Computer Vision-Driven Plant Inventory & Seedling Tray Viability Assessment

Omisha Sapra[1*], Yadunandan Ramanna[2*], Brian Keller[3*], Wyatt Young[4*],
David Sacharny[5], Ziad Al-Halah[6], Thomas C. Henderson[7]

***Abstract*— Nursery operations require daily assessment of sapling viability across hundreds of trays containing 50 - 200+ cells each. Manual inspection at this scale is labor-intensive and inconsistent, making early detection of germination failures difficult to sustain. We present an end-to-end computer vision pipeline for automated seedling tray viability assessment from mobile-captured images. The system accepts a raw overhead tray photograph captured on a mobile device and produces a tray-level viability estimate.**

The pipeline consists of four stages: YOLOv8 tray segmentation and perspective rectification achieving approximately 93% mask accuracy on a bootstrapped dataset of 730 labeled images; ResNet-18 grid type classification over 29 discrete tray configurations achieving 86.4% top-1 validation accuracy on a purpose-labeled 105-image dataset; grid inference recovering the exact tray configuration in 88.6% of labeled evaluation trays; and per-cell validity classification achieving 96.74% accuracy on 9,155 labeled cell images. End-to-end evaluation on a ground-truth-labeled subset of 105 trays yielded a tray-level viability mean absolute error of 6.01 percentage points with Pearson correlation $r = 0.942$, while analysis of the full 1,037-image production batch showed a median inference time of 1.65 s per image.

Tray-level viability is computed as the percentage of cells classified as occupied within the detected grid. The system is deployed as a modular pipeline with structured outputs, including per-cell predictions and annotated visualizations. A browser-based annotation tool with integrated perspective rectification was developed to efficiently generate the labeled datasets used across the pipeline.

I. INTRODUCTION

In large-scale commercial plant propagation, the ability to accurately forecast inventory is critical to operational success. Modern nurseries manage thousands of seedlings across hundreds of trays simultaneously. In this environment, the "Percent Loss" – the ratio of failed germinations to the total number of planted cells – is a primary driver for high-stakes business decisions. If germination rates drop below a specific threshold, nursery managers must rapidly reallocate greenhouse space or adjust sales expectations for peak seasons.

Traditional inventory management relies on manual visual estimation, where workers assess germination rates through rapid visual inspection of individual trays. This heuristic approach is inherently subjective and infeasible to perform at the single-cell level across hundreds of units, leading to significant variance between observers and poor scalability. While computer vision (CV) has made significant strides in agriculture – particularly in mature plant phenotyping and fruit counting [1], [2] – there is a notable lack of automated systems designed for the early germination stage. Most existing agricultural CV literature focuses on complex multi-class health indicators or individual plant measurements, which often introduce unnecessary computational overhead for a nursery's primary need: a fast, objective binary assessment of whether a cell is "Occupied"(germinated) or "Empty"(non-germinated).

To address these challenges, we present an automated pipeline for nursery viability assessment from mobile-captured images. Unlike prior agricultural vision systems that primarily target mature plant phenotyping and fruit counting [1], [2], our work focuses on the operational nursery problem of early-stage viability estimation. The system combines confidence-gated grid recovery, purpose-built annotation tooling and deployment-oriented structured outputs for real nursery workflows. It follows a hierarchical architecture in which a YOLOv8 model performs tray segmentation and enables perspective rectification, a ResNet-18 classifier predicts tray grid configuration, a separator-based method serves as a fallback when classifier confidence is low, and a second ResNet-18 model performs per-cell binary classification (empty vs. occupied) to produce a tray-level viability estimate. The primary contributions of this work are as follows:

1) **End-to-End Automated Pipeline:** A complete capture-to-index system that automates the transition from raw mobile imagery to a Viability Index (VI). The system removes the subjectivity of manual visual estimation, achieving a tray-level mean absolute viability error of 6.01 percentage points relative to manual ground truth while providing a scalable framework for nursery inventory management.
2) **Robust Spatial Rectification and Grid Segmentation**: A multi-stage preprocessing workflow – including YOLOv8 tray isolation, perspective warping, and a dual-method grid configuration system combining

[1]MS Student, Kahlert School of Computing, University of Utah, Salt Lake City, UT, USA omisha.s@utah.edu
[2]MS Student, Kahlert School of Computing, University of Utah, Salt Lake City, UT, USA yadunandan.ramanna@utah.edu
[3]BS/MS Student, Department of Biomedical Engineering, University of Utah, Salt Lake City, UT, USA brian.keller@utah.edu
[4]BS/MS Student, Department of Biomedical Engineering, University of Utah, Salt Lake City, UT, USA young.wyatt@utah.edu
[5]Adjunct Assistant Professor, Kahlert School of Computing, University of Utah, Salt Lake City, UT, USA sacharny@cs.utah.edu
[6]Assistant Professor, Kahlert School of Computing, University of Utah, Salt Lake City, UT, USA ziad.al-halah@utah.edu
[7]Professor, Kahlert School of Computing, University of Utah, Salt Lake City, UT, USA tch@cs.utah.edu

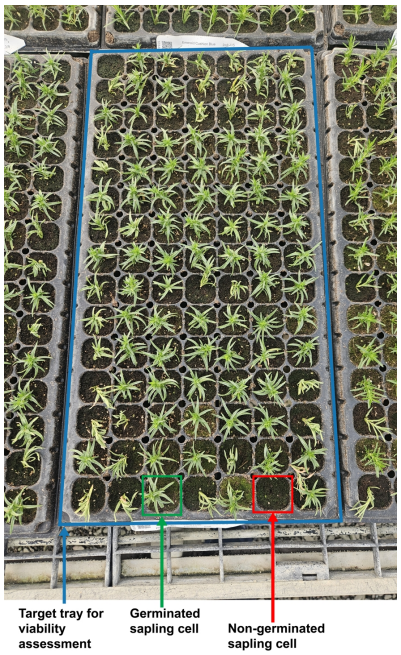

Fig. 1. Representative overhead capture of a commercial seedling tray. The blue boundary defines the segmented Region of Interest (ROI) for viability assessment. By performing binary classification at the cell level—Occupied (green) vs. Empty (red)—the system provides an objective, scalable alternative to subjective manual visual estimation.

a learned ResNet-18 classifier with a separator-based computer vision pipeline – that ensures consistent cell boundary placement regardless of camera angle, tray type, or foliage density.

3) **Empirical Optimization for Industrial Deployment**: Using lightweight ResNet-18 backbones and a confidence-gated grid cascade, the system achieves strong end-to-end accuracy while maintaining practical low-latency inference for mobile-assisted nursery workflows.
4) **Purpose-Built Annotation Infrastructure**: A browser-based annotation tool with integrated OpenCV.js perspective rectification that enables structured, grid-aligned labeling of tray images in a single workflow. The 105-image data set produced by this tool is the training data source for the grid type classifier and the cell-level label source for the viability classifier.

II. PREVIOUS WORK

A. *Object Detection and Segmentation*

Early deep learning approaches to agricultural detection relied on region-based methods such as R-CNN [3] and Faster R-CNN [4]. While precise, these two-stage detectors are computationally expensive for real-time nursery deployment. The introduction of the "You Only Look Once" (YOLO) framework [5] revolutionized the field by reformulating detection as a single regression problem, enabling real-time inference on mobile devices. Recent iterations, specifically YOLOv8 [6], have shown superior performance

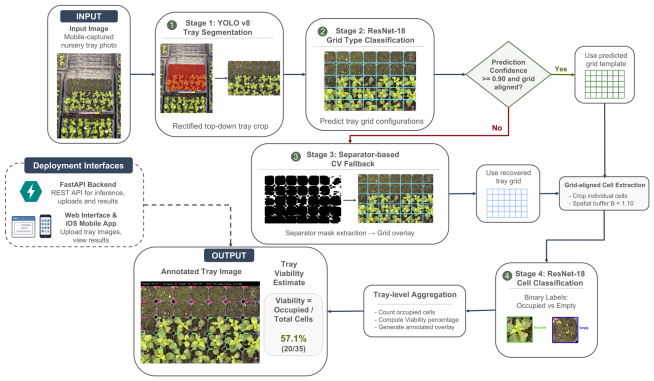

Fig. 2. System architecture of seedling tray viability assessment pipeline. The pipeline processes a mobile-captured tray image through tray segmentation and rectification, grid recovery, cell extraction and cell-level occupied/empty classification to produce the final tray-level viability estimate.

in dense scenes with small objects, making them well-suited for identifying the repetitive grid structures of nursery trays.

B. *Computer Vision in Horticulture and Phenotyping*

Automated plant phenotyping (species classification) has traditionally utilized deep learning to infer complex traits such as leaf count or biomass [7], [8]. However, these systems often struggle with the "annotation bottleneck"-the high cost of manual labeling required for diverse species [9]. Furthermore, most research in this domain focuses on plant-level metrics rather than inventory-level aggregation. Our work bridges this gap by focusing on viability as a binary operational metric, which is more robust for early-growth stages and directly supports decision-making in greenhouse management [10].

III. METHODS

A. *Dataset*

Images were collected across two sessions at a commercial nursery (Perennial Favorites, Layton, UT) using iPhone and Android cameras at approximately 5000×6000 pixel resolution in HEIC format. After manual review to exclude out-of-scope images (cropped focal trays, trays with no grid cells, and trays with foliage covering more than half of the tray boundary), approximately 800 usable images were retained. 729 of these were labelled and used to train a YOLOv8 model to isolate single trays via a bootstrapped pipeline; high-confidence model predictions were manually reviewed before being added to the training set, with data split by tray identity to prevent leakage. A subset of 105 images was further annotated using a custom React-based labeling tool described in Section III-B. These annotations support downstream tasks including grid type classification and generation of labeled cell crops for training the viability classifier.

B. *Annotation Tooling*

Grid-type annotation required purpose-built tooling because no existing workflow supported the simultaneous cap-

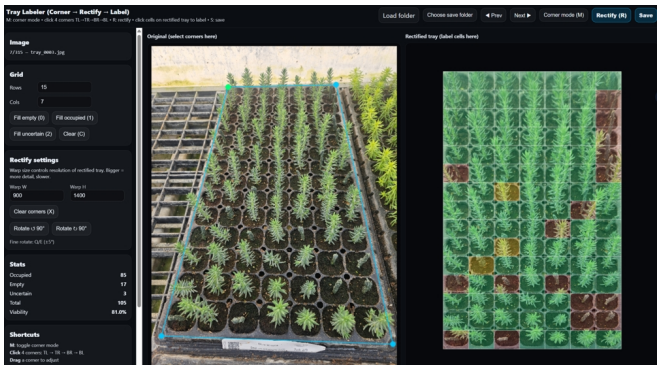

Fig. 3. Annotation tool interface- Left: raw tray image with manual-placed corner markers (TL→TR→BR→BL). Right: OpenCV.js perspective-rectified output with grid overlay and per-cell labels (green = occupied, red = empty, yellow = uncertain)

ture of tray geometry (four corner coordinates), grid configuration (rows, columns), warp output dimensions, and per-cell labels against a live rectified view. A browser-based annotation application was developed in React with OpenCV.js, enabling in-browser perspective rectification. (Fig. 3)

Using this tool, 105 tray images were annotated spanning 29 distinct grid configurations. These annotations form the training dataset for the grid type classifier and additionally provide the cell-level occupied/empty labels used to construct the 9,155-image dataset for the viability classifier.

C. *Tray Segmentation*

The tray segmentation component detects and isolates individual growing trays within each input image, producing a perspective-corrected crop passed to downstream modules. The task is framed as instance segmentation rather than simple bounding-box detection, because tray boundaries are rarely axis-aligned and a tight polygon mask is needed for accurate perspective warping.

1) Model Architecture and Training:

Tray segmentation is performed by a YOLOv8 instance segmentation model (YOLOv8n-seg), selected for its balance of inference speed and mask quality. The model was trained to detect a single class: the inner boundary of a growing tray.

Training used a two-stage schedule, beginning at 640×640 resolution and then fine-tuning at 1024×1024 to improve mask quality for downstream perspective warping. The final deployed model was trained on approximately 730 images using a bootstrapped labeling pipeline with tray-identity-based splits to prevent leakage.

2) Dataset Curation and Auto-Labeling:

Because manual annotation is the primary bottleneck for segmentation datasets, we used a bootstrapped labeling loop in which sufficiently strong model checkpoints generated candidate labels for new images, and high-confidence predictions were manually reviewed before inclusion in the training set. Data was split by tray identity rather than by image to prevent optimistic evaluation caused by multiple views of the same physical tray appearing across splits.

3) Inference and Acceptance Criteria:

At inference time, the model runs at 1024×1024 resolution with a low detection confidence threshold to maximize recall; the highest-confidence prediction is selected when multiple masks are returned. Each candidate mask is evaluated against three acceptance criteria before being passed downstream: (1) minimum detection confidence; (2) area fraction: the mask must cover a plausible proportion of the image; (3) convexity: the ratio of mask area to convex hull area must exceed a threshold, filtering irregular or partially-occluded detections. Images failing any criterion are routed to a manual review queue.

Accepted masks are converted to perspective-corrected crops via a four-corner estimation procedure: the mask polygon is rasterized, morphologically cleaned, and approximated by its convex hull. If the hull reduces cleanly to four corners via polygon approximation, those corners are used directly; otherwise a minimum-area rectangle is fitted as a fallback. The resulting four-point homography produces a rectified tray crop passed to subsequent pipeline stages.

D. *Grid Type Classifier*

Before individual cell crops can be extracted, the pipeline must determine the tray's grid configuration—the number of columns and rows—in order to place cell boundaries with geometric precision. We address this using a confidence-gated cascade: a learned classifier provides a fast global estimate from the rectified tray image, while a separator-based fallback (Section III-E) recovers grid structure when classifier confidence is low.

Grid type classification is formulated as a closed-set multi-class problem. Each class corresponds to a unique $(\text{cols}, \text{rows}, \text{warp_width}, \text{warp_height})$ tuple identified in the labeled dataset, yielding 29 classes in total. The warp output dimensions are included in the class label because two trays with identical column and row counts may have different physical aspect ratios (e.g., 7×9 at 900×900 vs. 900×1000), resulting in different visual grid structures. The input is a letterboxed, ImageNet-normalized RGB image resized to 384×640. Letterboxing preserves aspect ratio and avoids geometric distortion of grid spacing, which is the primary structural cue for classification.

1) Model Architecture:

The classifier uses a ResNet-18 backbone [11] pretrained on ImageNet and implemented via the `timm` library [12]. Global average pooling produces a 512-dimensional feature vector, followed by a linear layer mapping to 29 classes. The model contains approximately 11.2 million parameters. Differential learning rates are applied: 3×10^{-4} for the backbone and 3×10^{-3} for the classification head. Optimization is performed using AdamW with weight decay 1×10^{-4}.

2) Training and Augmentation Strategy:

The model is trained for 30 epochs with batch size 16. Label smoothing with $\epsilon = 0.05$ is applied to cross-entropy loss to

mitigate overconfidence on a sparsely sampled dataset. The best checkpoint is selected by validation accuracy.

Four augmentations are applied during training. Random erasing (probability 0.30, area 5-15%) occludes regions of the image to reduce reliance on plant canopy texture, encouraging the model to focus on structural features such as grid spacing and aspect ratio. This is motivated by the observation that different tray types are often photographed at different growth stages, introducing spurious correlations between plant appearance and grid type. Empirical results suggest that errors are primarily associated with low-sample classes rather than texture-driven confusion. Additional augmentations include horizontal flipping (0.50), brightness/contrast jitter (0.70), and Gaussian blur (0.20). Rotation and perspective augmentations are excluded because images are already geometrically rectified upstream.

3) Evaluation:
The best checkpoint (epoch 26) achieves 86.4% top-1 and 90.9% top-3 validation accuracy. Correct predictions exhibit a mean confidence of approximately 0.80, while incorrect predictions average approximately 0.47, indicating that model confidence is a useful indicator of reliability.

4) Failure Mode Analysis:
Most grid-classification errors occur in classes with only one or two training samples and are concentrated among visually similar tray layouts (e.g., aspect-ratio variants of 7 × 9 or 6 × 5 vs. 6 × 7). This pattern suggests that current errors are driven primarily by limited class support rather than by architectural limitations. Future improvements will focus on targeted data collection and multi-task prediction of rows and columns to improve sample efficiency.

5) Confidence-Based Prediction Gating:
Predictions are filtered using two rejection criteria. A prediction is labeled `UNKNOWN` if the maximum softmax probability is below a threshold of 0.90 or if a geometric alignment score falls below 0.08. The alignment score is computed by refining grid placement via a search over translation (dx, dy) and scale s, maximizing agreement between predicted grid lines and edges detected using a Canny operator:

$$\text{score}(dx, dy, s) = \frac{\sum(\text{edge_map} \cdot \text{line_mask})}{\sum(\text{line_mask})} \tag{1}$$

This refinement improves grid alignment and provides a secondary validation signal. Together, these criteria prevent low-confidence predictions from propagating to downstream cell extraction. Post-hoc temperature scaling is proposed as future work to further improve probability calibration without affecting classification accuracy.

E. *Grid-Based Computer Vision Fallback*

When tray-type classification does not satisfy the confidence threshold, tray structure is recovered using a separator-driven computer vision pipeline. Because upstream perspective rectification aligns the tray approximately with the image axes, lattice recovery can be treated as two 1-D periodic estimation problems rather than a full 2-D search.

The method first constructs a separator-oriented representation that emphasizes tray-structure pixels while suppressing canopy-dominated regions. Directional morphological operations then reinforce elongated horizontal and vertical separator responses and bridge local gaps (Fig. 4).

Projection profiles are then computed from the separator-enhanced image:

$$P_x(x) = \sum_{y=1}^{H} L(y, x), \tag{2}$$

$$P_y(y) = \sum_{x=1}^{W} L(y, x), \tag{3}$$

where L denotes the longlines image. Signals are smoothed prior to inference to suppress fragmented responses and high-frequency noise caused by incomplete separator visibility.

Separator spacing is estimated independently along each axis using normalized autocorrelation.

$$R(k) = \frac{\sum_i (s_i - \bar{s})(s_{i+k} - \bar{s})}{\sum_i (s_i - \bar{s})^2 + \epsilon}, \tag{4}$$

The dominant pitch is selected as the maximizing lag within a constrained interval. This favors repeated separator spacing over absolute line location, making the estimate more robust to partial occlusion.

After pitch estimation, a periodic lattice is synthesized along each axis. Peak detection on the smoothed projection signals yields candidate separator locations, and a phase estimate produces the expected separator sequence:

$$\hat{g}_n = \phi + nT, \tag{5}$$

For each predicted lattice location $\hat{g}_n$, the method searches a local window for observed evidence and snaps to the strongest nearby peak when available; otherwise, the line is placed at the predicted location:

$$g_n = \begin{cases} \arg\max_{q \in \mathcal{Q}_n} \tilde{P}(q), & \mathcal{Q}_n \neq \emptyset, \\ \hat{g}_n, & \mathcal{Q}_n = \emptyset, \end{cases} \tag{6}$$

This fallback-to-predicted rule preserves global geometric consistency under canopy occlusion. Resulting lines are filtered for minimum separation, edge margins are applied, and outer tray boundaries are inserted as terminal lattice lines (Fig. 4).

Cell crops are then obtained from consecutive pairs of lattice lines:

$$\mathcal{C}_{r,c} = I[y_r : y_{r+1},\, x_c : x_{c+1}], \tag{7}$$

This approach is effective because it relies on physical tray structure rather than variable plant appearance. Its main limitation is degraded performance when separator evidence is severely blurred or occluded.

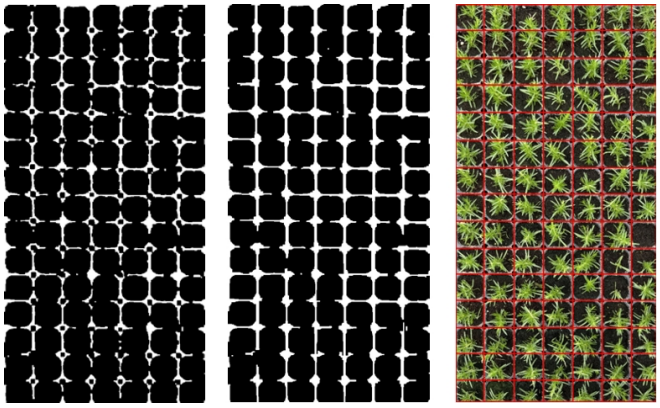

Fig. 4. Left: Initial separator mask obtained from color-based filtering, with visible noise and missing segments. Middle: Refined separator mask after morphological cleanup, producing a more regular grid structure for separator-based inference. Right: Final grid overlay on the rectified tray image, showing the estimated cell boundaries used for downstream cell extraction.

F. *Viability Classification*

The Viability Index (V_I) is defined as the ratio of germinated saplings to total tray capacity. Binary detection distinguishing "Occupied" (germinated) from "Empty" (non-germinated) cells was determined in collaboration with the project sponsor to provide the highest immediate value for inventory forecasting:

$$V_I = \frac{\sum_{i=1}^{N} C_i}{N} \tag{8}$$

where N is the total number of cells and $C_i = 1$ if classified as Occupied, 0 otherwise.

1) Model Architecture:

The cell validity classifier uses the same ResNet-18 backbone architecture as the grid type classifier (Section III-D), with the final fully-connected layer replaced by a 2-class output (Occupied vs. Empty).

2) Training Strategy and Data Augmentation:

The dataset consists of 9,155 unique sapling cell images extracted from warped tray images. As shown in Table I, the dataset exhibits a natural class imbalance (Occupied:Empty $\approx$ 3.3:1) reflecting the germination rates of the source nursery. A stratified random split maintains these proportions across all subsets. The dataset was partitioned into Training (80%), Validation (10%), and Testing (10%) sets. The model was optimized using Adam with learning rate 1×10^{-4} and weight decay 1×10^{-4} for 12 epochs using standard Cross-Entropy Loss. Data augmentation was applied exclusively to the training set, including random horizontal flips and rotations of up to 10°.

3) Spatial Buffering Optimization:

A critical challenge in grid-based classification is ensuring the entire sapling is captured within the cropped cell image, particularly when foliage overlaps the cell boundaries. Three spatial buffer scales were evaluated by expanding cell crop coordinates from the cell center (c_x, c_y):

$$x_{min} = \max(0, c_x - \frac{w \cdot B}{2}), \quad x_{max} = \min(W, c_x + \frac{w \cdot B}{2})$$

$$y_{min} = \max(0, c_y - \frac{h \cdot B}{2}), \quad y_{max} = \min(H, c_y + \frac{h \cdot B}{2})$$

TABLE I

VIABILITY DATASET CLASS DISTRIBUTION

Class Label	Total Samples	Percentage (%)
Occupied (Germinated)	7,036	76.85
Empty (Unoccupied)	2,119	23.15
Total	**9,155**	**100.00**

TABLE II

PERFORMANCE COMPARISON ACROSS SPATIAL BUFFER SCALES

Iteration	Buffer (B)	Accuracy (%)	Precision (%)	Recall (%)
Run 3	1.00	95.08	96.54	96.96
Run 2	1.05	95.52	96.73	97.42
Run 1	1.10	**96.74**	98.44	97.33

As shown in Table II, the $B = 1.10$ configuration achieved the highest accuracy (96.74%), with wider crops capturing overhanging foliage and compensating for minor misalignments in grid boundaries.

4) Performance Evaluation and Error Analysis:

The final model achieves 96.74% accuracy on a stratified test set. Qualitative analysis identifies two primary error modes: (1) false negatives in extremely early-stage germination where emerging leaf tissue is visually indistinguishable from localized soil shadows, and (2) false positives in later growth stages caused by foliage occlusion from adjacent cells. The latter is considered non-critical for industrial deployment, as the primary operational window is early-stage growth before foliage density creates high-occlusion environments.

To increase reliability, a threshold-based filtering strategy is applied: a cell predicted as "Occupied" is accepted only if its confidence exceeds a predefined threshold τ; otherwise, it is reassigned as "Empty." This asymmetric correction prioritizes precision in germinated cell detection, reducing false positives that would otherwise inflate the tray-level viability estimate. The confidence threshold τ is configurable at inference time and was empirically set to 0.90 to enforce conservative predictions in ambiguous scenarios.

G. *Pipeline Integration and Viability Scoring*

The four pipeline stages are integrated into a unified inference system that orchestrates data flow and produces a tray-level viability estimate.

1) Grid Detection via Confidence-Gated Cascade:

Grid detection is implemented as a confidence-gated cascade combining the learned classifier (Section III-D) and the separator-based grid segmentation module (Section III-E). The classifier serves as the primary predictor; when

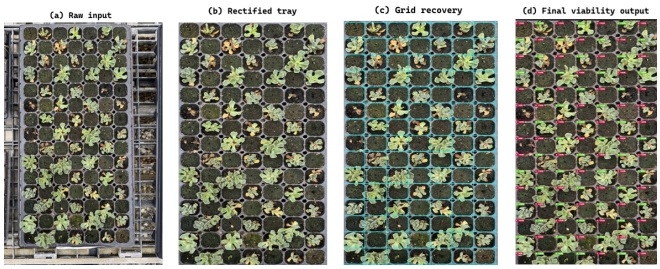

Fig. 5. End-to-end pipeline on a representative tray image. (a) Raw mobile-captured input image. (b) Perspective-rectified tray after segmentation and corner-based warping. (c) Recovered tray grid used for cell partitioning. (d) Final occupied/empty predictions with tray-level viability output.

confidence falls below the grid threshold τ_{grid}, the system falls back to the separator-based method.

Given a rectified tray image, the grid classifier predicts a grid configuration $(\text{cols}, \text{rows})$ with confidence score P_{cls}. If $P_{\text{cls}} \geq \tau_{\text{grid}}$, the prediction is accepted and refined using an edge-alignment procedure that adjusts translation and scale to maximize agreement with image gradients. If $P_{\text{cls}} < \tau_{\text{grid}}$, the separator-based module is used instead. This cascade design maintains efficiency for common tray types while providing a more geometry-driven fallback for lower-confidence cases.

2) Cell Extraction and Classification:
Using the inferred grid structure, the rectified tray image is partitioned into individual cells. A ResNet-18 validity classifier (Section III-F) is applied independently to each cell. A post-processing threshold downgrades low-confidence occupied predictions to empty.

3) Viability Scoring:
Tray-level viability is computed as:

$$V_I = \frac{N_{\text{occupied}}}{N_{\text{total}}} \times 100 \tag{9}$$

where N_{occupied} is the number of occupied cells and N_{total} is the total number of cells defined by the predicted grid. This metric aligns with standard nursery evaluation practices and provides an interpretable measure of germination success.

4) System Output and Deployment:
The integrated pipeline is deployed as a FastAPI service with a single `/analyze` endpoint that accepts a raw image and returns predicted grid configuration, per-cell classifications, and the final viability score, along with an annotated overlay image (Fig. 7) for visual interpretability. The system achieves practical low-latency inference, with a median runtime of 1.65 s per image on a 1,037-image production batch (see end-to-end evaluation below), and is accessible via a browser-based interface and a mobile iOS application (Fig. 6) for in-field capture.

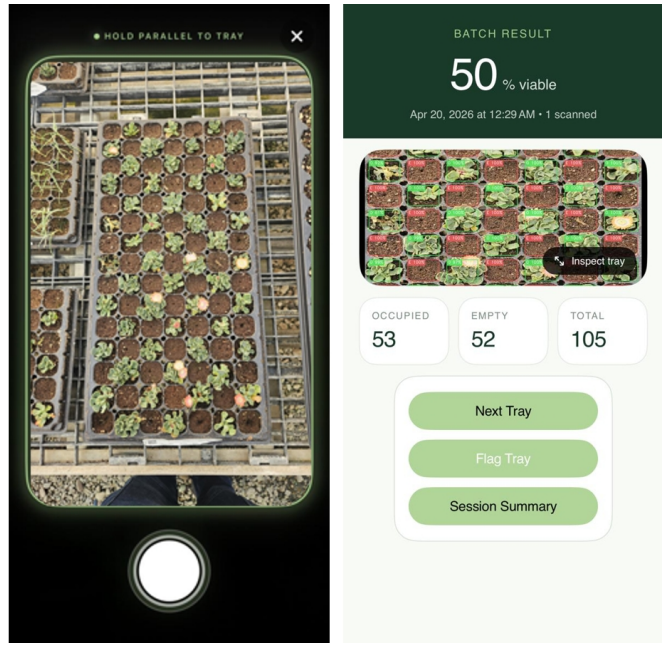

Fig. 6. Left: Capture interface of the mobile application, which guides users to acquire tray images under a consistent framing setup. Right: Result interface displaying the annotated tray image, per-cell occupancy predictions and tray-level viability metrics produced by the analysis pipeline.

IV. RESULTS

A. End-to-End Pipeline Evaluation

To evaluate the integrated system, we analyzed the full 1,037-image production batch collected under nursery conditions, of which 105 trays were additionally annotated and used as a ground-truth subset for end-to-end accuracy evaluation.

1) Viability accuracy:
On the 105-tray labeled subset, the pipeline achieved an exact grid-match rate of 88.6%, end-to-end cell accuracy of 91.0%, precision of 97.6%, recall of 90.4%, and F1 score of 93.9%. At the tray level, 72.4% of trays fell within ± 5 percentage points of ground-truth viability and 81.0% fell within ± 10 percentage points. The predicted-versus-ground-truth viability plot (Fig. 8) yielded a mean absolute error of 6.01 percentage points and a Pearson correlation of $r = 0.942$, indicating strong agreement between system output and manual annotation. The distribution of per-tray absolute errors (Fig. 9) is right-skewed, suggesting that while most trays are estimated accurately, a smaller number of difficult cases account for a disproportionate share of the residual error.

2) Runtime and deployment behavior:
Across the full 1,037-image production batch, median inference time was 1.65 s per image, with mean runtime 3.12 s. The separator-based fallback path was used on 68.4% of images under the current conservative confidence threshold, reflecting a robustness-first deployment setting for grid inference. The runtime distribution (Fig. 10) shows that most trays were processed quickly, while a smaller number of harder cases produced a longer right tail. These results support

Fig. 7. Example output of the proposed pipeline. Each cell is classified as occupied (green) or empty (red), with corresponding confidence scores. The system aggregates cell-level predictions to compute the tray-level viability percentage.

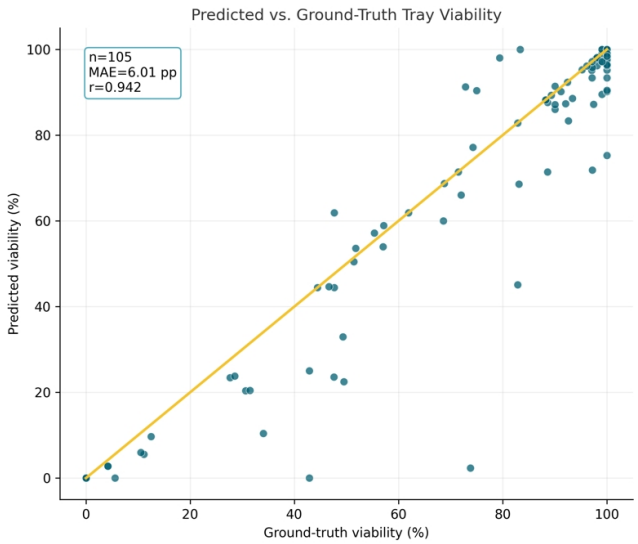

Fig. 8. Predicted versus ground-truth tray viability on the 105 labeled evaluation trays. Most points cluster near the identity line, indicating strong agreement between automated and annotated viability estimates.

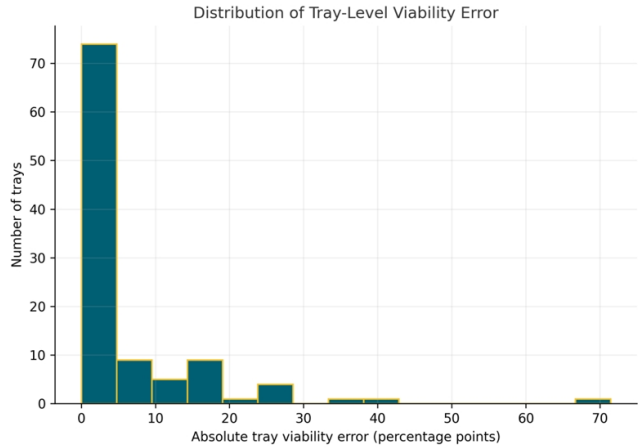

Fig. 9. Histogram of absolute tray-level viability error on the labeled evaluation set. Most trays have relatively small error, while a smaller number of difficult cases contribute disproportionately to the overall RMSE.

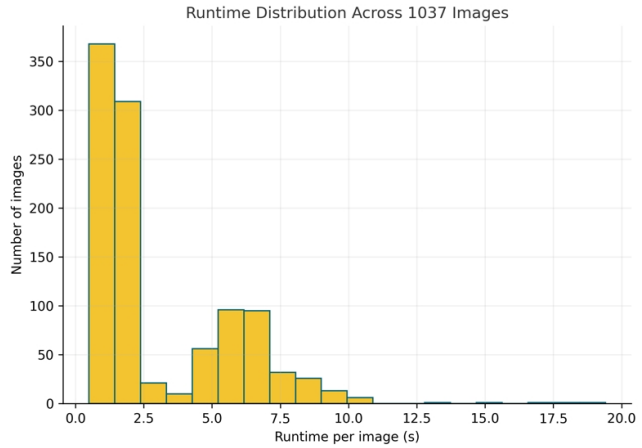

Fig. 10. Runtime distribution across the 1,037-image production batch. Most images were processed quickly, with a smaller number of harder cases contributing to a longer right tail in processing time.

practical low-latency deployment in mobile-assisted nursery workflows.

*3) **Production-batch output characteristics**:* Across the full 1,037-image production batch, the median predicted tray viability was 97.14%, with 33.1% of trays predicted as fully occupied and 3.0% predicted as fully empty. The batch also exhibited a wide range of detected tray structures, with 111 unique predicted grid types. Together, these findings suggest that the system remains operationally stable across diverse real-world tray images while maintaining consistent structured output generation.

V. DISCUSSION

End-to-end evaluation on the 105 labeled trays, drawn from the full 1,037-image production batch, shows that strong per-stage performance translates to accurate tray-level viability estimation. On the labeled subset, the integrated pipeline achieved a tray-level mean absolute error of 6.01 percentage points with Pearson correlation $r = 0.942$, while exact grid configuration was recovered in 88.6% of trays and end-to-end cell classification reached 93.9% F1. These results indicate that the system is effective for practical early-stage inventory estimation, while also suggesting that grid misclassification remains a major source of residual error because it propagates systematic cell-boundary mistakes across an entire tray.

The confidence-gated cascade for grid detection is a central strength of the system. The learned classifier provides fast predictions for common tray configurations, while the separator-based fallback handles lower-confidence or out-of-distribution cases. In the full 1,037-image production batch, the fallback path was used on 68.4% of images under the current conservative confidence threshold. This high fallback rate reflects a deliberate robustness-first deployment choice motivated by the limited size of the grid-classification training set, which contains 105 annotated trays spanning 29 grid configurations.

The tray-level and cell-level results also clarify the system's error profile. End-to-end cell precision was 97.6%, while recall was 90.4%, indicating a conservative tendency

when assigning the occupied label. This is operationally reasonable in a nursery setting where overstating viability may be more harmful than underestimating it. At the tray level, 72.4% of labeled trays fell within ± 5 percentage points of ground-truth viability and 81.0% fell within ± 10 percentage points, while the gap between mean absolute error and root mean squared error suggests that most trays are estimated well and a smaller number of difficult cases account for a disproportionate share of the residual error.

The production-batch analysis further supports practical deployment. Across the full 1,037-image batch, median inference time was 1.65 s per image and mean runtime was 3.12 s, supporting low-latency use in mobile-assisted workflows. The system also remained operationally stable across the batch, producing structured outputs over a wide range of tray configurations, with 111 unique predicted grid types observed.

Several limitations remain. The grid classifier is trained on a relatively small labeled set with uneven support across tray classes, which likely contributes to the high fallback rate and to some of the largest tray-level errors. In addition, the 105 labeled trays used for quantitative accuracy evaluation are a subset of the same 1,037-image production batch rather than an independently collected external test set, and all data were collected from a single commercial nursery. Finally, the system is intentionally scoped to early-growth-stage viability and does not attempt finer-grained health assessment or later-stage occlusion handling.

Overall, the results demonstrate that a modular computer vision pipeline can provide accurate, scalable, and operationally useful tray-level viability estimation from mobile-captured images. The clearest path for improvement is expanding the labeled grid-classification dataset, reducing reliance on fallback for common tray types, and validating the pipeline across more diverse production environments.

VI. Conclusion And Future Work

This work presents an end-to-end computer vision pipeline for automated seedling viability assessment, replacing subjective manual inspection with a consistent and scalable data-driven approach. By combining YOLOv8-based tray segmentation and perspective rectification with ResNet-18 models for grid type classification and cell-level viability assessment, the integrated system achieved a tray-level mean absolute viability error of 6.01 percentage points and Pearson correlation $r = 0.942$ on 105 annotated trays, while maintaining a median runtime of 1.65 s per image on a 1,037-image production batch.

The integrated pipeline - from mobile image capture to backend inference and structured output - demonstrates the feasibility of low-latency viability estimation in operational workflows. The modular design enables reliable grid detection, robust cell extraction, and interpretable tray-level metrics, supporting informed decision-making in large-scale plant propagation.

Future work will focus on extending the classification to finer-grained plant health indicators, such as distinguishing between cotyledons from true leaves, expanding the dataset with images from additional nurseries to evaluate generalization across tray types and imaging conditions, and exploring a multi-task grid formulation that predicts column and row counts independently to improve performance on data-limited classes.

Acknowledgment

This work was supported in part by BloomLogic. We would like to thank Cody Olsen (BloomLogic), for his valuable guidance, technical insights, and continued support throughout this project. This work was also supported in part by the University of Utah Deep Learning in AI & Robotics (DLAIR) Program.

References

[1] J. R. Ubbens and I. Stavness, "Deep plant phenomics: A deep learning platform for complex plant phenotyping tasks," *Frontiers in Plant Science*, vol. 8, p. 1190, 2017. [Online]. Available: https://doi.org/10.3389/fpls.2017.01190 (accessed Apr. 2, 2026).

[2] A. L. D. S. Safre, A. F. Torres-Rua, B. L. Black, and S. Young, "Deep learning framework for fruit counting and yield mapping in tart cherry using YOLOv8 and YOLO11," *SSRN*, 2025. [Online]. Available: https://doi.org/10.2139/ssrn.5192045 (accessed Apr. 5, 2026).

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2014, pp. 580-587.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Adv. Neural Inf. Process. Syst. (NIPS)*, 2015, pp. 91-99.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 779-788.

[6] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics (accessed Apr. 2, 2026).

[7] M. V. Giuffrida, A. Dobrescu, P. Doerner, and S. A. Tsaftaris, "Leaf counting without annotations using adversarial unsupervised domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, 2019, pp. 2590-2599.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 2961-2969.

[9] M. P. Pound et al., "Deep machine learning provides state-of-the-art performance in image-based plant phenotyping," *GigaScience*, vol. 6, no. 10, 2017.

[10] R. R. Shamshiri et al., "A review of greenhouse climate control and automation," *Int. J. Agric. Biol. Eng.*, vol. 11, no. 1, pp. 1-22, 2018.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770-778.

[12] R. Wightman, "PyTorch image models," GitHub repository, 2019. [Online]. Available: https://github.com/huggingface/pytorch-image-models (accessed March. 18, 2026).

[13] M. H. Saleem, J. Potgieter, and K. M. Arif, "Plant disease detection and classification by deep learning," *Plants*, vol. 8, no. 11, p. 468, 2019.