



Article

ALIKE-APPLE: A Lightweight Method for the Detection and Description of Minute and Similar Feature Points in Apples

Xinyao Huang^{1,2,3}, Tao Xu^{1,2,3}, Xiaomin Zhang^{1,2,3}, Yihang Zhu^{1,2,3}, Zheyuan Wu^{1,2,3}, Xufeng Xu^{1,2,3}, Yuan Gao^{1,2,3} , Yafei Wang⁴ and Xiuqin Rao^{1,2,3,*} 

¹ College of Biosystems Engineering and Food Science, Zhejiang University, Hangzhou 310058, China; 22113013@zju.edu.cn (X.H.); 12113053@zju.edu.cn (T.X.); 11813014@zju.edu.cn (X.Z.); yihang_zhu@zju.edu.cn (Y.Z.); 22213052@zju.edu.cn (Z.W.); 12213062@zju.edu.cn (X.X.); yuang@zju.edu.cn (Y.G.)

² Key Laboratory of Intelligent Equipment and Robotics for Agriculture of Zhejiang Province, Hangzhou 310058, China

³ National Key Laboratory of Agricultural Equipment Technology, Hangzhou 310058, China

⁴ Zhejiang Agricultural Machinery Industry Association, Hangzhou 310007, China; hgwyf888@126.com

* Correspondence: xqrao@zju.edu.cn

Abstract: Current image feature extraction methods fail to adapt to the fine features of apple image texture, resulting in image matching errors and degraded image processing accuracy. A multi-view orthogonal image acquisition system was constructed with apples as the research object. The system consists of four industrial cameras placed around the apple at different angles and one camera placed on top. Following the image acquisition through the system, synthetic image pairs—both before and after transformation—were generated as the input dataset. This generation process involved each image being subjected to random transformations. Through learning to extract more distinctive and descriptive features, the deep learning-based keypoint detection method surpasses traditional techniques by broadening the application range and enhancing detection accuracy. Therefore, a lightweight network called ALIKE-APPLE was proposed for surface feature point detection. The baseline model for ALIKE-APPLE is ALIKE, upon which improvements have been made to the image feature encoder and feature aggregation modules. It comprises an Improved Convolutional Attention Module (ICBAM) and a Boosting Resolution Sampling Module (BRSM). The proposed ICBAM replaced max pooling in the original image feature encoder for downsampling. It enhanced the feature fusion capability of the model by utilizing spatial contextual information and learning region associations in the image. The proposed BRSM replaced the bilinear interpolation in the original feature aggregator for upsampling, overcoming the apple side image's geometric distortion and effectively preserving the texture details and edge information. The model size was shrunk by optimizing the number of downsampling operations from the image encoder of the original model. The experimental results showed that the average number of observed keypoints and the average matching accuracy were improved by 166.41% and 37.07%, respectively, compared with the baseline model. The feature detection model of ALIKE-APPLE was found to perform better than the optimal SuperPoint. The feature point distribution of ALIKE-APPLE showed an improvement of 10.29% in average standard deviation (Std), 8.62% in average coefficient of variation (CV), and 156.12% in average feature point density (AFPD). Moreover, the mean matching accuracy (MMA) of ALIKE-APPLE improved by 125.97%. Thus, ALIKE-APPLE boasts a more consistent allocation of feature points and greater precision in matching.



Citation: Huang, X.; Xu, T.; Zhang, X.; Zhu, Y.; Wu, Z.; Xu, X.; Gao, Y.; Wang, Y.; Rao, X. ALIKE-APPLE: A Lightweight Method for the Detection and Description of Minute and Similar Feature Points in Apples. *Agriculture* **2024**, *14*, 339. <https://doi.org/10.3390/agriculture14030339>

Academic Editors: Ziliang Liu, Jun Wang and Lizhen Deng

Received: 9 January 2024

Revised: 11 February 2024

Accepted: 12 February 2024

Published: 21 February 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: local feature; apple; deep learning; feature detection; image matching

1. Introduction

The appearance of fruit is crucial in attracting consumers. With the help of machine vision technology, the quality of fruit can be ensured. Existing machine vision technology

uses multi-view imaging and image stitching to acquire fruit surface information comprehensively. Furthermore, indicators [1–3] such as color, texture, and defects are accurately extracted to assess fruit appearance quality.

The image feature extraction method is the key to image stitching [4]. Image feature extraction techniques extract keypoints from an image and create appropriate descriptors based on the keypoints' attributes. Traditional hand-crafted local feature extraction methods, including algorithms such as SIFT [5], SURF [6], BRISK [7], ORB [8], and KAZE [9], generally use a two-stage pipeline. During the keypoint detection phase, various algorithms are employed to extract keypoints from an image. These algorithms use different strategies and mathematical models. For instance, the SIFT algorithm searches for local extremes in the DoG pyramid, and the SURF algorithm approximates Hessian matrix-based [10] detection using a Haar [11] wavelet. The ORB algorithm selects the FAST corners using the Harris response, and the KAZE algorithm employs different scaling spaces and nonlinear diffusion for feature extraction. After identifying the keypoints, the next step is to compute local descriptors for each one. These descriptors, such as the Hessian matrix of local intensity values for SIFT, the integral image strategy for SURF, the grayscale center-of-mass vector for ORB, and the nonlinear diffusion for KAZE, define the criteria for extracting the keypoints. However, these methods are unsuitable for large-angle image rotations, texture similarity, or tiny keypoints. Therefore, a more advanced and discriminative approach is required to describe image features.

The method for detecting keypoints based on deep learning [12] can achieve better results than traditional methods by learning to extract more distinctive and descriptive features, which has expanded the range of applications and improved the accuracy of keypoint detection. SuperPoint [13] introduced a fully convolutional model that jointly computed pixel-level keypoint locations and descriptors by feeding a full-size image in a single forward pass. Other approaches, such as R2D2 [14], integrated feature detection into the entire matching pipeline by training jointly with feature description and matching. R2D2 introduced the Siamese decoding structure [15], which obtains high matching performance through the discriminability of the extracted points. DISK [16] utilized reinforcement learning principles to optimize end-to-end for numerous correct feature matches. In contrast to methods that rely on fixed acceptance fields, ALIKE [17] introduced a deformable descriptor detection head that learns the deformable position of each keypoint, enabling the extraction of descriptors with intense expressiveness and high accuracy. However, current deep learning-based feature extraction methods are still deficient in dealing with the similarity of feature structures and feature minutiae on apple surfaces and fail to consider these essential feature channels and spatial domains sufficiently. Therefore, these existing methods may not accurately identify the surface features of apples, which could lead to the loss of critical information and affect the detection results.

To solve the problems mentioned above, a deep learning-based algorithm (ALIKE-APPLE) for extracting the keypoints of an apple image was proposed. ALIKE-APPLE is a convolutional neural network that combines an Improved Convolutional Attention Module (ICBAM) and a Boosting Resolution Sampling Module (BRSM). This combination allows the network to generate uniformly distributed keypoints and their corresponding descriptors. This is particularly useful for finding reliable correspondences between image pairs, even when the texture in the images is sparse. Combining the proposed feature extraction method with the state-of-the-art feature matching algorithm improved the accuracy and reliability of image matching. It provides a credible reference for subsequent external quality detection and the grading of spherical-like fruit.

2. Materials and Methods

2.1. Experimental Materials

The experiment was conducted in Hangzhou City, Zhejiang Province, China. The experimental samples were 96 fresh apples purchased in May 2023 at the Hangzhou apple

trading market. The apples were stored at room temperature and left for 24 h before the experiment was conducted.

2.2. Multi-View Orthogonal Image Acquisition System

The apple multi-view orthogonal image acquisition system mainly consisted of a color industrial camera (Hikvision, MV-CA023-10GC, with a resolution of 1920×1200 pixels, Hangzhou, China), an 8 mm lens, an LED light source, a fruit tray, and a computer. The overall schematic diagram of the apple multi-view orthogonal image acquisition system is shown in Figure 1. To gather comprehensive information about the surface of an apple, a layout system comprising four circumferential cameras and one top-view camera was used. The cameras were positioned at 90-degree angles to each other to form an orthogonal system. LED light sources were arranged on both sides of each camera in a symmetrical distribution to create a uniform and bright light environment in the lightbox.

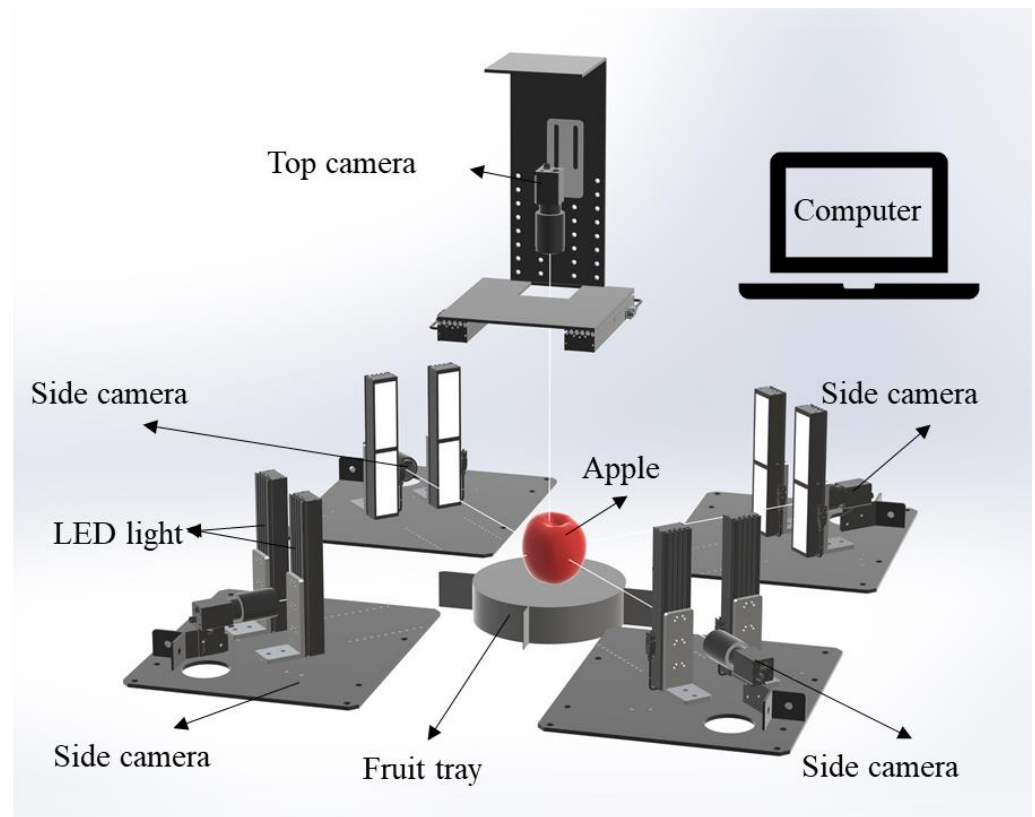


Figure 1. The apple multi-view orthogonal image acquisition system schematic diagram.

All models were trained on a deep learning platform with an Intel® Core i9-10900F CPU @ 2.80 GHz, 32 GB RAM, and an NVIDIA GeForce RTX 3060 GPU (BUYNOW CORPORATION, Hangzhou, China). The PyCharm integrated development environment was used to implement image processing algorithms. The Anaconda package manager and environment manager were employed to control the system runtime environment. PyTorch1.12.0+cu116 deep learning framework and OpenCV computer vision library were utilized to develop these algorithms.

2.3. Apple Dataset Construction

A total of 96 sets of apple images were captured by five cameras simultaneously, resulting in 480 images. Data augmentation methods [17] were employed to increase data diversity, including rotation, flipping, brightness, and contrast adjustments. Through this process, data augmentation was successfully performed on the apple dataset, and

480 images were generated. The images before and after data augmentation were integrated, totaling 960 images.

The 960 synthetic image pairs before and after the transformation were obtained by manipulating each image by applying random transformations [13,18], such as translating, rotating, scaling, or distorting. The training, validation, and test sets were divided according to 8:1:1, where the validation set was applied in advance with transformation to construct the synthetic image pairs.

2.4. A Lightweight Algorithm for Tiny and Similar Apple Feature Extraction

A lightweight apple feature extraction method (ALIKE-APPLE) based on improved ALIKE [19] is proposed. ALIKE-APPLE aims to solve the problem of difficult detection caused by the similarity of apple features and minor features and improve the model's ability to perceive image details. The overall structure of the model is shown in Figure 2a.

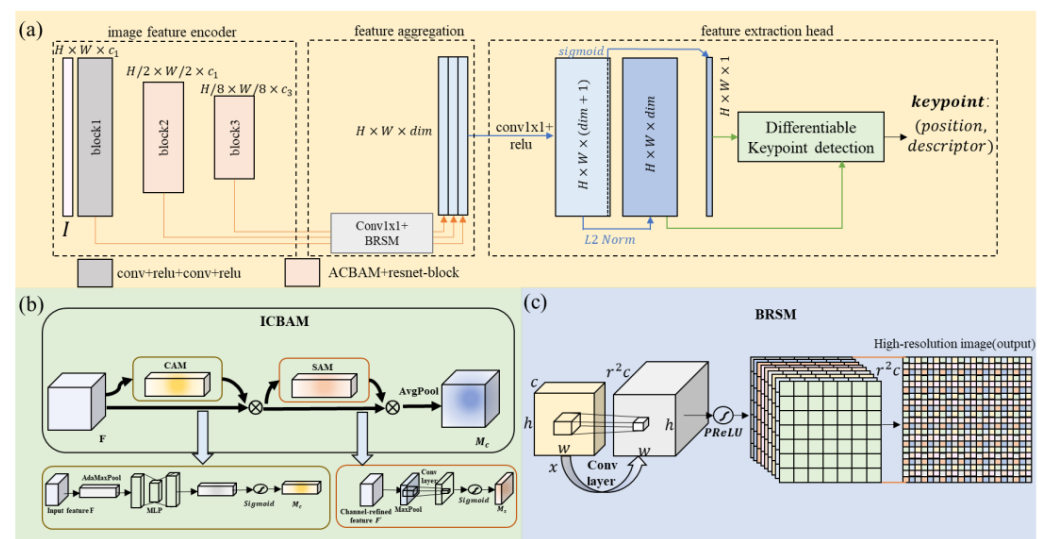


Figure 2. ALIKE-APPLE algorithm framework: (a) its components, (b) ICBAM block, and (c) BRSM block.

The model consists of three main parts: (1) The image feature encoder encodes the input image as a feature map with three blocks. The first block is a two-layer 3×3 convolution with “ReLU” activation [20]. The last two blocks contain an ICBAM and a 3×3 basic ResNet block [21]. The downsample rate of the ICBAM in block 2 is 1/2 and 1/4 in block 3. (2) The feature aggregation module aggregates multi-level features from the encoder. A 1×1 convolution and the BRSM upsampling are first used to adapt the channels and then concatenated. (3) The feature extraction head [19] outputs an $H \times W \times (\text{dim} + 1)$ feature map in which the first dim channels are L2 normalized as the descriptor map D and the last channel is normalized by “Sigmoid” activation as score map S . The differentiable keypoint detection and descriptor sampling first detects the sub-pixel keypoints from the score map S , and then samples their descriptors from the dense descriptor map D . The improvements of the proposed ALIKE-APPLE model over the ALIKE baseline model are specified below:

(1) The ICBAM replaces the max pooling in the image feature encoder for downsampling. Max pooling [22] is often used to reduce the dimensionality of the data in the traditional image encoding process. Still, this method may result in the loss of information in specific categories. The Convolutional Block Attention Module (CBAM) [23], on the other hand, fully captures the channel and spatial relationships between the feature mappings in a layer, enabling richer and more detailed feature information to be obtained in the downsampling process. However, the original design of the CBAM may not be able to adequately focus on minor texture differences and color variations in the apple image. In order to improve the model's ability to distinguish subtle differences, an ICBAM was

created. This was achieved by removing the average pooling layer from the channel and spatial attention modules of the CBAM and keeping only the max pooling layer. The image feature encoder part for downsampling now includes an ICBAM, which is designed to identify image changes caused by geometric distortion after fruit surface imaging. The ICBAM does this by utilizing spatial contextual information and learning region associations in the image, which helps to improve accuracy even with limited parameters and computational power. As shown in Figure 2b, the ICBAM mainly consists of a Channel Attention Module (CAM) and a spatial attention module (SAM). The ICBAM formulas are shown in (1)–(5). The adaptive max pooling layer [24] is first initialized in the CAM to integrate the spatial dimensions of the input feature map R^{C*H*W} to a size of 1×1 to obtain the spatial context descriptor $F \in R^{C*1*1}$. The descriptors are then fed into a multilayer perceptron [25] (MLP) to generate the channel attention map $M_C \in R^{C*1*1}$. The MLP consists of two convolutional layers and a ReLU activation function. Finally, by applying the Sigmoid activation function, the attention weight of each channel is made to fall in the range of 0 to 1, ultimately resulting in a one-dimensional convolution $M_C \in R^{C*1*1}$. The CAM utilizes adaptive max pooling [26] and MLP for feature channel adaptive weighting, allowing for the efficient extraction of essential features in the global context and improving the model's performance. The channel information of the feature maps is integrated into the SAM by a max pooling operation to generate a cross-channel feature map $F \in R^{1*H*W}$, which is concatenated and convolved through a standard convolutional layer [27]. This feature map then generates a 2D spatial attention map $M_S \in R^{1*H*W}$. The output of the spatial attention module is passed through an average pooling layer [28] to obtain the final output.

$$M_C(F) = \sigma(\text{MLP}(\text{AdaMaxPool}(F))) \quad (1)$$

$$M_S(F) = \sigma(f^{7 \times 7}(\text{MaxPool}(F))) \quad (2)$$

$$F' = M_C(F) \otimes F \quad (3)$$

$$F'' = M_S(F') \otimes F' \quad (4)$$

$$F''' = \text{AvgPool}(F'') \quad (5)$$

where *AvgPool* stands for average pooling layer; *AdaMaxPool* stands for adaptive max pooling layer; *MaxPool* stands for max pooling layer; MLP stands for multilayer perceptron; σ stands for sigmoid operation; and $f^{7 \times 7}$ denotes the convolutional kernel with a size of 7×7 convolutional operation.

(2) The proposed BRSM replaces the bilinear interpolation in the feature aggregation module for upsampling. In image processing or image restructuring tasks, common upsampling methods, including nearest-neighbor interpolation [29], bilinear interpolation [30], and bicubic interpolation [31], are typically employed. However, these methods suffer from edge blurring and loss of texture details. The PixelShuffle [32] method is a super-resolution model that reorganizes pixels for efficient sub-pixel convolution. It can improve accuracy and efficiency in feature aggregation by learning complex patterns from input data. However, PixelShuffle focuses only on pixel-level changes and does not consider the global or higher-level information of the image for model upsampling. A new solution called a BRSM has been developed to rectify the limitation inherent in the PixelShuffle method. It incorporates a convolutional layer along with the PixelShuffle technique to maintain intricate details at the pixel level while simultaneously capturing a more comprehensive view of the image. Additionally, a BRSM is used in the feature aggregation module to combine different levels of encoded features. This technique enables the model to learn complex pixel relationships, reduce information loss, and better preserve the original image's details and structure. The designed BRSM is shown in Figure 2c, and the computational equations are shown in (6)–(8). A convolution operation is first performed to extract features from the input data. The output of the convolutional layer is directly fed into the PreLU activation function [33]. PixelShuffle is used to transform high-dimensional feature maps into low-dimensional, high-resolution feature maps, achieving upsampling of

the feature maps. PixelShuffle has a step size of $1/r$ and organizes tensor element of shape $(*, C \times r^2, H, W)$ by rearranging them into the shape $(*, C, H \times r, W \times r)$. This method divides a low-resolution pixel into $[r \times r]$ parts, and a high-resolution pixel is formed from the feature pixels at the corresponding pixel positions of the feature map.

$$x' = Conv(c, c * factor * factor)(x) \tag{6}$$

$$x'' = PReLU(x') \tag{7}$$

$$x''' = PixelShuffle(factor)(x'') \tag{8}$$

where the input feature map is x ; the number of channels is c ; the upsampling factors $factor$, $Conv(c, c * factor * factor)$ represent a convolutional layer with c input channels and $c * factor * factor$ output channels (this operation enlarges the number of channels while keeping the spatial dimensionality of the feature map unchanged); x' is the output of the convolutional layer; x'' is the output of the convolutional layer; x'' is outputted by the PReLU activation function; the output x'' is passed to a PixelShuffle layer for processing; and x''' is the output feature map.

(3) Optimizing the number of feature encoder downsamples. In the task of detecting apple features, it is essential to maintain detailed information and capture global details. However, the resolution of the apple region in the training image is only 352×352 pixels. The feature encoder module in the original model performs downsampling three times. This downsampling process reduces the image resolution to 11×11 pixels, resulting in a significant loss of information in input feature maps, particularly in detailed information. In addition, a reduced sensory field can prevent the model from effectively capturing global or larger-scale contextual details. Therefore, downsampling operations must be performed carefully. The downsampling reduction strategy preserves detailed and global information while reducing computational loss. The image feature encoder was restructured by adjusting the number of downsamples from three to two, i.e., removing the last block. The purpose of this adjustment is to balance computational efficiency and information retention to better fit the training data's resolution.

2.5. Evaluation Indicators

The leading indicators used to evaluate the feature detection effect are standard deviation (Std), coefficient of variation (CV), and average feature point density (AFPD). For the Std, a more significant value indicates that keypoints are more widely spread out and evenly distributed. The CV measures the relative degree of variability of keypoints and the uniformity of their distribution. The AFPD measures the number of feature points detected in a unit area. In this study, the standard deviation (std) and the coefficient of dispersion (CV) of the keypoints were calculated in both horizontal and vertical coordinates using Equations (9)–(12). This analysis was conducted to evaluate the distribution of keypoints.

$$CV = std / MN \tag{9}$$

$$std_{axis} = \sqrt{\frac{1}{N} \sum_{i=1}^N (axis_val_i - \mu)^2} \tag{10}$$

$$MN = \frac{\sum_{i=1}^n axis_val_i}{n} \tag{11}$$

$$AFPD = \frac{\sum_{i=1}^n P_i}{n} \tag{12}$$

where $axis$ denotes the x direction or the y direction, $axis_val$ indicates the coordinate value in the x direction or the y direction, MN is the mean value, P is the number of points in the unit area, and n denotes the number of areas.

The ALIKE-APPLE method used Nearest Neighbor Matching (NN-matching) [34] and Optimal Transport Matching (OT-matching) [35] for feature extraction of apple images to obtain metrics for image matching. NN-matching determines the similarity between two feature vectors based on matching feature descriptor distances. The nearest neighbor is found for a given query point by calculating its distance from all other data points. A smaller distance indicates that the two feature vectors are more similar. OT-matching is a mathematical method to measure the similarity between two probability distributions. This method aims to find the optimal mapping for moving masses from one distribution to another to minimize the total transportation cost. Mean co-visible keypoint number (MCKN) and mean matching accuracy (MMA) are the primary metrics used to evaluate feature matching. A higher MCKN indicates more observed keypoints in the corresponding scenario. This means that the coverage of feature points is increased. A higher MCKN value indicates a broader coverage of feature points in the scene, meaning that more keys are observed. The higher the MMA, the more accurately the feature points are matched, resulting in higher accuracy. Regarding image matching, the evaluation benchmarks will be thoroughly illustrated in the subsequent discourse: Image A and Image B have N'_A and N'_B extracted keypoints, respectively, which can be extracted as $N_{conv} = (N'_A + N'_B)/2$. Assumed matches $N_{putative}$ are obtained by mutual matching of descriptors, and finer inner matches N_{inlier} are obtained by evaluating the reprojection distances within different pixel thresholds. Referring to previous work [19], the following metrics are used:

(1) Mean co-visible keypoint number, MCKN:

$$MCKN = \frac{1}{N} \sum_{k=1}^N (N'_{Ak} + N'_{Bk})/2 \quad (13)$$

(2) Mean matching accuracy, MMA:

$$MMA = N_{inlier}/N_{putative} \quad (14)$$

To evaluate the structural size and speed of the model, the following metrics were used: floating-point operations (FLOPs), number of parameters (Params), CPU image processing time per frame (Inference Speed-CPU), and GPU image processing time per frame (Inference Speed-GPU). FLOPs is a metric quantifying an algorithm's computational complexity involving floating-point units. Params refer to the total count of parameters that need to be trained. In addition, the CPU is a more versatile and economical hardware device than the GPU. It can better match the economic level of applying AI in agriculture. Therefore, the GPU and CPU performance were tested separately on the PC, and the inference speed was used as an evaluation metric. The inference speed can be calculated using the following steps. First, the model loads and processes 100 images. The total processing time is recorded and then divided by 100 to obtain the average processing time per image.

3. Results

3.1. ALIKE-APPLE Algorithm Structure Ablation Study

To evaluate and validate the performance and effectiveness of the ALIKE-APPLE algorithm, a series of ablation experiments were designed to analyze the effect of different algorithm components on its performance. First, ablation tests were performed with fewer downsamples from the feature encoder to evaluate the two modules, ICBAM and BRSM, respectively. The following four sets of experimental configurations were designed using the dataset constructed in Section 2.3 for image feature detection and matching for training as well as testing:

(1) ALIKE: Baseline configuration.

- (2) ALIKE-BRSM-ICBAM-SLIM (in this paper: ALIKE-APPLE): Based on ALIKE, both an ICBAM and a BRSM were introduced, and the number of downsamples were reduced to form an improved model.
- (3) ALIKE-BRSM-SLIM: Based on ALIKE-BRSM-ICBAM-SLIM, the ICBAM in the down-sampled header portion of the image feature extraction was ablated and the BRSM was retained as part of the feature integration.
- (4) ALIKE-ICBAM-SLIM: Based on ALIKE-BRSM-ICBAM-SLIM, the BRSM of the upsampling part of the image feature aggregation was ablated, and the ICBAM was retained as a part of the downsampling for feature detection.

During the testing process, the performance metrics of each experimental configuration were recorded and analyzed. This allowed for a comprehensive evaluation of the impact of different model combinations on image-matching performance, ultimately leading to the determination of the best model configuration. The training process of each model is shown in Figure 3. The slight difference in training iteration performance for model validation accuracy is worth mentioning. Accessing the ICBAM in the model structure decreased training loss.

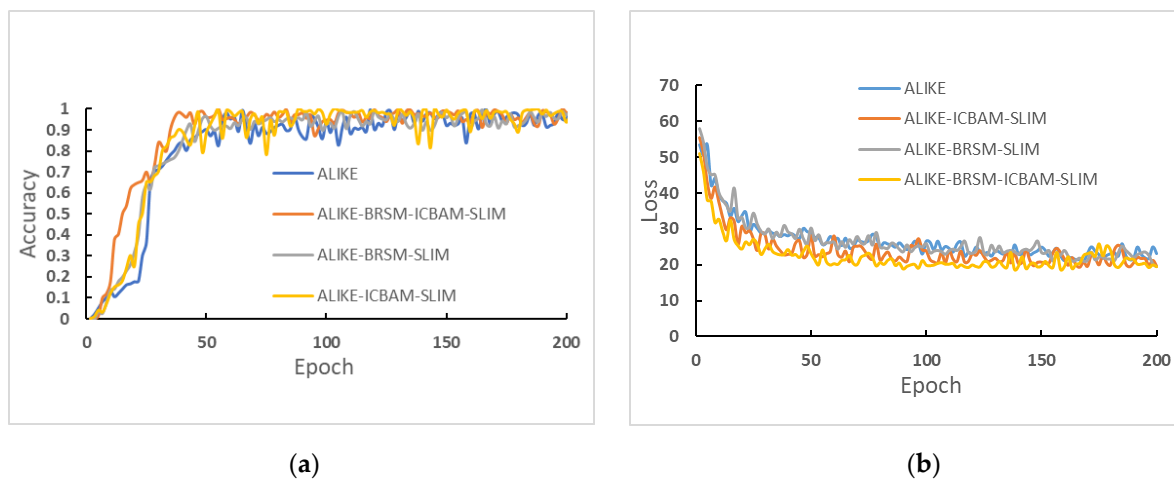


Figure 3. ALIKE-APPLE ablation training procedure: (a) accuracy of the validation set for each model; (b) loss of the validation set for each model.

The test results of each model on the test set after training are shown in Table 1. The experimental results were compared with ALIKE, ALIKE-BRSM-SLIM, and ALIKE-ICBAM-SLIM. The MCKN of ALIKE-APPLE improved by 166.41%, 124.76%, and 50.87%, respectively. The MMA was improved by 9.50%, 1.15%, and 3.46%, respectively. The inference speed of the three models after pruning SLIM was improved by 11.15%, 9.01%, and 7.06% on the CPU and 15.09%, 6.82%, and 3.50% on the GPU, respectively, over the original ALIKE baseline model. ALIKE-APPLE outperformed ALIKE on MCKN and MMA metrics. Additionally, ALIKE-APPLE reduced FLOPs and Params by 10.88G and 0.335MB, respectively, resulting in a lighter-weight model. This suggested that the ALIKE-APPLE model maintained high performance while requiring less computation and storage.

The model's performance was slightly degraded upon the removal of both the BRSM and the ICBAM, compared to the ALIKE-BRSM-ICBAM-SLIM model. This indicated that applying an ICBAM to downsampled locations and adding a BRSM to the upsampled network can enhance the model's MCKN and MMA.

After comparing the results of ALIKE, it was discovered that the ALIKE-APPLE algorithm reduced both FLOPs and Params metrics while increasing Inference Speed-CPU and Inference Speed-GPU metrics through downsampling. The optimization session revealed that adjusting the number of times the feature encoder was downsampled can improve the model's performance.

Table 1. Experimental results of model structure ablation.

Model (352 × 352 Pixels)	Performance			Lightweight		
	MCKN (Individual)	MMA (%)	FLOPs (G)	Params (MB)	Inference Speed-CPU (fps)	Inference Speed-GPU (fps)
(1) ALIKE	207.58	25.63	12.88	0.704	10.76	80.69
(2) ALIKE-BRSM-SLIM	246.05	33.98	10.82	0.324	11.96	95.03
(3) ALIKE-ICBAM-SLIM	366.56	31.67	10.20	0.178	11.73	86.60
(4) ALIKE-BRSM-ICBAM-SLIM (Ours: ALIKE-APPLE)	553.01	35.13	10.88	0.335	11.52	83.62

The ALIKE-APPLE model's performance was assessed across different hardware sessions. The model was able to achieve an inference speed of 11.52 frames per second on the CPU. In comparison, it reached an inference speed of 83.62 frames per second on the GPU. This suggested that ALIKE-APPLE had high real-time performance and could meet the demand for fast image matching in practical applications [36,37].

3.2. Comparative Study of Feature Detection Performance

The ALIKE-APPLE method was compared with five methods for feature detection and description, such as SuperPoint [13], DISK [16], R2D2 [14], and ALIKE [19]. All five methods were able to extract feature points from the apple image dataset. However, accurate extraction and assignment of features between images could become challenging when the points were too close together or when the texture was weak [18]. By contrast, when keypoints are distributed uniformly, it helps improve the accuracy and stability of matching and reconstruction.

The keypoint frequencies of different methods were counted, and the statistical results are shown in Table 2. Based on the statistical results, DISK had poor performance, and the range of detected feature points did not cover the entire apple, indicating poor uniformity. In contrast, SuperPoint, R2D2, and ALIKE-APPLE exhibited a more uniform distribution of keypoints, with a higher standard deviation and coefficient of variation. The keypoints identified by the ALIKE-APPLE algorithm had the highest standard deviation, coefficient of variation, and mean feature point density, and displayed the best overall performance. ALIKE-APPLE could extract features from both texture-rich regions and weakly textured regions. In summary, the ALIKE-APPLE algorithm performed best regarding uniformity of keypoint distribution and achieved robust feature extraction in the different areas. Thus, ALIKE-APPLE outperformed other methods in terms of comprehensive performance.

Table 2. The statistical indexes of keypoints of different methods.

Methods/Indicators	std_w	std_h	CV_w	CV_h	AFPD
SurperPoint	0.2756	0.2716	0.5615	0.5436	915.08
DISK	0.2196	0.2454	0.4487	0.5076	816.39
R2D2	0.2861	0.2853	0.5812	0.5658	430.32
ALIKE	0.2559	0.2534	0.5344	0.5525	382.83
Ours: ALIKE-APPLE	0.3039	0.2996	0.6065	0.5938	2343.74

3.3. Reliability Test Methods for Feature Detection

After the keypoints were extracted, the feature detection reliability test was performed by combining NN-matching and OT-matching. It was tested using a matching method called SuperGlue [38], which is based on deep learning. SuperGlue is an attention-based graph neural network that uses readily available local features to establish high-quality correspondences. It is worth noting that SuperGlue had two available pre-trained weight models that you could choose from. The first model was trained on the ScanNet dataset [39] for indoor scenes, while the second model was trained on the MegaDepth dataset [40] for

outdoor scenes. These models are referred to as “SG (in)” and “SG (out)”, respectively. Furthermore, SuperGlue employs SuperPoint for data augmentation while training and has been proven to work exceptionally well alongside SuperGlue and SuperPoint [38]. As a result, the effectiveness of combining SuperPoint and SuperGlue as a method for feature extraction and matching is compared. The results of the feature detection reliability test matching are shown in Figure 4.

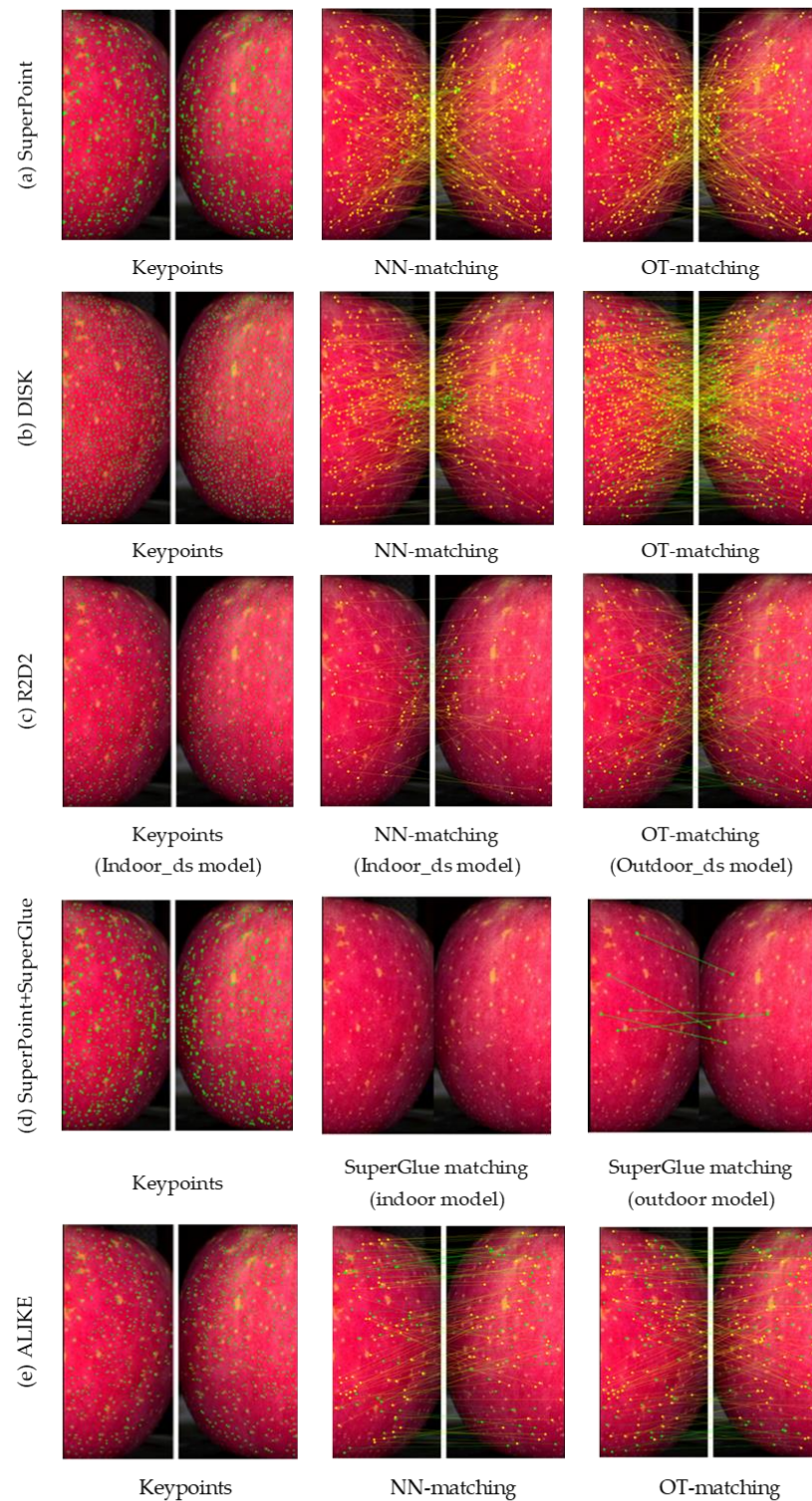


Figure 4. Cont.

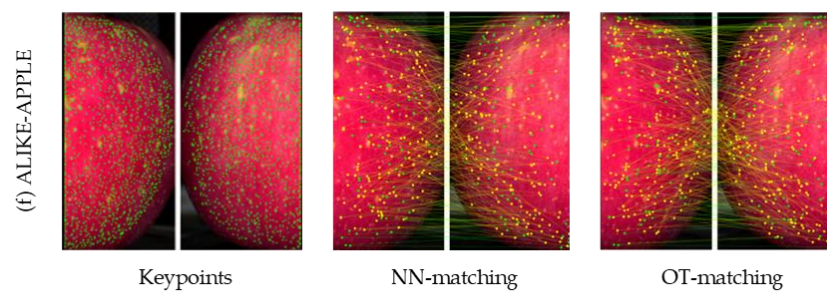


Figure 4. Qualitative comparison of apple image matching. (a) SuperPoint; (b) DISK; (c) R2D2; (d) SuperPoint + SuperGlue; (e) ALIKE; (f) ALIKE-APPLE (Ours). Green lines indicate correct matches, while yellow lines indicate mismatches.

By observing Figure 4a–c,e,f, it is evident that using NN-matching of keypoints resulted in a significant number of mismatches, indicated by the yellow lines. On the other hand, OT-matching improved the accuracy of matched pairs of points, leading to a decrease in the yellow line and an increase in the green line. After comparing five local feature extraction methods based on deep learning, it was observed that the ALIKE-APPLE algorithm showed an increased percentage of correctly matched green lines compared to the other algorithms. This indicates that the ALIKE-APPLE algorithm has improved the accuracy of matching. Additionally, it was observed that ALIKE-APPLE provided more correspondences in the untextured region, which is beneficial for incremental reconstruction. As a learned matching method, SuperGlue’s (Figure 4d) performance was heavily influenced by its weights. According to experiments conducted on SuperPoint localized features, the indoor model produced almost no accurate matches, whereas the outdoor model only produced a few scattered correct matches. These results suggest that the performance of SuperGlue was significantly influenced by the training dataset used.

The NN-matching algorithm and the OT-matching algorithm were compared side-by-side to calculate the MMA of the different methods. The results of the experiments are presented in Table 3, while the visualization results can be found in Figure 4. When using the NN-matching method, ALIKE-APPLE outperformed the SuperPoint, DISK, R2D2, and ALIKE models with 7.68%, 0.67%, 2.82%, and 0.30% higher average matching accuracies, respectively. The MMA of the proposed ALIKE-APPLE model was improved by 22.91%, 7.30%, 13.20%, and 15.00% when using the OT-matching method. The proposed ALIKE-APPLE model showed the highest matching rate, both operating NN-matching and OT-matching. According to the research, the proposed ALIKE-APPLE model showed a significant improvement in inference speed compared to other models such as SurperPoint, DISK, R2D2, and ALIKE. It resulted in 836.59%, 121.11%, 1103.63%, and 7.06% faster inference speed on the CPU, respectively, and 342.43%, 81.98%, 170.97%, and 3.63% faster inference speed on the GPU, respectively. Additionally, the ALIKE-APPLE model is lighter in weight compared to the other models. This showed that the proposed ALIKE-APPLE algorithm maintained high performance while having more efficient computation and storage requirements.

Table 3. The performance of different methods.

Feature Detector (352 × 352 Pixels)	NN-Matching	OT-Matching	Lightweight	
	MMA		Inference Speed-CPU (fps)	Inference Speed-GPU (fps)
SurperPoint	10.67	12.73	1.23	18.90
DISK	17.68	28.34	5.21	45.95
R2D2	15.53	22.44	0.9571	30.86
ALIKE	18.05	20.64	10.76	80.69
Ours: ALIKE-APPLE	18.35	35.64	11.52	83.62

4. Conclusions

The ALIKE-APPLE algorithm for detecting and describing feature points of apple images was established. Adding an ICBAM to the image feature encoder part for downsampling enhances the model's attention to the keypoints region. This allows the model to effectively recognize image changes caused by geometrical distortions after imaging apple surfaces. As a result, the model's detection performance and adaptability to complex scenes are improved. The added BRSM for feature aggregation helps the model learn more complex pixel relationships and better preserve the detail and structure of the original image. Reducing the number of downsampling times from three to two can increase the accuracy and richness of features by reducing the loss of information. The size of the model and the computational requirements can be reduced so that the model has a lighter feature. The ALIKE-APPLE model has been found to outperform other deep learning-based models such as SuperPoint, DISK, R2D2, and ALIKE when it comes to detecting image features in apples. The model detects the most uniform feature points with the best dispersion, highest density, and highest accuracy in matching. Additionally, it is the most lightweight among the models tested. Hence, it is the most suitable model for apple image matching. To further enhance the performance of the apple feature detection algorithm, it is necessary to construct a larger-scale and more diverse dataset. Hardware and software integration can be intensified by tailoring the PyTorch platform-based algorithm to the online system's hardware requirements, enhancing model detection efficiency via hardware acceleration. The study-proposed algorithm, instrumental in apple 3D reconstruction, enables a more accurate characterization of apple phenotypes and promotes the accuracy and standardization of apple classification, thereby augmenting commercial apple quality and market stability.

Author Contributions: Conceptualization, X.H. and X.R.; methodology, X.H. and Y.Z.; software, X.H.; validation, X.H., T.X. and X.Z.; formal analysis, X.H.; investigation, X.H.; resources, X.H., X.R. and Y.W.; data curation, X.H.; writing—original draft preparation, X.H.; writing—review and editing, X.H., T.X., X.Z., Y.Z., Z.W., X.X., Y.G. and X.R.; visualization, X.H.; supervision, X.H. and X.R.; project administration, X.R. and X.H.; funding acquisition, X.R. and Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the national key technology support program of China (2023YFD1600301) and the Key R&D Program of Zhejiang Province (2023C02007).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Acknowledgments: The authors gratefully acknowledge the technical support of Yicheng Xu, whose expertise and assistance with SolidWorks were indispensable in drawing the parts of the machine vision system device.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. *UNECE Standard FFV-50*; Apples. United Nations Economic Commission for Europe: New York, NY, USA; Geneva, Switzerland, 2020. Available online: <https://unece.org/trade/wp7/FFV-Standards> (accessed on 10 January 2023).
2. United States Department of Agriculture. *United States Standards for Grades of Apples for Processing*; United States Department of Agriculture: Washington, DC, USA, 1976. Available online: <https://www.ams.usda.gov/grades-standards/apples-processing-grade-standards> (accessed on 3 February 2023).
3. *GB/T 10651-2008*; Fresh Apple. China Standard Press: Beijing, China, 2010. Available online: <https://openstd.samr.gov.cn/bzgk/gb/newGbInfo?hcno=39F27DD712D12CB6B8AA606228978445> (accessed on 3 February 2023).
4. Jiang, X.; Ma, J.; Xiao, G.; Shao, Z.; Guo, X. A review of multimodal image matching: Methods and applications. *Inf. Fusion* **2021**, *73*, 22–71. [[CrossRef](#)]
5. Lowe, D.G. Object Recognition from Local Scale-Invariant Features. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Corfu, Greece, 20–27 September 1999.

6. Bay, H.; Ess, A.; Tuytelaars, T.; van Gool, L. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.* **2008**, *110*, 346–359. [[CrossRef](#)]
7. Leutenegger, S.; Chli, M.; Siegwart, R.Y. BRISK: Binary Robust Invariant Scalable Keypoints. In Proceedings of the 2011 International Conference on Computer Vision, Washington, DC, USA, 6–13 November 2011.
8. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An Efficient Alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Washington, DC, USA, 6–13 November 2011.
9. Alcantarilla, P.F.; Bartoli, A.; Davison, A.J. KAZE Features. In Proceedings of the Computer Vision—ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, 7–13 October 2012.
10. Shima, H. *The Geometry of Hessian Structures*; World Scientific: Singapore, 2007.
11. Stanković, R.S.; Falkowski, B.J. The Haar wavelet transform: Its status and achievements. *Comput. Electr. Eng.* **2003**, *29*, 25–44. [[CrossRef](#)]
12. Yi, K.M.; Trulls, E.; Lepetit, V.; Fua, P. LIFT: Learned Invariant Feature Transform. In Proceedings of the Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016.
13. DeTone, D.; Malisiewicz, T.; Rabinovich, A. SuperPoint: Self-Supervised Interest Point Detection and Description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, New York, NY, USA, 18–23 June 2018.
14. Revaud, J.; Weinzaepfel, P.; De Souza, C.; Pion, N.; Csurka, G.; Cabon, Y.; Humenberger, M. R2D2: Repeatable and reliable detector and descriptor. *arXiv* **2019**, arXiv:1906.06195.
15. Chicco, D. Siamese neural networks: An overview. In *Artificial Neural Networks*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 73–94.
16. Tyszkiewicz, M.; Fua, P.; Trulls, E. DISK: Learning local features with policy gradient. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 14254–14265.
17. Qi, Y.; Yang, Z.; Sun, W.; Lou, M.; Lian, J.; Zhao, W.; Ma, Y. A comprehensive overview of image enhancement techniques. *Arch. Comput. Methods Eng.* **2021**, *29*, 583–6077. [[CrossRef](#)]
18. Zhong, J.; Yan, J.; Li, M.; Barriot, J.P. A deep learning-based local feature extraction method for improved image matching and surface reconstruction from Yutu-2 PCAM images on the Moon. *ISPRS J. Photogramm. Remote Sens.* **2023**, *206*, 16–29. [[CrossRef](#)]
19. Zhao, X.; Wu, X.; Miao, J.; Chen, W.; Chen, P.C.; Li, Z. Alike: Accurate and lightweight keypoint detection and descriptor extraction. *IEEE Trans. Multimed.* **2022**, *25*, 3101–3112. [[CrossRef](#)]
20. Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010.
21. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
22. Wu, H.; Gu, X. Max-Pooling Dropout for Regularization of Convolutional Neural Networks. In Proceedings of the Neural Information Processing: 22nd International Conference, Istanbul, Turkey, 9–12 November 2015.
23. Woo, S.; Park, J.; Lee, J.; Kweon, I. CBAM: Convolutional Block Attention Module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
24. Stergiou, A.; Poppe, R. Adapool: Exponential adaptive pooling for information-retaining downsampling. *IEEE Trans. Image Process.* **2022**, *32*, 251–266. [[CrossRef](#)] [[PubMed](#)]
25. Pinkus, A. Approximation theory of the MLP model in neural networks. *Acta Numer.* **1999**, *8*, 143–195. [[CrossRef](#)]
26. Graham, B. Fractional max-pooling. *arXiv* **2014**, arXiv:1412.6071.
27. O’Shea, K.; Nash, R. An introduction to convolutional neural networks. *arXiv* **2015**, arXiv:1511.08458s.
28. Yu, D.; Wang, H.; Chen, P.; Wei, Z. Mixed Pooling for Convolutional Neural Networks. In Proceedings of the Rough Sets and Knowledge Technology: 9th International Conference, Shanghai, China, 24–26 October 2014.
29. Jiang, N.; Wang, L. Quantum image scaling using nearest neighbor interpolation. *Quantum. Inf. Process.* **2015**, *14*, 1559–1571. [[CrossRef](#)]
30. Smith, P. Bilinear interpolation of digital images. *Ultramicroscopy* **1981**, *6*, 201–204. [[CrossRef](#)]
31. Gao, S.; Gruev, V. Bilinear and bicubic interpolation methods for division of focal plane polarimeters. *Opt. Express* **2011**, *19*, 26161–26173. [[CrossRef](#)] [[PubMed](#)]
32. Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A.; Bishop, R.; Rueckert, D.; Wang, Z. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
33. Crnjanski, J.; Krstić, M.; Totović, A.; Pleros, N.; Gvozdić, D. Adaptive sigmoid-like and PReLU activation functions for all-optical perceptron. *Opt. Lett.* **2021**, *46*, 2003–2006. [[CrossRef](#)] [[PubMed](#)]
34. Plötz, T.; Roth, S. Neural nearest neighbors networks. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 1095–1106.
35. Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 2292–2300.
36. Fan, S.; Liang, X.; Huang, W.; Zhang, V.J.; Pang, Q.; He, X.; Li, L.; Zhang, C. Real-time defects detection for apple sorting using NIR cameras with pruning-based YOLOV4 network. *Comput. Electron. Agric.* **2022**, *193*, 106715. [[CrossRef](#)]
37. Agarla, M.; Napoletano, P.; Schettini, R. Quasi Real-Time Apple Defect Segmentation Using Deep Learning. *Sensors* **2023**, *23*, 7893. [[CrossRef](#)] [[PubMed](#)]

38. Sarlin, P.; DeTone, D.; Malisiewicz, T.; Rabinovich, A. Superglue: Learning Feature Matching with Graph Neural Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020.
39. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. ScanNet: Richly-Annotated 3d Reconstructions of Indoor Scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
40. Li, Z.; Snavely, N. MegaDepth: Learning Single-View Depth Prediction from Internet Photos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.