parameters, with a learning rate equal to 5E−4, and binary cross entropy (Mannor et al. 2005) was used as the loss function. Each model was trained for 100 epochs and a batch size equal to 5 was used. Even though a common practice is to set the batch size as a power of 2, we trained our models in a memory-bound environment that only allows us to choose a maximum batch size of 5. The closest power of 2 is 4. Moreover, when we trained our model with a batch size of 4, it led to a noisy training, which made the model fail to converge. As a result, we chose the maximum batch size possible, which made the training more stable and enabled us to achieve the results reported in this paper. The choice of the batch size for each model was set based on the maximum batch size possible, which was not necessarily a power of 2. Moreover, the training duration was roughly 30 min for all models in this work. To implement the 3D CNN models, we used the Python programming language and its open-source neural network library Keras (Chollet et al. 2015). We conducted the experiments using an NVIDIA Tesla P100 GPU with 12 GB of GPU memory.

For ETNS, the model architectures that achieved the best average CV MAE were trained on the full training set and tested on the test set. Table 5 shows the training parameters (depth, optimizer, regularizer, epochs, and batch size) corresponding to the best-performing models. Moreover, the optimizer, regularizer, epochs, and batch size also represent the training parameters for all the models. Starting with the 3D CNN, a batch size equal to 24 was used. The Adam optimization algorithm was used to update the network parameters with a learning rate equal to 1E−3, and MSE was used as the loss function. The 3D CNN was trained for 200 epochs.

For training each of the hundred ETNIS and FHBSA models, a batch size equal to 4 was used because there was not enough memory on the GPU to store a bigger batch. The RMSProp and the Adam optimization algorithms were used to update the network's parameters. For ETNIS, learning rates 1E−4, 5E−4, and 1E−3 were used during training, while a learning rate equal to 5E−4 was used for FHBSA. In both cases, MSE was used as a loss function, and each model was trained for 100 epochs.

2.3.7. Predefined model adaptation

Two approaches were followed to create regression models for the estimation of the total number of spikelets. In the first approach, 3D CNN networks were created from scratch through a monitored grid search using the approach reported in the previous section (this approach is labelled ETNS model development). In the second approach, three well-known CNN architectures were adapted for use with 3D data to solve the regression problem. These three networks were deep residual learning (ResNet) (He et al. 2016) in two variations (ResNet v1 and ResNet v2 (Hamila 2022)) and densely connected convolutional networks (DenseNet) (Huang et al. 2017; Hamila 2022). Taking this approach, 3D ResNet v1, 3D ResNet v2, and 3D DenseNet models were created. 3D ResNet v1 and 3D ResNet v2 were created by transforming every 2D convolutional layer and 2D average pooling layer into a 3D convolutional layer and a 3D average pooling layer, respectively. Moreover, the activation function of all the output layers was changed from a sigmoid function to a ReLU function. Similarly, a 3D DenseNet was created by changing every 2D convolutional layer and 2D average pooling layer in DenseNet into a 3D convolutional layer and a 3D average pooling layer, respectively. Furthermore, the activation function of the output layer was changed from a sigmoid function to a ReLU function. In total, three 3D ResNet v1, three 3D ResNet v2, and two 3D DenseNet were created.

Next, we selected the best version of each of the above predefined adapted models as follows: Beginning with 3D ResNet v1, each of the three networks consisted of one, two, or three residual blocks, respectively, with depths equal to 8, 14, and 20 layers (Hamila 2022). The best-performing model in the 5-fold CV for 3D RestNet v1 models was an architecture with a depth equal to 20 layers. Similarly, the next models investigated were two 3D ResNet v2 models, such that each network consists of one and two residual blocks, respectively, and their depths are equal to 11 and 20 layers (Hamila 2022). Here, the best-performing model in the 5-fold CV was an architecture with a depth equal to 11 layers. Finally, two 3D DenseNet networks were created, each having a four-layer dense block and a five-layer dense block with depth values equal to 23 and 29 layers, respectively. Per each model, each dense layer was preceded by a bottleneck layer, and each dense or bottleneck layer was followed by a dropout layer with a dropout rate equal to 0.2 (Hamila 2022). The best-performing 3D DenseNet model in the 5-fold CV was one with a depth equal to 23 layers.

To train both 3D ResNet v1 and v2 models, an Adam optimizer was employed with a learning rate equal to 1E−3. Both models employed L2 regularizer with a regularization factor equal to 1E−4 and MSE as the loss function. 3D ResNet v1 and 3D ResNet v2 used a batch size equal to 12 and 6, respectively. Both models were trained for 200 epochs. Finally, to train 3D DenseNet, an Adam optimizer was employed with a learning rate equal to 1E−3 and a dropout regularizer was used. The model was trained for 200 epochs with a batch size equal to 4.

2.4. FHB Disease assessment and statistical analysis

FHB visual assessment was carried out at 7, 14, and 21 DPI. However, FHB severity data collected by visual observation at 14 DPI for 19 *Fg* isolates were analyzed using SAS Studio software version 3.8 (SAS Institute Inc., Cary, NC). A generalized linear mixed model with a beta distribution function was fitted to the data using PROC GLIMMIX with the LOGIT link function and BETA distribution (SAS, 2014). The isolates were treated as a fixed factor and replicated as a random fac-

Table 5. The training parameters of the best-performing architecture per model in the estimation of the total number of spikelets.

Model	Depth	Optimizer	Regularizer	Epochs	Batch size
3D CNN	10	Adam	None	200	24
ResNet v1	20	Adam	l2	200	12
DenseNet	23	Adam	Dropout	200	4
ResNet v2	11	Adam	l2	200	6

Note: CNN, convolutional neural network.

tor. When a factor effect was significant, as indicated by a significant F test ($p \leq 0.05$), differences between the respective means were determined using the least significant difference test ($p \leq 0.05$). Regression analysis was used to determine the relationship between the 3D CNN model for severity estimation and severity collected by visual observation from 7 to 21 DPI.

3. Results

3.1. Point cloud to 3D image conversion

To visualize a few 3D images produced by the CUDA conversion model using various resolution factors, 2D projections were performed on 3D images representing a wheat plant. Figure 4 shows three 2D projections of a wheat plant obtained with different resolution factors R. Figures 4A–4C depict images obtained from R values equal to 2, 1, and 0.5, respectively. The resolution of the images differs depending on the value of R, such that in Fig. 4C where $R = 0.5$, the image has a low resolution due to the diminishing of the real dimensions by half during the conversion, whereas in Fig. 4B where $R = 1$, a higher resolution with more details and sharp edges can be observed due to the conservation of the real dimensions during the conversion. Figure 4A, where $R = 2$, shows a slightly better quality than Fig. 4B, such that its contours and details are more distinct.

3.2. Detection of fusarium head blight

Table 6 shows the detection performance metrics of the top three 3D CNN models over the three versions of datasets. Models 8, 10, and 11 achieved the highest average CV values among the batch of 20 models on the 3DWP_RGB dataset (characterized by RGB 3D images) by achieving 88.4%, 87.4%, and 86.8% average CV accuracy, respectively. These three models were retrained and evaluated over the test set, and achieved 100%, 91.3%, and 91.3% test accuracy, respectively. Models 11, 8, and 9 are the top three models among the 20 models that attained the highest average CV accuracies on the 3DWP_RGB_NIR dataset (characterized by RGB + NIR 3D images) and achieved 87.4%, 86.8%, and 85.8% average CV accuracy, respectively. These three models were retrained and tested on the dataset's test set, and despite Model 11 having the highest mean CV accuracy, it did not beat Model 8 in test accuracy. In fact, Model 8 achieved 95.7% test accuracy, followed by Models 11 and 9 that achieved 91.3% and 87.0% accuracy, respectively. Finally, Models 3, 5, and 9 achieved the highest average CV accuracies on the 3DWP_NIR dataset (characterized by NIR 3D images) by attaining 84.2%, 83.7%, and also 83.7% average CV accuracy, respectively. Despite the fact that Model 9 achieved the lowest mean CV accuracy among the top three models, it obtained the highest test accuracy of 87.0%, followed by Models 5 and 3 that achieved 82.6% and 78.3%, respectively.

3.3. Estimation of the total number of spikelets

Table 6 shows the results corresponding to the best-performing models (3D CNN, 3D ResNet v1, 3D ResNet v2, and 3D DenseNet) in the regression problem on Dataset I of wheat heads. The table shows the performance metrics of the models, which are the average CV MAE, the test MAE, and the average prediction time per sample in milliseconds. Both the 3D CNN and ResNet v2 achieved the best test MAE of 1.13. However, the 3D CNN outperformed the 3D ResNet v2 in the prediction time per sample with 14 ms versus 112 ms for the 3D RestNet v2. Moreover, even though 3D ResNet v1 obtained the best average CV MAE of 0.91, it failed to obtain it on the test set with an MAE of 1.23. However, 3D ResNet v1 produced the second-best prediction time of 62 ms per sample. Although 3D DenseNet was ranked last in terms of average CV MAE in the group of models by obtaining a 1.28 average CV MAE, it succeeded in achieving a 1.19 MAE on the test set, which is ranked third. 3D DenseNet also achieved an average prediction time per sample of 140 ms.

3.4. Estimation of the total number of infected spikelets

Table 6 shows the results corresponding to the best-performing 3D CNN models in the regression application, corresponding to the ETNIS on the Dataset II. Among the hundred models that were tested, only three achieved the lowest MAE, which are depicted as Models 1, 2, and 3 in the table. Model 1 achieved the best result of 1.56 MAE, meaning that the predicted total number of infected spikelets in a wheat head is, on average, equal to the true label with a tolerance of 1.56. Models 2 and 3 achieved the second- and third-lowest MAEs among the hundred models, which are equal to 1.57 and 1.63, respectively.

3.5. Fusarium head blight severity assessment

The results for the top-performing 3D CNN models in the regression application, which correspond to the estimation

Fig. 4. Two-dimensional projections of a three-dimensional image converted with different resolution factors *R* using the CUDA kernel.

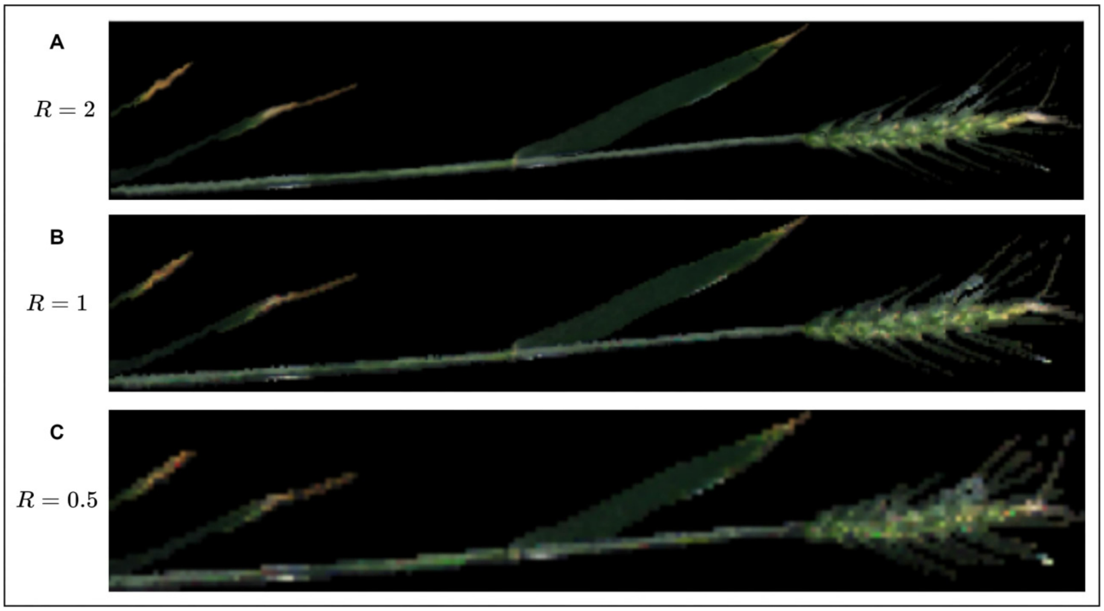


Table 6. Evaluation metrics of the top three-dimensional convolutional neural network (CNN) models per dataset and by application.

Application	Dataset	Model	Avg CV acc	Test acc %	Avg CV MAE	Test MAE	Inference time
FHB detection	3DWP_RGB	Model 8	88.4	100	—	—	—
		Model 10	87.4	91.3	—	—	—
		Model 11	86.8	91.3	—	—	—
	3DWP_RGB_NIR	Model 11	87.4	91.3	—	—	—
		Model 8	86.8	95.7	—	—	—
		Model 9	85.8	87.0	—	—	—
	3DWP_NIR	Model 3	84.2	78.3	—	—	—
		Model 5	83.7	82.6	—	—	—
		Model 9	83.7	87.0	—	—	—
Total No. of spikelets estimation	Dataset I (heads)	3D CNN	—	—	1.26	1.13	14
		ResNet v1	—	—	0.91	1.23	62
		DenseNet	—	—	1.28	1.19	140
		ResNet v2	—	—	1.05	1.13	l 42
No. of infected spikelets estimation	Dataset II	Model 1	—	—	2.06	1.56	—
		Model 2	—	—	2.09	1.57	—
		Model 3	—	—	3.05	1.63	—
Severity estimation	Dataset II	Model 1	—	—	12.4	8.6	—
		Model 2	—	—	12.6	8.8	—
		Model 3	—	—	12.9	9.0	—

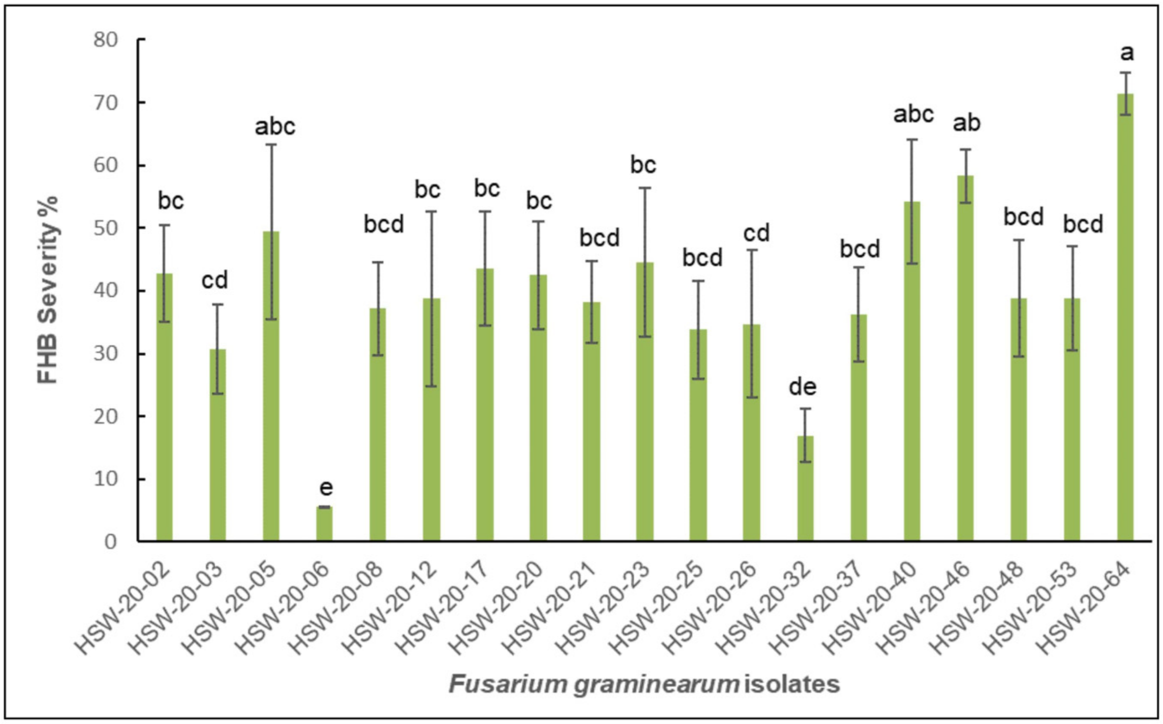
Note: Average cross-validation (CV) accuracy percentage (AVG CV acc %), Test accuracy percentage (Test acc %), and Average CV MAE (Avg CV MAE). FHB, fusarium head blight; RGB, red, green, blue; NIR, near-infrared; MAE, mean absolute error.

of FHB severity on Dataset II, are shown in Table 6. Only three of the 100 models that were tested, identified in the table as Models 1, 2, and 3, had the lowest MAE. Model 1 achieved the best result of 8.6 MAE, meaning that the predicted FHB severity of a wheat plant is, on average, equal to the true label with a tolerance of 8.6%. The FHB severity value varies from 0% (i.e., all the spikelets are healthy) to 100% (i.e., all the spikelets are infected). Models 2 and 3 achieved the second- and third-lowest MAEs among the hundred models, which are equal to 8.8 and 9.0, respectively.

3.6. Visual FHB severity assessment versus automated assessment via 3D CNN

We performed visual assessments of the *Fg* infection at 7, 14, and 21 DPI using a set of 19 different *Fg* showing that

Fig. 5. Visual fusarium head blight (FHB) disease assessment of the CWRS wheat cultivar 5602HR inoculated with *Fg* isolates at 14 days post inoculation. Means followed by a common letter are not statistically different at the 0.05% level of significance according to Fisher's unprotected LSD. HSW, Henriquez Spring Wheat.



all strains are pathogenic and a wide range of aggressiveness levels was observed. At 14 DPI, there were significant differences among the *Fg* isolates inoculated into the wheat cultivar 5602HR (Fig. 5). The FHB severity mean ranges from 5.6% to 71.3%. Randomly selected wheat heads, both inoculated with the 19 different *Fg* isolates and not inoculated, were used for linear regression. There was a significant relationship ($R^2 = 0.94$, $P = 0.0001$) between the visual disease assessment and the data obtained with the 3D CNN model (Fig. 6).

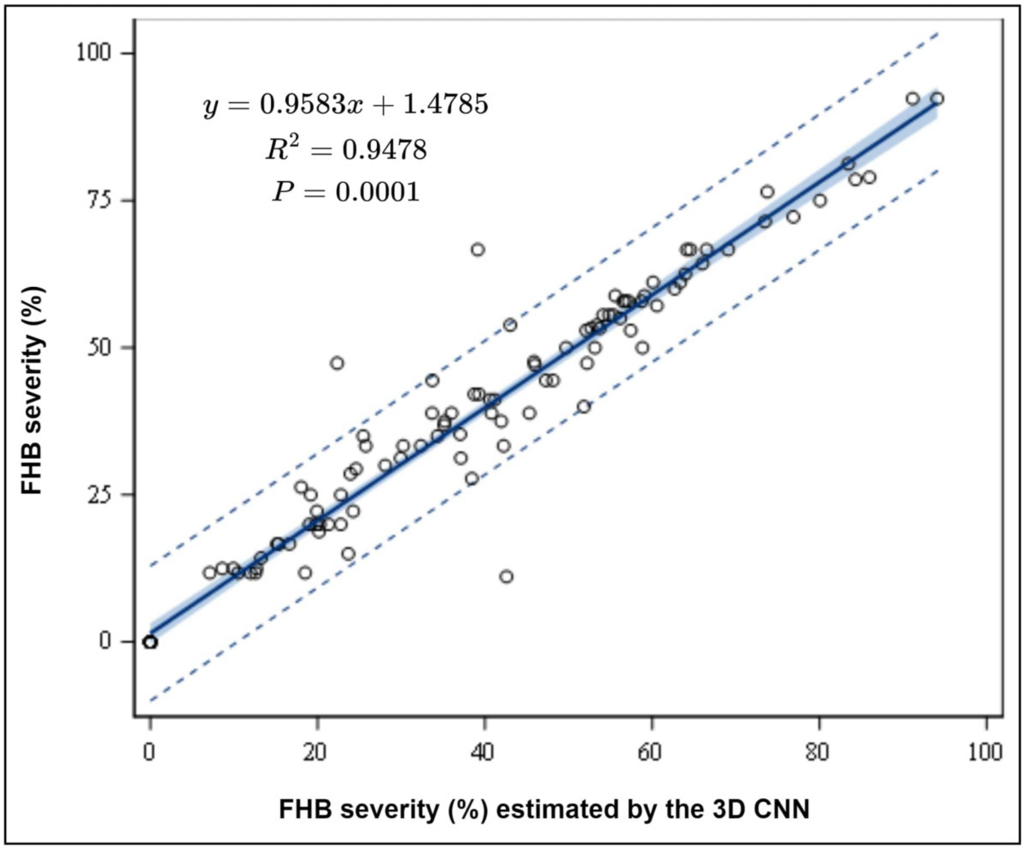
4. Discussion

4.1. Detection of Fusarium head blight

To date, the highest accuracy achieved in the DFHB was 98% (Liu et al. 2020) and was attained using hyperspectral 2D imagery and a pixel-wise classification. Our method for wheat FHB detection surpassed state-of-the-art methods in terms of accuracy using 3D multispectral PCs, and our best model achieved 100% accuracy. Krishnaswamy Rangarajan et al. (2022) also achieved 100% accuracy in the detection of wheat FHB. However, they used hyperspectral data in the visible range (400–750 nm), which they converted to image data to train detection models. Moreover, we explored the impact of multispectral information on the detection performance of the models. Our experiments suggested that the RGB colour model is superior to NIR of the Phenospex (i.e., 720 nm) and RGB and NIR combined. In fact, adding NIR information to RGB reduced the accuracy of the wheat FHB detection from 100% to 95.7%, which is a peculiar finding since, in general, adding more information to the data has the tendency to enrich it and give more information that should positively impact the performance of the CNN. However, in our case, NIR information (i.e., 720 nm) is observed to have a negative influence on the learning performance of the classifiers from 3D images of wheat. Using data consisting of only NIR information resulted in the lowest accuracy of 87.0% on the task of FHB detection in 3D images of wheat. These findings may be explained by the fact that correlations between FHB severity and spectral indices linked to water, nitrogen, and cellulose occur around spectral bands 1150 nm and from 1400 to 1550 nm (Vincke et al. 2023). As reported in Section 2.2.3, the spectral range of the PlantEye scanner is ∼450–750 nm. As a result, the important non-visible light information occurs outside of the spectral range of our equipment, and so the model must use the differences in visual light information to make decisions on the presence or absence of FHB.

A crucial phenotypic feature that can be used to evaluate the severity of FHB is the number of spikelets on a wheat head. However, developing an efficient phenotyping tool to automate the estimation of this number is challenging. Alkhudaydi and De La lglesia (2022) developed several models for counting wheat spikelets from infield images and concluded that the task is very difficult given the high complexity of the image acquisition context, such as varying background, severe occlusion, and density. While Liu et al. (2017) used images of grains spread on a solid colour board to develop an application for fast rice and wheat gain counting.

Fig. 6. Linear regression analysis between visual fusarium head blight (FHB) disease assessment (FHB severity %) and three-dimensional convolutional neural networks (CNNs). Randomly selected inoculated and non-inoculated wheat heads from 4 to 21 days post-inoculation were included in the analysis ($n = 112$). The black solid line represents the fit line, the blue shaded area represents the 95% confidence interval, and the dotted lines represent the prediction interval.



Their method achieved an error ratio for counting of less than 2%. Another study (Fernandez-Gallego et al. 2018) attempted to estimate wheat ear density under field conditions by using images of wheat plots at different growth stages. Their algorithm has a success rate between the predicted counting and the manual counting of 90%. In our model, we used 3D data and developed a 3D CNN model to estimate the number of spikelets on a wheat spike using PC data (i.e., the spikelets were not separated from the spike). The error value obtained by our 3D CNN regression model is 1.13, meaning that the difference between the predicted label and the true label, which was estimated manually, is on average equal to 1.13. Even though the MAE is not negligible, it is sufficient for automated assessment, and it is still much better than the rough estimation that is manually performed by humans.

The second crucial phenotypic feature that is essential to assessing FHB severity is estimating the number of infected spikelets on a wheat head. However, most of the studies developed methods to directly estimate the percentage of FHB severity. For example, Delwiche et al. (2019) used hyperspectral images containing individual kernels to estimate the percentage of fusarium-damaged kernels. Also, Gao et al. (2022*b*) used the automatic tandem dual NlendMask deep learning framework to segment wheat spikes and diseased areas for rapid disease severity detection from images of wheat spikes. Their method achieved 85% precision. Similarly, Gao et al. (2022*a*) attempted to improve the accuracy of FHB severity estimation in wheat using an augmented dataset and transfer learning to develop a CNN for the automatic recognition of FHB. Their model achieved 98% accuracy. Despite the fair results of these works, they either used 2D data, augmented data samples, or heavy preprocessing. Our model for the automatic ETNIS on a wheat head achieved an MAE of 1.56. The results are still within an acceptable error range, and the automated tool can still be considered an efficient and time-saving replacement for the manual and subjective calculation of the total number of infected spikelets. Furthermore, our results obtained from the 3D CNNs for the direct and automated estimation of the FHB severity in wheat are very promising. Our best-performing model achieved an 8.6 MAE on Dataset II. These results can serve as an efficient replacement for manual estimations.

The linear regression results of $R^2 = 0.94$ and $P = 0.0001$ show the important correlation between the FHB severity determined by 3D CNN and the actual FHB severity determined by visual observation. This implies that, even at very early stages of the infection, automated assessment of disease severity in wheat is a successful method for identifying the severity of FHB in wheat. Phenotyping hundreds of wheat lines for FHB resistance in the growth cabinet, greenhouse, or field is often a burdensome, time-consuming, and expensive process. New techniques like that proposed in this research could overcome these challenges. Moreover, this technology can be implemented and applied in different areas that focus on the management of FHB, such as plant breeding programs, precision crop protection, or the evaluation of fungicidal compounds. For future work, we intend to enlarge our dataset by scanning more wheat samples and use it to improve the performance of our detection and estimation models. Moreover, we intend to explore the unobvious symptoms of wheat FHB by acquiring more data samples that have non-visible FHB symptoms.

Finally, a discussion regarding the complexities of using deep learning models in comparison to human agents is helpful in further motivating this work. FHB visual assessment by human agents is a time-consuming, expensive, and error prone process. Manual assessment of FHB severity requires the training of personnel by an experienced specialist during several days. In fact, assessing FHB severity in the field requires several people to perform the rating as there is a very short time window to finish the disease rating when heads are still green. Using a rating scale from 0% to 100% tends to produce inconsistent values due to human subjectivity, tendency to overestimate/underestimate the severity, or bias (Shi et al. 2023). On the other hand, assessing FHB severity under controlled environmental conditions requires that individual raters identify and count the total number of infected spikelets and the total number of spikelets on the same wheat spike per inoculated plant. This counting process takes several weeks to complete. In contrast, creating ML models can take many months to achieve results sufficient for automated analysis. As reported here, this includes time-consuming data collection, labelling, and processing, as well as ML model development. However, once completed, ML models produce consistent results—with known misclassification rates or severity estimation errors—at a rate significantly faster than humans performing this same task.

Acknowledgements

We thank Otto Gruenke and Debbie Miranda for their technical support in maintaining plants and preparing inoculations. We thank Dr. Ezzat Ibrahim for establishing the Dr. Ezzat A. Ibrahim GPU Educational Lab at the University of Winnipeg, which provided the computing resources needed for this work.

An un-refereed preliminary version of this article was posted on the arXiv preprint server (Hamila et al. 2023*b*).

Article information

History dates

Received: 2 September 2023
Accepted: 30 January 2024
Accepted manuscript online: 26 February 2024
Version of record online: 10 April 2024

Copyright

Data availability

The original contributions presented in this study were produced using a public dataset created by the authors (Hamila et al. 2023*a*). The dataset is available at https://borealisdata.ca/dataset.xhtml?persistentId=doi:10.5683/SP3/QJWBEM.

Author information

Author ORCIDs

Oumaima Hamila https://orcid.org/0000-0003-4223-1598
Christopher J. Henry https://orcid.org/0000-0002-2624-1502
Oscar I. Molina https://orcid.org/0009-0005-4318-4721
Maria Antonia Henriquez https://orcid.org/0000-0001-9774-7925

Author contributions

Conceptualization: OH, CJH, MAH
Data curation: CJH
Formal analysis: OH, CJH, OIM, MAH
Funding acquisition: CJH, CPB
Investigation: OH, CJH, MAH
Methodology: OH
Project administration: CJH, CPB
Resources: CJH, CPB, MAH
Software: OH
Supervision: CJH, CPB, MAH
Validation: OH, CJH, MAH
Visualization: OH, OIM
Writing – original draft: OH, CJH
Writing – review & editing: OH, CJH, OIM, CPB, MAH

Competing interests

The authors declare there are no competing interests.

Funding information

This research was funded by Mitacs (Accelerate IT25876), Western Economic Diversification Canada (Project No. 15453), and Agriculture and Agri-Food Canada.

References

Alkhudaydi, T., and De Lalglesia, B. 2022. Counting spikelets from infield wheat crop images using fully convolutional networks. Neural Comput. Appl. **34**(20): 17539–17560. doi:10.1007/s00521-022-07392-1.

Anderson, J.A. 2007. Marker-assisted selection for fusarium head blight resistance in wheat. Int. J. Food Microbiol. **119**(1–2): 51–53.

Anderson, J.A., Stack, R., Liu, S., Waldron, B., Fjeld, A., Coyne, C., et al. 2001. DNA markers for fusarium head blight resistance qtls in two wheat populations. Theor. Appl. Genet. **102**: 1164–1168.

Barbedo, J.G., Tibola, C.S., and Fernandes, J.M. 2015. Detecting fusarium head blight in wheat kernels using hyperspectral imaging. Biosyst. Eng. **131**: 65–76. doi:10.1016/j.biosystemseng.2015.01.003.

Buerstmayr, M., and Buerstmayr, H. 2015. Comparative mapping of quantitative trait loci for fusarium head blight resistance and anther retention in the winter wheat population capo × arina. Theor. Appl. Genet. **128**: 1519–1530.

Buerstmayr, M., Steiner, B., and Buerstmayr, H. 2020. Breeding for fusarium head blight resistance in wheat–progress and challenges. Plant Breed. **139**(3): 429–454.

Charles, R., Su, H., Kaichun, M., and Guibas, L.J. 2017. Pointnet: deep learning on point sets for 3D classification and segmentation. *In* 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society, Los Alamitos, CA, USA. pp. 77–85. doi:10.1109/CVPR.2017.16.

Chin, T., Pleskach, K., Tittlemier, S.A., Henriquez, M.A., Bamforth, J., Withana Gamage, N., et al. 2023. A status update on fusarium head blight on western Canadian wheat. Can. J. Plant Pathol. 1–13.

Chollet, F. 2015. Keras. Available from https://keras.io [accessed 26 March 2024].

Dauphin, Y.N., Vries, H.D., and Bengio, Y. 2015. Equilibrated adaptive learning rates for non-convex optimization. *In* Proceedings of the 28th International Conference on Neural Information Processing Systems. Vol. 1. MIT Press, Cambridge, MA, USA, NIPS'15. pp. 1504–1512.

Delwiche, S., Rodriguez, I.T., Rausch, S., and Graybosch, R. 2019. Estimating percentages of fusarium-damaged kernels in hard wheat by near-infrared hyperspectral imaging. J. Cereal Sci. **87**: 18–24. doi:10.1016/j.jcs.2019.02.008.

Dhariwal, R., Henriquez, M.A., Hiebert, C., McCartney, C.A., and Randhawa, H.S. 2020. Mapping of major fusarium head blight resistance from canadian wheat cv. AAC Tenacious. Int. J. Mol. Sci. **21**(12). doi:10.3390/ijms21124497.

Ferentinos, K.P. 2018. Deep learning models for plant disease detection and diagnosis. Comput. Electron. Agr. **145**: 311–318. doi:10.1016/j.compag.2018.01.009.

Fernandez-Gallego, J.A., Kefauver, S.C., Gutiérrez, N.A., Nieto-Taladriz, M.T., and Araus, J.L. 2018. Wheat ear counting in-field conditions: high throughput and low-cost approach using rgb images. Plant Methods, **14**.

Ferrigo, D., Raiola, A., and Causin, R. 2016. Fusarium toxins in cereals: occurrence, legislation, factors promoting the appearance and their management. Molecules, **21**: 627. doi:10.3390/molecules21050627.

Gao, C., Gong, Z., Ji, X., Dang, M., He, Q., Sun, H., and Guo, W. 2022a. Estimation of fusarium head blight severity based on transfer learning. Agronomy, **12**(8).

Gao, Y., Wang, H., Li, M., and Su, W.H. 2022b. Automatic tandem dual blendmask networks for severity assessment of wheat fusarium head blight. Agriculture, **12**(9).

Ghimire, B., Sapkota, S., Bahri, B.A., Martinez-Espinoza, A.D., Buck, J.W., and Mergoum, M. 2020. Fusarium head blight and rust diseases in soft red winter wheat in the southeast united states: state of the art, challenges and future perspective for breeding. Front. Plant Sci. **11**. doi:10.3389/fpls.2020.01080.

Gu, C., Wang, D., Zhang, H., Zhang, J., Zhang, D., and Liang, D. 2021. Fusion of deep convolution and shallow features to recognize the severity of wheat fusarium head blight. Front. Plant Sci. **11**. doi:10.3389/fpls.2020.599886.

Hamila, O. 2022. 3D convolutional neural networks for solving complex digital agriculture and medical imaging problems. MSc thesis, University of Winnipeg. doi:10.36939/ir.202206021141.

Hamila, O., Henry, C.J., Molina, O.I., Bidinosti, C.P., and Henriquez, M.A. 2023a. UW-MRDC 3D WHEAT. Borealisdoi:10.5683/SP3/QJWBEM.

Hamila, O., Henry, C.J., Molina, O.I., Bidinosti, C.P., and Henriquez, M.A. 2023b. Fusarium head blight detection, spikelet estimation, and severity assessment in wheat using 3D convolutional neural networks. arXivdoi:10.48550/arXiv.2303.05634.

Harfouche, A.L., Nakhle, F., Harfouche, A.H., Sardella, O.G., Dart, E., and Jacobson, D. 2023. A primer on artificial intelligence in plant digital phenomics: embarking on the data to insights journey. Trends Plant Sci. **28**(2): 154–184. doi:10.1016/j.tplants.2022.08.021.

He, K., Zhang, X., Ren, S., and Sun, J. 2016. Deep residual learning for image recognition. *In* 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778. doi:10.1109/CVPR.2016.90.

Huang, G., Liu, Z., Maaten, L.V.D., and Weinberger, K.Q. 2017. Densely connected convolutional networks. *In* 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society, Los Alamitos, CA, USA. pp. 2261–2269. doi:10.1109/CVPR.2017.243.

Huang, L., Li, T., Ding, C., Zhao, J., Zhang, D., and Yang, G. 2020. Diagnosis of the severity of fusarium head blight of wheat ears on the basis of image and spectral feature fusion. Sensors, **20**(10).

Jordan, M.I., and Mitchell, T.M. 2015. Machine learning: trends, perspectives, and prospects. Science, **349**(6245): 255–260. doi:10.1126/science.aaa8415.

Khan, M.K., Pandey, A., Athar, T., Choudhary, S., Deval, R., Gezgin, S., et al. 2020. Fusarium head blight in wheat: contemporary status and molecular approaches. 3 Biotech, **10**: 1–17.

Klokov, R., and Lempitsky, V. 2017. Escape from cells: deep kd-networks for the recognition of 3D point cloud models. *In* Proceedings of the IEEE International Conference on Computer Vision. pp. 863–872.

Krishnaswamy Rangarajan, A., Louise Whetton, R., and Mounem Mouazen, A. 2022. Detection of fusarium head blight in wheat using hyperspectral data and deep learning. Expert Syst. Appl. **208**: 118240. doi:10.1016/j.eswa.2022.118240.

Li, Y., Bu, R., Sun, M., Wu, W., Di, X., and Chen, B. 2018. PointCNN: convolution on x-transformed points. Adv. Neural Inf. Process. Syst. **31**.

Liu, L., Dong, Y., Huang, W., Du, X., and Ma, H. 2020. Monitoring wheat fusarium head blight using unmanned aerial vehicle hyperspectral imagery. Remote Sens. **12**: 3811. doi:10.3390/rs12223811.

Liu, T., Chen, W., Wang, Y., Wu, W., Sun, C., Ding, J., and Guo, W. 2017. Rice and wheat grain counting method and software development based on android system. Comput. Electron. Agr. **141**: 302–309. doi:10.1016/j.compag.2017.08.011.

Lu, B., Dao, P.D., Liu, J., He, Y., and Shang, J. 2020. Recent advances of hyperspectral imaging technology and applications in agriculture. Remote Sens. **12**(16).

Lu, Y., and Young, S. 2020. A survey of public datasets for computer vision tasks in precision agriculture. Comput. Electron. Agric. **178**: 105760. doi:10.1016/j.compag.2020.105760.

Ma, Y., Soatto, S., Kosecka, J., and Sastry, S.S. 2003. An invitation to 3-D vision: from images to geometric models. Springer Verlag.

Mannor, S., Peleg, D., and Rubinstein, R. 2005. The cross entropy method for classification. *In* Proceedings of the 22nd International Conference on Machine Learning. pp. 561–568.

Mao, R., Wang, Z., Li, F., Zhou, J., Chen, Y., and Hu, X. 2023. Gseyoloxs: an improved lightweight network for identifying the severity of wheat fusarium head blight. Agronomy, **13**(1). doi:10.3390/agronomy13010242.

Nair, V., and Hinton, G.E. 2010. Rectified linear units improve restricted boltzmann machines. *In* Proceedings of the 27th International Conference on International Conference on Machine Learning. Omnipress, Madison, WI, USA. pp. 807–814.

Nilsen, K.T., Walkowiak, S., Kumar, S.V., Molina, Ó.I., Randhawa, H.S., Dhariwal, R., et al. 2020. Histology and RNA sequencing provide insights into fusarium head blight resistance in AAC Tenacious. Front. Plant Sci. **11**.

Nvidia Vingelmann, P., and Fitzek, F.H. 2020. Cuda, release: 10.2.89.

Qiu, R., Yang, C., Moghimi, A., Zhang, M., Steffenson, B.J., and Hirsch, C.D. 2019. Detection of fusarium head blight in wheat using a deep neural network and color imaging. Remote Sens. **11**(22). doi:10.3390/rs11222658.

Refaeilzadeh, P., Tang, L., and Liu, H. 2009. Cross-validation. *In* Encyclopedia of database systems. *Edited by* L. Liu and M.T. Özsu. Springer, US, Boston, MA. pp. 532–538. doi:10.1007/978-0-387-39940-9_565.

Sakuma, S., Golan, G., Guo, Z., Ogawa, T., Tagiri, A., Sugimoto, K., et al. 2019. Unleashing floret fertility in wheat through the mutation of a homeobox gene. Proc. Natl. Acad. Sci. USA **116**(11): 5182–5187. doi:10.1073/pnas.1815465116.

Sharp, N., 2015. hapPLY API. Available from https://github.com/nmwsharp/happly.

Shi, S., Wang, X., and Li, H. 2019. PointRCNN: 3D object proposal generation and detection from point cloud. *In* Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 770–779.

Shi, T., Liu, Y., Zheng, X., Hu, K., Huang, H., Liu, H., and Huang, H. 2023. Recent advances in plant disease severity assessment using convolutional neural networks. Sci. Rep. **13**(1): 2336.

Singh, R., Srivastava, S., and Mishra, R. 2020. AI and iot based monitoring system for increasing the yield in crop production. *In* 2020 International Conference on Electrical and Electronics Engineering (ICE3). pp. 301–305. doi:10.1109/ICE348803.2020.9122894.

Teke, M., Deveci, H.S., Haliloğlu, O., Gürbüz, S.Z., and Sakarya, U. 2013. A short survey of hyperspectral remote sensing applications in agriculture. *In* 2013 6th International Conference on Recent Advances in Space Technologies (RAST). pp. 171–176. doi:10.1109/RAST.2013.6581194.

Vázquez-Arellano, M., Griepentrog, H., Reiser, D., and Paraforos, D. 2016. 3-D imaging systems for agricultural applications—a review. Sensors, **16**(5): 618. doi:10.3390/s16050618.

Vincke, D., Eylenbosch, D., Jacquemin, G., Chandelier, A., Pierna, J.A.F., Stevens, F., et al. 2023. Near infrared hyperspectral imaging method to assess fusarium head blight infection on winter wheat ears. Microchem. J. **191**: 108812.

Zadoks, J.C., Chang, T.T., and Konzak, C.F. 1974. A decimal code for the growth stages of cereals. Weed Res. **14**(6): 415–421.

Zhang, D., Wang, D., Gu, C., Jin, N., Zhao, H., Chen, G., et al. 2019. Using neural network to identify the severity of wheat fusarium head blight in the field environment. Remote Sens. **11**(20). doi:10.3390/rs11202375.

Zhang, D.Y., Luo, H.S., Wang, D.Y., Zhou, X.G., Li, W.F., Gu, C.Y., et al. 2022. Assessment of the levels of damage caused by fusarium head blight in wheat using an improved yolov5 method. Comput. Electron. Agr. **198**: 107086. doi:10.1016/j.compag.2022.107086.

Appendix A. Data preprocessing with CUDA

The multispectral PCs generated from the PlantEye 3D scanner are stored in a polygon file format known as PLY, which is a file format designed specifically to save 3D models. A PLY file contains tuples of flat polygons in addition to tuples of colour information. Flat polygons and colour information are described by a tuple of (x, y, z) coordinate values varying between negative and positive floating-points and a tuple of (R, G, B, NIR) intensity values, where each value is stored as an integer varying between [0, 255]. Moreover, the tuple of point coordinates stored in a PLY file is unordered, such that each point is independent and unrelated to the remaining points within the file. The ensemble of points is useful to reconstruct 3D models in space by placing each coordinate in its specific spatial position. However, PLY representation does not support complex operations such as convolutions and matrix manipulations that require points within a data signal to be correlated and organized such that a meaningful change in space or time between points can be defined. As a result, in this work, a C++ program that converts PLY files into 3D images and that runs on GPUs was developed to overcome the limitations of using CNNs on PLY files. The CUDA (NVIDIA et al. 2020) parallel computing platform and programming model was employed to develop the conversion model.

A.1 Theory and implementation of point cloud to 3D image conversion

Our proposed solution for converting PLY files into 3D images was based on linear interpolation, such that every point coordinate in the tuple of points stored in a PLY file was converted through linear interpolation into a new voxel coordinate within the constructed 3D image. This interpolation was necessary because the coordinates stored in a PLY file can be either negative or positive floating points, while the coordinates required by 3D CNNs have to be positive integers for indexing. The conversion operations are repetitive and separable, meaning that they can be applied independently to all the point coordinates in the PLY file, which provided a perfect opportunity to exploit GPU parallelism. Thus, the proposed CUDA-based method applies the same linear operations simultaneously on all the points in a PLY file. The conversion equations defining the linear interpolation along the x-, y-, and z-axes are

$$\text{(A1)}\quad \begin{aligned} x_{\text{matrix}} &= \lceil a_x \; x_{\text{PC}} + b_x \rceil \\ y_{\text{matrix}} &= \lceil a_y \; y_{\text{PC}} + b_y \rceil \\ z_{\text{matrix}} &= \lceil a_z \; z_{\text{PC}} + b_z \rceil \end{aligned}$$

such that a_x, a_y, and a_z are, respectively, the function slope corresponding to the x-, y-, and z-axes, and b_x, b_y, and b_z are, respectively, their intercepts. $x_{\text{matrix}}, y_{\text{matrix}}$, and z_{matrix} are the positions of the point along the width, height, and depth of the output 3D image, corresponding, respectively, to the transformation of x, y, and z values of a point coordinate in the PLY file, noted, respectively, as $x_{\text{PC}}, y_{\text{PC}}$, and z_{PC}. Moreover, $\lceil x \rceil$ defines the ceiling function of a real number x that is defined as the smallest integer that is not smaller than x. The function's slopes and intercepts are calculated as

$$\text{(A2)}\quad \begin{aligned} a_x &= \frac{\lceil R\left(\max_{1\le i\le N}(x_i) - \min_{1\le i\le N}(x_i)\right)\rceil}{\max_{1\le i\le N}(x_i) - \min_{1\le i\le N}(x_i)}, \; b_x = -a_x \min_{1\le i\le N}(x_i) \\ a_y &= \frac{\lceil R\left(\max_{1\le i\le N}(y_i) - \min_{1\le i\le N}(y_i)\right)\rceil}{\max_{1\le i\le N}(y_i) - \min_{1\le i\le N}(y_i)}, \; b_y = -a_y \min_{1\le i\le N}(y_i) \\ a_z &= \frac{\lceil R\left(\max_{1\le i\le N}(z_i) - \min_{1\le i\le N}(z_i)\right)\rceil}{\max_{1\le i\le N}(z_i) - \min_{1\le i\le N}(z_i)}, \; b_z = -a_z \min_{1\le i\le N}(z_i) \end{aligned}$$

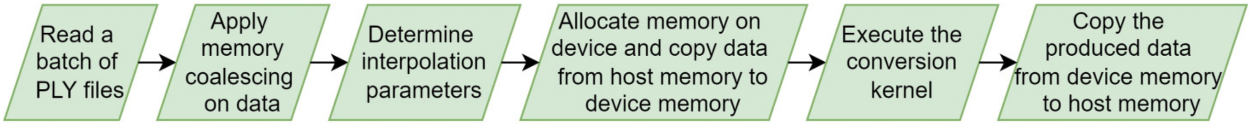
such that N is the total number of points in the tuple of point coordinates in the PLY file, (x_i, y_i, z_i) is the coordinate of the ith point in the tuple, and R is the resolution factor that serves to enlarge or reduce the resolution of the output 3D image. Finally, for the linear transformation, only the spatial coordinates (x, y, z) were used to estimate the new voxel coordinates $(x_{\text{matrix}}, y_{\text{matrix}}, z_{\text{matrix}})$, while their corresponding colour intensities (R, G, B, NIR) were reallocated in the new voxel coordinates within the 3D image. The dimensions of the output 3D image are

$$\text{(A3)}\quad \begin{aligned} \text{width} &= \left\lceil R\left(\max_{1\le i\le N}(x_i) - \min_{1\le i\le N}(x_i)\right)\right\rceil \\ \text{height} &= \left\lceil R\left(\max_{1\le i\le N}(y_i) - \min_{1\le i\le N}(y_i)\right)\right\rceil \\ \text{depth} &= \left\lceil R\left(\max_{1\le i\le N}(z_i) - \min_{1\le i\le N}(z_i)\right)\right\rceil \end{aligned}$$

such that width, height, and depth correspond to the range of values along the x-, y-, and z-axes, respectively.

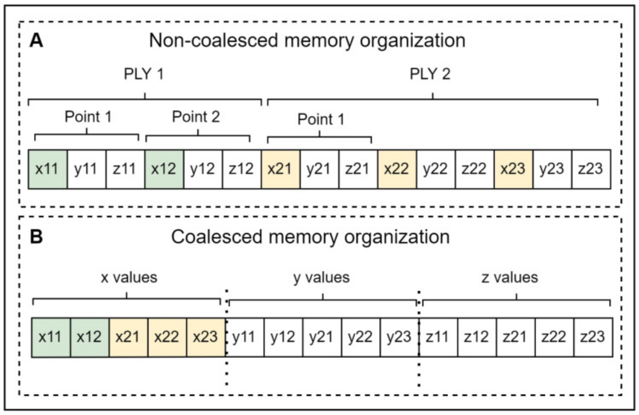
To implement eqs. A1–A3, a general C++ API called hapPLY (Sharp et al. 2015) was used to load PLY files. The API allows the reading and writing of the properties of a PLY file, such as the point coordinates and their corresponding colour

Fig. A1. Diagram of CUDA implementation steps to convert a batch of PLY files into a batch of 3D images.


Read a batch of PLY files
Apply memory coalescing on data
Determine interpolation parameters
Allocate memory on device and copy data from host memory to device memory
Execute the conversion kernel
Copy the produced data from device memory to host memory

intensities, and loads them as two separate tuples of real values. Figure A1 shows the implementation steps followed in the CUDA code to convert a batch of PLY files into a batch of 3D images. The code started by reading the properties of a batch of PLY files. Processing the data in batches allows for further optimization of parallel execution with CUDA, such that the code processes all data points of $n \times$ PCs simultaneously, rather than only the data points of a single PC. Next, all elements of the tuples of coordinates and tuples of colours of the $n \times$ PCs were rearranged in a manner that ensures memory coalescing (see Section A.2 for more details), which enables accessing consecutive memory locations within a single I/O operation. Following that, the maximum and minimum values of the coordinates needed to estimate the parameters of the interpolation functions and the dimensions of the output batch of 3D images were determined and used for calculations. Next, the memory space needed for the data that were used during the kernel execution was allocated on the device memory, and the data were copied from the host memory to the device memory. Then, the conversion kernel, which is the function executed on the GPUs, was launched to convert the batch of PLY files into their corresponding 3D images. Finally, the produced batch of 3D images was copied from the device memory to the host memory.

Fig. A2. Data organization in a memory array for both coalesced and non-coalesced patterns of data. (A) depicts the raw storage of data in a memory array, and (B) depicts the storage of the same data in memory in a coalesced manner.



A.2 Memory coalescing

With respect to the CUDA programming model, threads within a thread block are organized into warps, where a warp is a group of 32 consecutive threads assigned to execute the same set of operations. In practice, threads within a warp access sequential memory locations for read and write operations. This means that memory access operations can be a major bottleneck for GPU applications if the data accessed by sequential threads in a warp are not sequentially ordered in memory. Therefore, the solution is memory coalescing (NVIDIA et al. 2020), which is a technique used by CUDA where global memory accesses of threads in a warp are grouped together into one operation to minimize global memory bandwidth. In fact, each time a global memory location is accessed, a set of consecutive locations, including the requested location, are also accessed. Thus, in order to reduce the latency caused by data access operations, we made sure that the data used by consecutive threads in a warp are stored in consecutive memory locations.

The kernel that performs the data conversion operations was programmed to estimate each value within the tuple of point coordinates separately, which means that x_{matrix}, y_{matrix}, and z_{matrix} are all estimated independently of one another. Thus, threads within a block were designed such that each block of threads was programmed to load and operate on either the x_{PC}, y_{PC}, or z_{PC} values to calculate either the x_{matrix}, y_{matrix}, or z_{matrix} values, respectively. The kernel architecture was designed to take advantage of memory coalescing during data loading, so that one thread within a warp loads all consecutive x_{PC}, y_{PC}, or z_{PC} values from memory into the cache, allowing the remaining threads to load their corresponding data directly from the cache and execute their operations faster. Figure A2 shows the data organization in a memory array for both coalesced and non-coalesced patterns. The illustrated examples use only a few points per PC for the purpose of demonstration only. Figure A2A depicts the raw storage of data in a memory array in which point coordinates corresponding to a batch of PLY files are arranged in such a way that points corresponding to the first file are stored first, followed by points corresponding to the second file, and so on, and each point is stored by its (x, y, z) coordinates, where each memory slot contains one coordinate value. The first two point coordinates represent the first PLY file of the batch, while the following points in the array correspond to the second PLY file within the same batch. ($x11$, $y11$, $z11$) represents the first point of the first PLY file within the batch, followed by ($x12$, $y12$, $z12$), which represents the second point of the first PLY file. Once all the points corresponding to the first PC within the batch are stored, the points corresponding to the second PC are added to the same array. In the example, ($x21$, $y21$, $z21$) represents the first point of the second PLY file, and so on. This kind of arrangement is not suitable for an optimized CUDA kernel execution because the memory accesses will be inefficient.

Thus, point coordinates in memory were rearranged to ensure that threads access coalesced data locations during kernel execution. Figure A2B shows an array where all the x values representing all the points from the PLY files in a batch that were stored consecutively are placed in successive memory slots, followed by all the y values, and finally all the z values. Not only were the tuple of point coordinates rearranged to support data coalescing, but also the tuple of colours. Colour intensities were loaded such that each (R, G, B, NIR) tuple corresponding to the first point of the first PLY file was the first element of the memory array, followed by the second (R, G, B, NIR) tuple corresponding to the first PLY file and so on. Thus, the tuples of colours were rearranged so that all R values representing the points of the first PLY file within a batch were put first in the memory array, followed by all the R values of the second PLY file, and so on. Once the R values were stored, G, B, and NIR values were then stored consecutively in the memory array according to the same memory coalescing principle.

A.3 Conversion kernel

The conversion kernel function was implemented to perform point coordinate transformations from their original spatial placement within the PC to their new voxel positions within the 3D dimensions of a 3D image. Each thread was designed to calculate the linear interpolation of a single point, which means that each thread executed the linear interpolation functions defined in eq. A1 and related to the (x, y, z) values of a point's coordinates. Firstly, the number of threads allocated on the device memory was determined to be $\frac{1}{3}$ of the coordinates list, and those threads were each programmed to execute three linear interpolations related to their designated (x, y, z) coordinates in order to determine the new voxel coordinates within the output 3D image. Next, each thread loaded the (R, G, B) colour intensities and placed the tuple of colours in their corresponding voxel position within the output 3D image. In fact, the interpolation functions defined in eq. A1 convert floating-point coordinates into integer coordinates (with the ceiling operation) that define the voxel positions within the constructed 3D image. Moreover, in some cases, more than one real-valued point coordinate may get converted into the exact same voxel coordinate. In that case, the newer point would override the existing one, resulting in a reduction in the total number of points defined in the 3D image. Furthermore, the size of the constructed 3D image, as defined in eq. A3, ensured that the object defined in the PC was converted into a minimum bounding box, which was the generated 3D image. Moreover, the voxel values that remained empty after reassigning the colour tuples from their positions in the PC to their new voxel positions within the constructed 3D image, were set to zero. The conversion kernel described in this section produced three-channel 3D images with each voxel value consisting of a tuple of (R, G, B) colour intensities, while NIR intensity values were processed through a second kernel to produce one-channel 3D images.