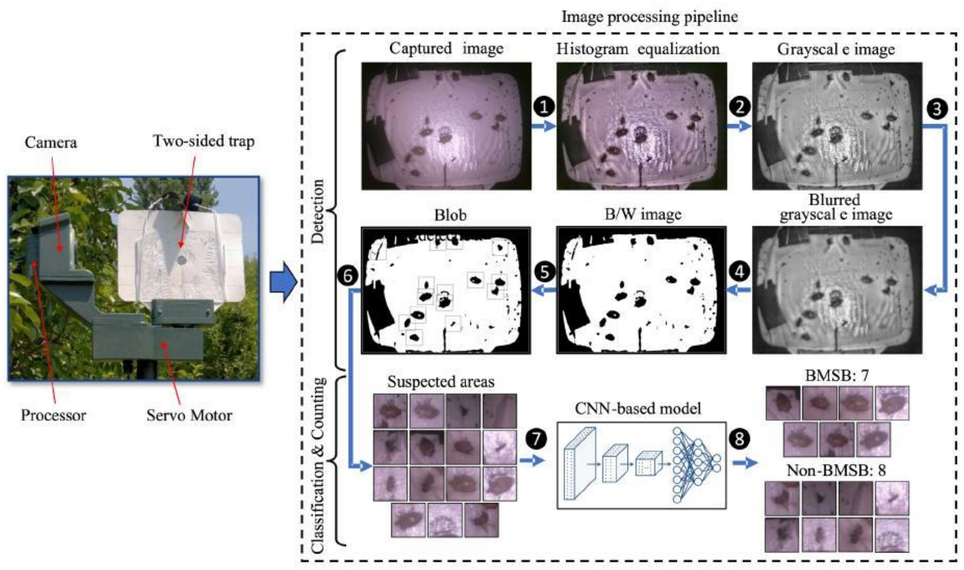

Fig. 4. Smart sticky trap (upper) and the steps followed for the detection and classification approach (lower).

develop datasets has been described in [43] and [44]. Recent studies have proposed edge-based systems, such as [11] for precision agriculture, most of which suggested power-hungry hardware, such as Raspberry Pi for computation. This research utilized a microcontroller (MCU)-based system to minimize power consumption, which is a crucial factor for edge-based systems, particularly in agricultural settings where devices must operate on batteries due to the absence of power lines in orchards. The image processing solution is here compared to other commonly used approaches, this time using real image data captured in the field throughout the deployment of the IoT device in orchards for several months in 2023. In addition, the impact of considering past data on the system's accuracy is examined.

A. Trap Architecture

We designed and developed an edge-based smart trap to make the identification and insect counting procedure automated and more accurate. The system is an ultra low-cost and low-power device able to detect and count insects on the device itself in the orchard without relying on expensive and power-hungry equipment. Fig. 4 depicts the second version of the proposed device. As described in [44] in detail, the camera unit is an OpenMV board based on an MCU with an embedded camera capable of running lightweight image processing and deep learning methods. In addition, this device supports two-sided traps and a servo motor is used to rotate the trap, so the camera can capture images of the trap on both sides. The purpose of the mechanism is to increase the probability of capturing insects (as the captured area is doubled) and, at the same time, make it more approachable to insects since the sticky area is not shadowed by the camera. The image processing pipeline is shown in Fig. 4. This is a modified version of the image processing pipeline proposed in [44].

The analysis of real-world data captured in 2023 revealed that the detection phase was unable to properly detect insects. Consequently, we modified and enhanced this aspect by implementing a histogram equalization technique on the captured images to normalize the contrast and brightness. This adjustment, denoted as the first step in Fig. 4, resulted in a significant improvement in the output of the detection phase, reducing both the mean squared error (mse) from 50.8 to 13.6 and mean absolute error (MAE) from 5.1 to 2.1, indicating a reduction by a factor of approximately 3.7 in mse and approximately 2.5 in MAE. This highlights how important it is to incorporate a wide range of real-world images into the algorithm development and help the algorithm adapt to the complexities of real-world situations.

TABLE IV
COMPARISON OF DIFFERENT MODELS ON THE NEW DATASET COLLECTED FROM THE REAL WORLD FOR BMSBS COUNTING

Model	MAE	MSE	P	R	MCU Comp.
YOLOv8-N	0.71	1.33	86.6%	98.5%	No
YOLOv5-N	0.88	0.55	85.1%	97.6%	No
YOLOv3-N	0.5	0.40	88.9%	98.8%	No
Ours in [44]	5.07	50.79	88.5%	36.7%	Yes
This study	2.07	13.55	84.0%	70.6%	Yes
This study (7-day)	1.05	1.81	83.0%	80.2%	Yes

As shown in Fig. 4, in the detection phase, the process begins with the application of histogram equalization to the raw captured image (①), followed by conversion to gray scale (②). Then, a smoothing filter is applied to eliminate small objects, such as windblown dust, leaves, or tiny insects (③). Then, the image is converted to black-and-white using Otsu's method (④) and finally the suspected area that might be filled by the target insect is detected using a blob detection algorithm (⑤). It should be noted that the detected blobs are filtered based on their size and only those with a size approximating that of a BMSB are considered as suspected areas. So, these suspected areas are cropped and fed to the classification phase (⑥), where a lightweight CNN-based image classification model is used to classify the images (⑦). Finally, the number of the target insects on the trap along with the cropped images (as an option) is sent to the growers (⑧).

Our CNN-based model has a program size of only 80 kB, with a peak memory usage of 75 kB and an inference time of 0.07 s (on the OpenMV board), making it suitable for running on memory-constrained MCUs. In designing the model, as detailed in [44], the main considerations are itemized as follows.

1) *Utilizing depthwise separable convolution instead of standard convolution:* This reduces computations by factorizing the process, significantly decreasing computational cost.
2) *Using global average pooling instead of flatten layer:* This decreases parameter numbers, leading to a substantial reduction in computation and model size.
3) *Quantizing:* The model weights were quantized to 8-bit integers from the original 32-bit float data, which is critical for deploying the model on memory-constrained MCUs.
4) *Utilizing skip connections:* Skip connections enhance performance by facilitating data flow, combining features from different layers, and addressing vanishing gradient issues during training.

B. Experimental Results

Table IV reports the accuracy of the proposed BMSB identification and quantification based on images from the new dataset gathered from the real world as compared to that proposed in [44] using known YOLO methods that are widely used in