A Vision-Guided Deep Learning Framework for Dexterous Robotic Grasping Using Gaussian Processes and Transformers †

KADALAGERE SAMPATH, Suhas, WANG, Ning, YANG, Chenguang, WU, Howard, LIU, Cunjia and PEARSON, Martin

applied sciences

MDPI

Article

A Vision-Guided Deep Learning Framework for Dexterous Robotic Grasping Using Gaussian Processes and Transformers †

Suhas Kadalagere Sampath [1,*], Ning Wang [2,*], Chenguang Yang [3], Howard Wu [4], Cunjia Liu [5] and Martin Pearson [1]

[1] Faculty of Engineering and Technology, University of the West of England, Bristol BS16 1QY, UK; martin.pearson@uwe.ac.uk
[2] School of Computing and Digital Technologies, Sheffield Hallam University, Sheffield S9 2AA, UK
[3] Department of Computer Science, University of Liverpool, Liverpool L3 5TR, UK; cyang@ieee.org
[4] Antobot Ltd., Cambridge CB1 1AH, UK; howard.wu@antobot.ai
[5] Department of Aeronautical and Automotive Engineering, Loughborough University, Leicestershire LE11 3TT, UK; c.liu5@lboro.ac.uk
* Correspondence: suhas.kadalageresampath@uwe.ac.uk (S.K.S.); ning.wang@shu.ac.uk (N.W.)
† This paper is an extended version of our paper published in Kadalagere Sampath, S.; Wang, N.; Wu, H.; Liu, C.; Jiang, J.; Yang, C. Integrating Vision and Learning-Based Method for Dexterous Grasping. In Proceedings of the 2024 29th International Conference on Automation and Computing (ICAC), Sunderland, UK, 28 August 2024; IEEE: Piscataway, NJ, USA, 2024.

Abstract: Robotic manipulation of objects with diverse shapes, sizes, and properties, especially deformable ones, remains a significant challenge in automation, necessitating human-like dexterity through the integration of perception, learning, and control. This study enhances a previous framework combining YOLOv8 for object detection and LSTM networks for adaptive grasping by introducing Gaussian Processes (GPs) for robust grasp predictions and Transformer models for efficient multi-modal sensory data integration. A Random Forest classifier also selects optimal grasp configurations based on object-specific features like geometry and stability. The proposed grasping framework achieved a 95.6% grasp success rate using Transformer-based force modulation, surpassing LSTM (91.3%) and GP (91.3%) models. Evaluation of a diverse dataset showed significant improvements in grasp force modulation, adaptability, and robustness for two- and three-finger grasps. However, limitations were observed in five-finger grasps for certain objects, and some classification failures occurred in the vision system. Overall, this combination of vision-based detection and advanced learning techniques offers a scalable solution for flexible robotic manipulation.

Keywords: dexterous robotic grasping; adaptive grasping; deep learning in robotics; transformer networks; Gaussian processes; vision-based force modulation

Academic Editor: Yutaka Ishibashi

Received: 23 December 2024
Revised: 23 February 2025
Accepted: 24 February 2025
Published: 28 February 2025

Citation: Kadalagere Sampath, S.; Wang, N.; Yang, C.; Wu, H.; Liu, C.; Pearson, M. A Vision-Guided Deep Learning Framework for Dexterous Robotic Grasping Using Gaussian Processes and Transformers. *Appl. Sci.* **2025**, *15*, 2615. https://doi.org/10.3390/app15052615

1. Introduction

The manipulation of objects with varying shapes, sizes, and fragilities presents significant challenges in robotic automation [1,2]. Unlike rigid industrial objects, delicate or deformable items are prone to damage from excessive force or poor grasping techniques [3,4]. Traditional grippers often lack adaptability and compliance, resulting in deformation or damage [5]. As a result, industries still depend on manual labour, which is costly and error prone. Achieving human-like dexterity and sensitivity in robotic systems is critical for automating such tasks [6,7].

Humans excel at object manipulation by integrating tactile feedback, vision, and cognitive strategies to adapt to grasp forces based on object properties [8]. Reproducing

these abilities in robotics requires dexterous hands, tactile and visual sensors, and adaptive control algorithms [9]. Advances in robotic vision, such as YOLOv8, enable real-time detection, segmentation, and pose estimation, supporting precise grasping [1]. However, tactile sensing and predictive models remain essential for addressing grasp force modulation and stability challenges [6,7] Gaussian Processes (GPs) have become a powerful tool in learning from demonstration (LfD), offering a non-parametric Bayesian framework to capture predictive accuracy and uncertainty [10–12]. GPs address variability in demonstrations by modelling precise trajectories in low-variability regions and compliance in high-variability ones [13], enabling smooth trajectory planning under uncertainty [14]. Recent advancements include integrating expert and novice demonstrations for robust policy learning, even with noisy data [15]. Frameworks like PILCO utilize GPs for system dynamics modelling and uncertainty-aware optimization, achieving data efficiency in reinforcement learning (RL) tasks [16]. Combining GPs with RL allows self-exploration to refine motion skills, enhancing robustness and safety through dynamic model updates based on uncertainty estimates [17,18]. In vision-based systems, GPs bridge perception and action for cost-effective, adaptable manipulation [10,11].

Transformers have recently addressed challenges in LfD and time-series analysis, particularly for generalization and long-term dependencies. For multi-task manipulation, parameter sharing and auxiliary objectives enhance generalization [19], while hierarchical Transformers tackle sparse rewards and long-horizon tasks through sub-goal planning [20]. Large-scale Task and Motion Planning (TAMP) datasets train visuomotor Transformer policies for extended manipulation tasks [21]. Techniques like action chunking and data augmentation enable robust single-demonstration cloning [22]. For multi-modal inputs, Transformers outperform LSTMs and CNNs in managing extended sequences for decision-making tasks [23]. They also improve dynamic prediction in industrial robots by capturing temporal and attribute dependencies more effectively [24]. In time-series forecasting, Transformers leverage convolutional self-attention and memory-efficient architectures to handle long sequences with reduced computational costs [25], as seen in the Informer model [26]. In robotic grasping, Transformers integrate tactile and visual data to develop robust strategies for deformable objects, surpassing traditional CNN+LSTM approaches [27].

In summary, Gaussian Processes and Transformers address key challenges in robotic manipulation and LfD. GPs enable uncertainty-aware, data-efficient learning, while Transformers excel in handling long-term dependencies and multi-modal inputs. Combining these tools can advance adaptive, robust robotic systems capable of human-like manipulation and decision-making. In addition, a Random Forest-based classifier enables the system to dynamically select optimal grasp configurations (e.g., two-, three-, or five-finger grasps) based on object-specific features like geometry and texture. The synergy of deterministic classification, probabilistic GPs, and deep learning frameworks creates a robust and adaptable system for handling diverse objects in unstructured environments.

Grasp force adaptation is crucial for handling objects of varying stiffness and fragility. Prior studies [28] explored compliance-based grasping, including variable stiffness control [29] and isotropic compliance [30] These methods rely on predefined mechanical properties, limiting adaptability in unstructured environments. While compliance-based mechanisms offer some flexibility, they require precise tuning and additional mechanical complexity. Instead of relying on predefined stiffness models, our approach directly utilises force sensors embedded in a dexterous robotic hand to measure applied forces and adjust grasping configurations dynamically.

Unlike vision-based tactile reconstruction techniques [31] which estimates force fields using indirect measurements, our system processes real-time force sensor data to enable adaptive grasping. Gaussian Processes (GPs) were initially explored for force estimation

but struggled to adjust dynamically in response to rapid force variations. Our experiments demonstrate that Transformers outperform GPs in force prediction, achieving higher success rates in dynamic grasping scenarios. This study provides new insights into integrating deep learning with sensor-driven grasping strategies to enhance force modulation in robotic grasping tasks.

Our prior work [32] introduced a system that combines YOLOv8 with Long Short-Term Memory (LSTM) networks for adaptive grasping. YOLOv8 efficiently localized objects and estimated their poses, offering precise 3D coordinates for the robotic hand to approach. LSTM networks, trained on teleoperated demonstrations, adjusted grasping forces dynamically based on object feedback. This framework demonstrated effectiveness in handling diverse objects under controlled environments and served as a baseline for evaluating Gaussian Process and Transformer models for grasp force modulation. The framework of the proposed approach is shown in Figure 1. The key objectives include:

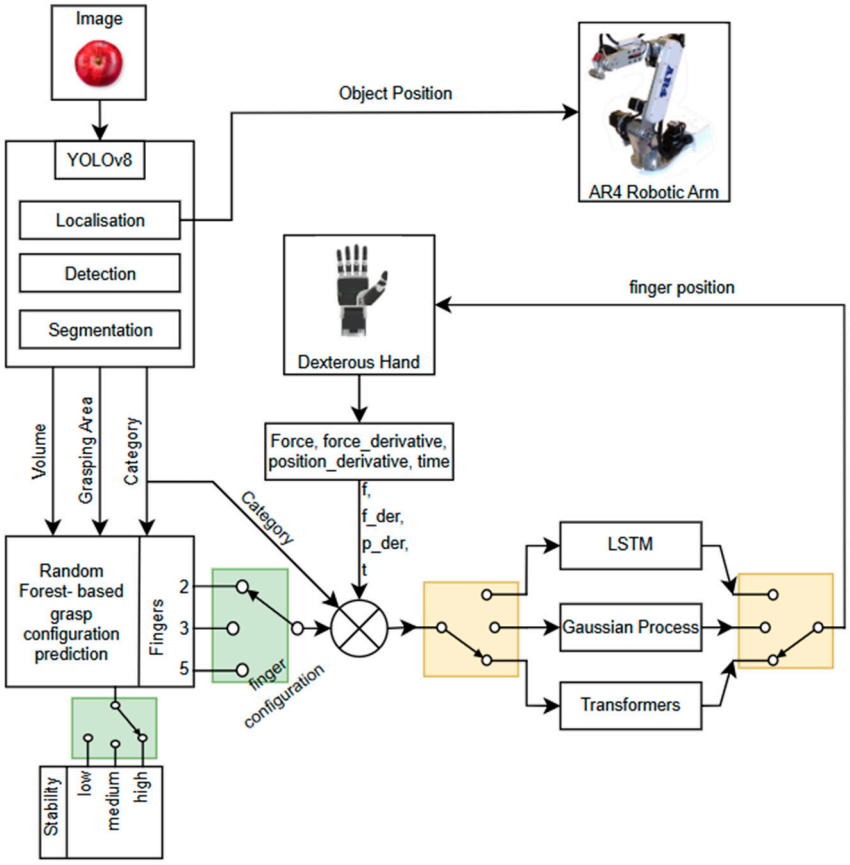


Figure 1. The framework of the proposed approach.

- Implementing YOLOv8 for robust real-time object detection and grasp configuration prediction;
- Comparing LSTM, Gaussian Process (GP), and Transformer models for grasp force prediction and analysing their efficiency;
- Evaluating the system's performance across multiple grasp configurations (two-, three-, and five-finger grasps) and object categories to assess stability and adaptability.

However, certain limitations persist in our framework. The five-finger grasp configuration exhibited instability when handling objects with complex shapes, particularly those with non-uniform stiffness, leading to inconsistent force distribution. While the vision model demonstrated strong performance in object detection, misclassification was observed for objects with visually similar characteristics (e.g., a tomato classified as a ball). Future

work will address these concerns by refining grasp configuration strategies and improving dataset diversity.

2. Materials and Methods

2.1. Hardware Setup

The experimental setup consists of an Annin Robotics AR4 [33], stepper motor driven six degrees of freedom (6DoF) robotic arm and a five-fingered tendon-driven 6DoF Beijing Inspire Robot Dexterous Hand [34,35], designed to replicate human-like grasping capabilities. Each finger of the robotic hand is equipped with force sensors that continuously measure contact forces, allowing for dynamic adjustments during manipulation. Additionally, a high-resolution Intel RealSense D435i RGB-D camera (Intel, Santa Clara, CA, USA) is mounted above the workspace to provide optimal visibility for object detection and segmentation, as shown in Figure 2. We employed an ArUco marker-based calibration method to map the depth camera with the robotic arm.

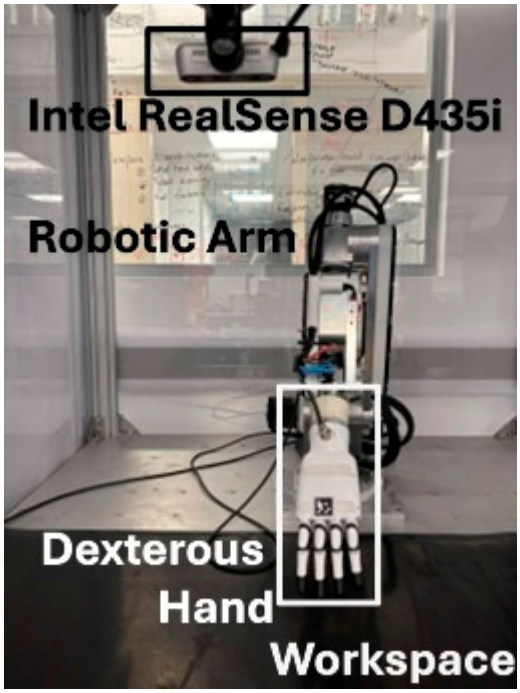
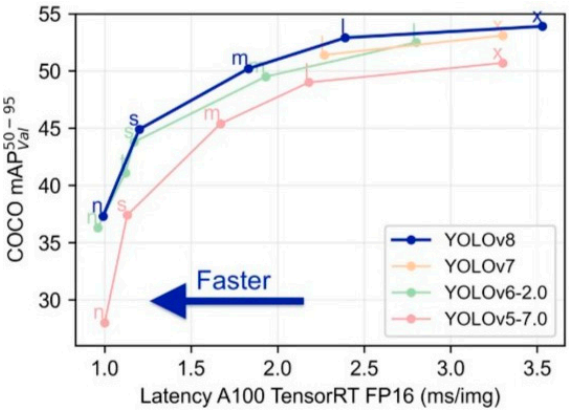


Figure 2. Experimental setup consisting of Intel RealSense D435i Depth Camera, AR4 Robotic Arm (Annin Robotics, Meridian, ID, USA) and Inspire Robot Dexterous Hand (INSPIRE-ROBOTS, Beijing, China).

2.2. Vision

2.2.1. Object Detection, Segmentation and Localization

Vision-based object detection and localization are accomplished using YOLO [36], a cutting-edge convolutional neural network model. YOLOv8 is renowned for its real-time processing capabilities and its effectiveness in detecting and classifying objects with a single forward pass. This significantly reduces computational demands and enhances speed, making it ideally suited for real-time applications. The architecture is built upon a deep convolutional neural network that is optimized for both speed and accuracy, featuring an advanced backbone for feature extraction and multiple scale predictors designed to detect objects of varying sizes. Below, Figure 3 illustrates a comparison of YOLO models trained on the COCO dataset.



(**a**)

Object Detection Performance Comparison (YOLOv8 vs YOLOv5)

Model Size	YOLOv5	YOLOv8	Difference
Nano	28	37.3	+33.21%
Small	37.4	44.9	+20.05%
Medium	45.4	50.2	+10.57%
Large	49	52.9	+7.96%
Xtra Large	50.7	53.9	+6.31%

*Image Size = 640

(**b**)

Instance Segmentation Performance Comparison (YOLOv8 vs YOLOv5)

Model Size	YOLOv5	YOLOv8	Difference
Nano	27.6	36.7	+32.97%
Small	37.6	44.6	+18.62%
Medium	45	49.9	+10.89%
Large	49	52.3	+6.73%
Xtra Large	50.7	53.4	+5.33%

*Image Size = 640

(**c**)

Figure 3. Comparison of YOLO models: (**a**) latency, (**b**) object detection performance (*Image Size = 640), (**c**) instance segmentation of the models (*Image Size = 640).

Based on Figure 3, the YOLOv8-n model is both smaller and faster compared to others, making it an ideal choice. To enhance the accuracy and estimate the object size, we opted for the YOLOv8-n-segmentation model. This model has been trained on a custom dataset that incorporates various lighting conditions to ensure robust detection across different environments. The network outputs bounding boxes, segmentation masks, and 3D coordinates for the objects, which are utilized to guide the robotic arm to the appropriate grasping position.

2.2.2. Dataset and Training

For our application, we expanded our custom dataset from a list of five [Apple, Ball, Box, Bottle, Strawberry] to nine object classes [Apple, Banana, Ball, Box, Bottle, Lemon, Orange, Strawberry, Tomato]. In our previous paper [32], we were not able to identify strawberries as the dataset did not include them from a certain distance; hence, we downloaded custom images from Google Open Image [37] that were manually labelled for detection and segmentation using [38], as shown in Figure 4.

(**a**) (**b**) (**c**)

Figure 4. Vision dataset with custom labelling: (**a**) apple, (**b**) orange, (**c**) tomato.

The YOLOv8 model was trained for 300 epochs with a batch size of 4 and an image size of 640 × 480. The optimizer was set to auto, with an initial learning rate of 0.01, a learning rate final factor of 0.01, and momentum set at 0.937. The overlap mask was enabled with a mask ratio of 4.0, and Non-Maximum Suppression (NMS) was set with an IoU threshold of 0.7 to refine object detection. The training was performed on a single GPU (device: 0) with 8 worker threads to optimize data loading. The model was fine-tuned on this dataset, achieving mAP@0.5—0.566 and F1—0.55. Figure 5a–c show instances per class, dataset, and training loss.

(**a**) (**b**) (**c**)

Figure 5. YOLOv8-n-seg model performance: (**a**) instances per class, (**b**) training dataset, (**c**) training loss.

During training, challenges were encountered when handling a higher number of instances per class, as the system became unstable and frequently crashed due to increased computational demands. To address this, batch sizes and data-loading strategies were optimized, ensuring a balance between memory efficiency and model performance. However, this limitation influenced the dataset composition, requiring adjustments to instance distributions across object categories. While the trained model demonstrated strong generalization, particularly in segmenting well-defined objects, misclassifications persisted for visually similar objects, such as the Bottle, Lemon, and Tomato.

2.2.3. Grasp Configuration

Robotic grasping necessitates a precise configuration to ensure secure and efficient manipulation of a variety of objects. This study introduces a machine learning-based approach designed to predict the optimal number of fingers for robotic grasping, thereby enhancing adaptability in handling objects with diverse geometries and stability requirements. The system leverages features extracted from object detection, such as object category, grasping area, shortest segment for grasping, and surface area. To quantify the ease of grasp, a stability parameter is categorized as low, medium, or high, along with a computed grasping ratio Equation (1).

$$\text{Ratio} = \frac{\text{Grasping Area}}{\text{Surface Area}} \tag{1}$$

Features are normalized, and categorical data are label-encoded before inputting into a pre-trained Random Forest model. The model predicts whether a two-, three-, or five-finger grasp is optimal based on the encoded features, achieving an accuracy of 78.9%. Integrated into the perception-action pipeline, the system predicts configurations in real-time, with

adjustable stability levels or default random assignments when stability data is unavailable. A detailed explanation can be found in Appendix A.3.

2.3. Dexterous Grasping

2.3.1. Data Collection

Achieving human-like dexterity and adaptability in grasping objects with varying properties is a challenging task. To address this challenge, we employ a learning-from-demonstration (LfD) approach, where the robotic system learns grasping strategies by observing human demonstrations [39]. The LfD data collection process involves an immersive teleoperation setup, where a Leap Motion Controller (LMC) [40] is used to capture finger movements [41,42]. The LMC can track up to 60 frames per second, but an average of 10 frames per second is considered to reduce noise.

Additionally, a Kalman filter was employed to further smoothen the tracking data for the dexterous hand. For the dexterous hand teleoperation, the hand was kept in the pre-grasp position. Later, the average direction vectors of the palm and individual fingers, $\text{avg}_{\text{finger}}$ (thumb, index, middle, ring, and pinky) were calculated over the past 10 frames (N), as shown in Equation (2). The bending angle $\left(\text{angle}_{\text{finger}}\right)$ for each finger was computed as the angle between the palm's average direction vector $\left(\text{avg}_{\text{palm}}\right)$ and the finger's average direction vector ($\text{avg}_{\text{finger}}$), as shown in Equation (3).

$$\text{avg}_{\text{finger}} = \frac{1}{N}\sum_{i=1}^{N} \text{finger}_i \cdot \text{direction} \tag{2}$$

$$\text{angle}_{\text{finger}} = \cos^{-1}\left(\frac{\text{avg}_{\text{palm}} \cdot \text{avg}_{\text{finger}}}{\left\|\text{avg}_{\text{palm}}\right\| \times \left\|\text{avg}_{\text{finger}}\right\|}\right) \times \frac{180}{\pi} \tag{3}$$

The bending angles of the dexterous hand are mapped to the corresponding finger angle limits, enabling it to replicate the hand gestures of a human operator. The system illustrated in Figure 6 incorporates a gesture-based condition to achieve diverse grasping configurations, specifically 2-finger, 3-finger, and 5-finger grasp, depending on the number of fingers extended by the operator's left hand. The dexterous hand is equipped with force sensors set to a predetermined threshold, and the Kalman filter is employed to smooth out minor fluctuations in the finger positions received from the LMC. At this stage, the robotic arm is positioned for pre-grasping, while the operator teleoperates the dexterous hand to grasp various objects using combinations of 2, 3, or 5 fingers, as depicted in Figure 7. For certain objects, only data for 2-finger and 3-finger grasps were collected due to their size, while others were grasped using 2, 3, and 5-fingers.

The teleoperation data, including finger positions, forces and their derivatives were normalized using Z-score normalization as shown in Equation (4):

$$z = \frac{x - \mu}{\sigma} \tag{4}$$

where μ and σ are the mean and standard deviation of the feature, respectively. Later, the data were used to train the LSTM, GP, and Transformer models responsible for adaptive grasping.

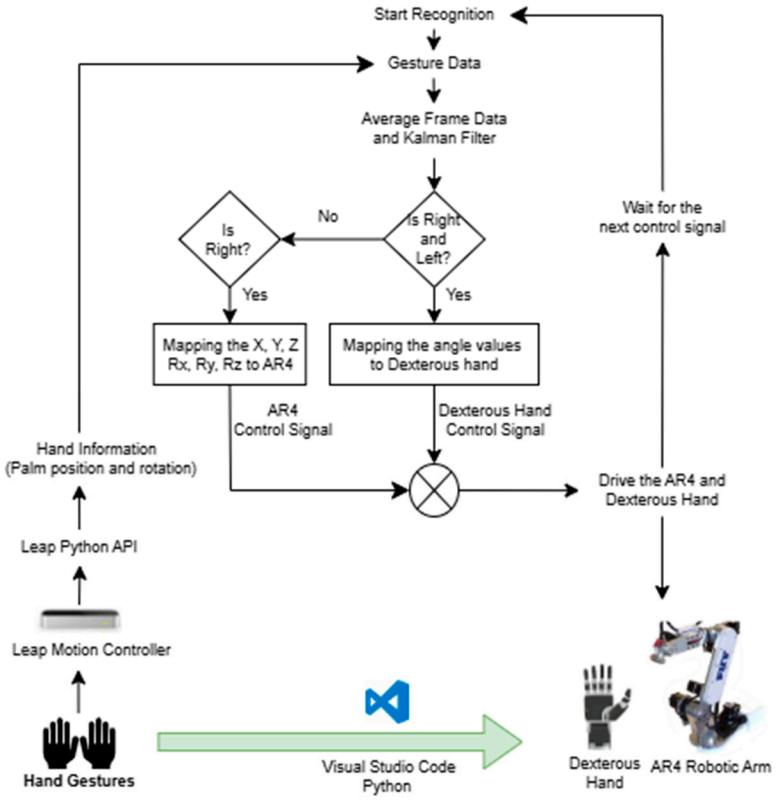
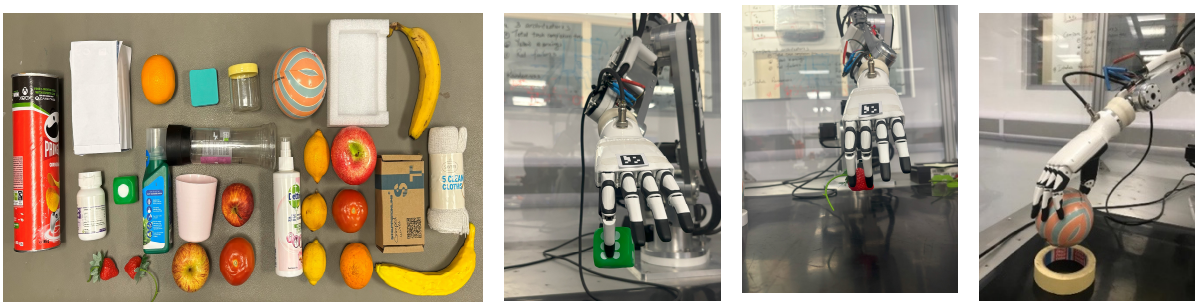


Figure 6. Control architecture for the teleoperation of dexterous hand and AR4 using LMC.



(**a**) (**b**) (**c**) (**d**)

Figure 7. Grasping dataset: (**a**) dataset, (**b**) 2-finger grasp, (**c**) 3-finger grasp, (**d**) 5-finger grasp.

2.3.2. Input Representation

The input to the LSTM, GP, and Transformer models consists of a sequence of feature vectors $X = [x_1, x_2, \ldots, x_T]$ where each $x_t \in \mathbb{R}^d$ at each timestep t contains forces ($f_t = [f_{t,P}, f_{t,R}, f_{t,M}, f_{t,I}, f_{t,T}]$), force derivatives $\left(\dot{f}_t = \left[\dot{f}_{t,P}, \dot{f}_{t,R}, \dot{f}_{t,M}, \dot{f}_{t.I}, \dot{f}_{t,T}\right]\right)$, position derivatives $\left(\dot{\theta}_t = \left[\dot{\theta}_{t,P}, \dot{\theta}_{t,R}, \dot{\theta}_{t,M}, \dot{\theta}_{t,I}, \dot{\theta}_{t,T}\right]\right)$ of five fingers, and a categorical identifier for the object being grasped, which was numerically encoded. The model predicted the finger positions $Y = [y_1, y_2, \ldots, y_T]$, where each $y_T \in \mathbb{R}^5$ representing the finger angles ($y_t = [\theta_{t,P}, \theta_{t,R}, \theta_{t,M}, \theta_{t,I}, \theta_{t,T}]$).

2.3.3. Long Short-Term Memory (LSTM)

The Long Short-Term Memory (LSTM) network served as the baseline model for predicting finger positions in human-like grasping tasks. Building on previous work [28], this study enhanced the approach by incorporating object categories as an additional parameter, alongside forces, force derivatives, and position derivatives, to improve generalization and

robustness. The architecture of the LSTM model features a single LSTM layer containing 64 hidden units to process input sequences, followed by a dense layer with 64 units. A final dense layer with five units is designed to predict finger angles at each timestep. The model was optimized using the Mean Squared Error (MSE) loss function, as shown in Equation (5):

$$\text{Loss} = \frac{1}{N}\sum_{i=1}^{N}\|\hat{y}_i - y_i\|^2 \tag{5}$$

where N, $\hat{y}_i$, and y_i are the number of samples, predicted, and ground truth finger angles, respectively. Trained using K-fold Cross-Validation (K = 5) with the Adam optimizer and a learning rate of 0.001, the model achieved an average MSE of 0.00085 across all folds. Training and validation loss trends are presented in Figure 8a.

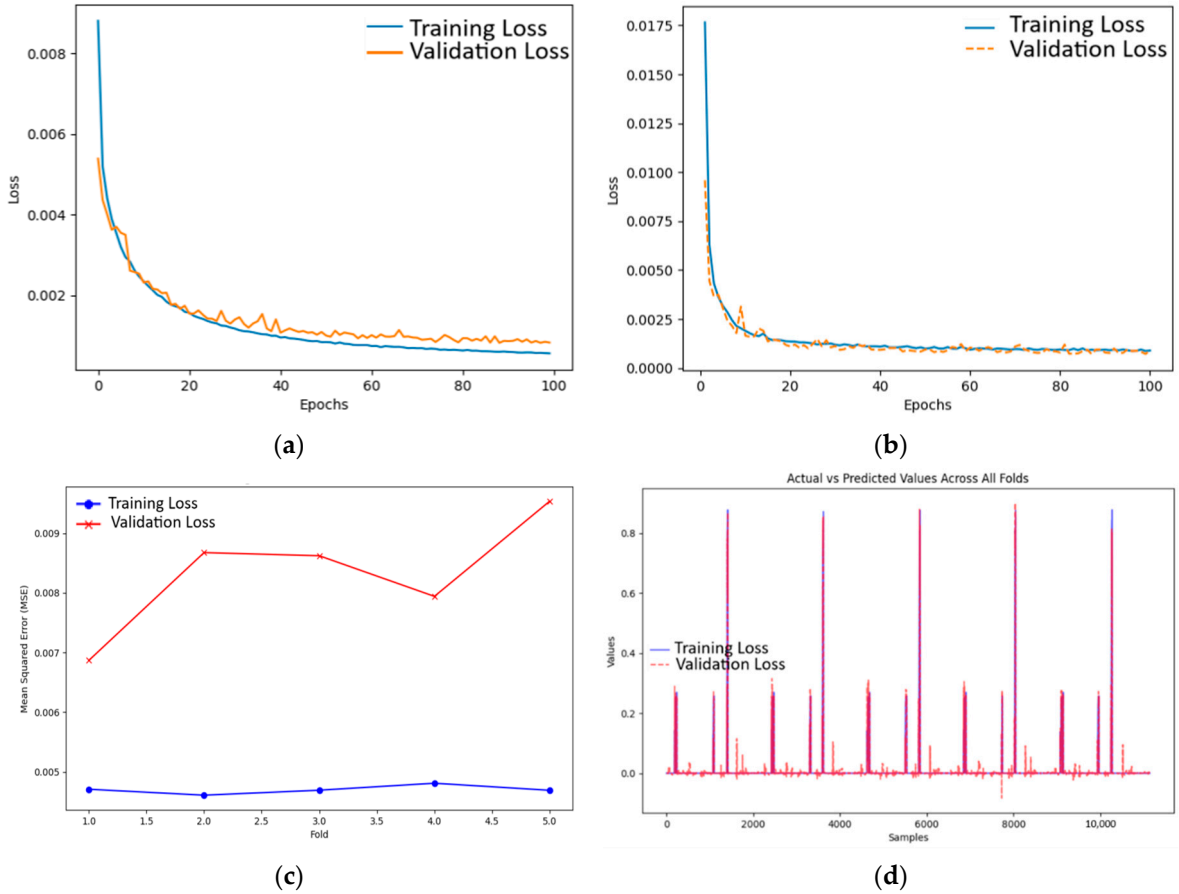


Figure 8. Training results for LSTM, GP and Transformer models: (**a**) training and validation plot for LSTM, (**b**) training and validation plot for GP, (**c**) actual vs prediction for GP, (**d**) training and validation plot for Transformer model.

2.3.4. Gaussian Process

The Gaussian Process (GP) models the relationship between inputs (X) and outputs (Y) as a distribution over functions as in Equation (6):

$$f(x) \sim GP(m(x), k(x, x')) \tag{6}$$

where $m(x)$ is the mean function and $k(x, x')$ is the kernel or covariance function. Assuming a zero mean $m(x) = 0$, the Radial Basis Function (RBF) kernel was employed due to its ability to capture both linear and non-linear relationships, as shown in Equation (7):

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{\|x - x'\|^2}{2l^2}\right) \quad (7)$$

where l is the length scale controlling the smoothness and σ_f^2 is the signal variance.

Given training data X and y, Equation (8) shows the GP posterior distribution at a new input x_*:

$$p(f_*|X, y, x_*) = \mathcal{N}\left(\mu_*, \sigma_*^2\right) \quad (8)$$

where:

$$\mu_* = k_*^T\left(K + \sigma_n^2 I\right)^{-1} y \quad (9)$$

$$\sigma_*^2 = k(x_*, x_*) - k_*^T\left(K + \sigma_n^2 I\right)^{-1} k_* \quad (10)$$

Here, K is the covariance matrix of the training data, k_* is the covariance vector between x_* and the training points, and σ_n^2 is the noise variance. The posterior variance σ_*^2 provides a measure of uncertainty in the predictions.

The hyperparameters l, σ_f, σ_n were optimized by maximizing the log marginal likelihood:

$$\log p(y|X) = -\frac{1}{2} y^T\left(K + \sigma_n^2 I\right)^{-1} y - \frac{1}{2}\log\left|K + \sigma_n^2 I\right| - \frac{N}{2}\log 2\pi \quad (11)$$

The model achieved an average Mean Squared Error (MSE) of 0.00833. Figure 8c,d illustrates the training data and prediction results.

2.3.5. Transformers

Transformer models were used to predict finger angles, effectively addressing the sequential nature of the problem. To integrate temporal sequence information, positional encoding p_t was added to the input feature vectors, providing a unique representation for each time step using sine and cosine functions. This positional information enhances the model's ability to discern sequential patterns.

The Transformer architecture consists of an input layer, multiple encoder layers, and an output layer. The input layer transforms the normalized input x_t into a higher-dimensional space h_t, where $h_t \in \mathbb{R}^{d_{model}}$ is the hidden representation. Each encoder layer features a multi-head self-attention mechanism which calculates attention scores between elements of the sequence followed by a feedforward network with ReLU activation. The output layer maps the final hidden states to predict the finger angles $\hat{y}_t$. The model was trained using the Mean Squared Error (MSE) loss function, similar to Equation (6). Training was performed using the K-Fold Cross Validation method (K = 5) for 100 epochs, ensuring robustness and generalization across datasets. The training and validation results are presented in Figure 8b.

3. Results

To evaluate the performance of the proposed system for dexterous robotic grasping, we compared the LSTM, GP, and Transformer models with the YOLOv8-n-seg model for classifying and grasping the list [Apple, Banana, Ball, Bottle, Box, Lemon, Orange, Strawberry, Tomato]. The experimental setup consisted of an Intel RealSense D435i RGB-D camera for the vision system, a 6-DoF Annin Robotics AR4 robotic arm, and a Beijing Inspire Robot Dexterous Hand with integrated force sensors. The computing hardware included an Intel Core i7 CPU, 16GB RAM, and an NVIDIA RTX 4070 8 GB GPU.

The proposed method demonstrated superior grasping performance compared to conventional approaches. The Transformer model achieved a 95.6% success rate, surpassing LSTM and GP (91.3%). Unlike rule-based methods, which rely on predefined force

thresholds, our approach dynamically adjusts grasp forces using real-time sensor feedback, ensuring adaptability across various object geometries. While Gaussian Processes effectively quantified uncertainty, they struggled with rapid force variations, making them less suitable for highly dynamic interactions. The Transformer model excelled in force prediction, enabling smoother modulation and more precise grasping. The grasp configuration system further enhanced stability in two- and three-finger grasps, particularly for delicate objects like strawberries and lemons. These results highlight the effectiveness of deep learning-driven grasp adaptation, creating a robust and flexible strategy for robotic manipulation.

3.1. Vision

The YOLOv8-n-seg model was trained on a custom dataset consisting of nine classes [Apple, Ball, Banana, Bottle, Box, Lemon, Orange, Strawberry, Tomato]. During training, the model achieved an overall Mean Average Precision (mAP@0.5) of 0.566 and mAP for individual categories, as shown in Table 1 and the confusion matrix in Appendix A.2. The objects listed in Table 1 were carefully selected to evaluate the grasping model's adaptability across diverse geometries and material properties. Spherical objects like apples and balls test rotational stability, while elongated objects like bananas challenge force modulation due to their deformability. Rigid objects such as bottles and boxes assess grasp robustness under different stability constraints, whereas small objects like strawberries and lemons test the precision of finger grasp configurations. This selection ensures a comprehensive evaluation of the grasping model's effectiveness in real-world scenarios.

Table 1. Vision model training results and real-time detections for two-, three-, and five-finger grasp configurations. (✓—Successfully detected and classified; ✗—Unsuccessful (misclassified)).

Object Class	mAP@0.5	Detection Based on Grasp Configurations		
		2 Fingers	3 Fingers	5 Fingers
Apple	0.612	✓	✓	✓
Ball	0.476	✓	✓	✓
Banana	0.221	✓	✓	✓
Bottle	0.371	✗	✓	✗
Box	0.472	✓	✓	✓
Lemon	0.635	✗	✓	-
Orange	0.643	✓	✓	-
Strawberry	0.948	✓	✓	-
Tomato	0.716	✗	✗	-
All Classes	**0.566**			

Figure 9 shows real-time detection, classification, and segmentation results for nine classes using a three-finger grasp configuration. The model effectively classified objects and predicted appropriate grasp configurations based on their surface characteristics. As detailed in Table 1, the vision system achieved efficiencies of 66.66%, 88.88%, and 80% for two, three, and five-finger configurations, respectively. Challenges arose in classifying the Bottle, Lemon, and Tomato in the two-finger configuration, and with the Tomato and Bottle in the three and five-finger configurations. No five-finger configurations were generated for the small Lemon, Orange, Strawberry, and Tomato. Misclassification issues occurred due to similarities between classes, such as confusing a Lemon for an Orange and a Tomato for a Ball. These issues could be resolved with more data and careful labelling. Detailed object detection pictures can be found in Appendix A.2.

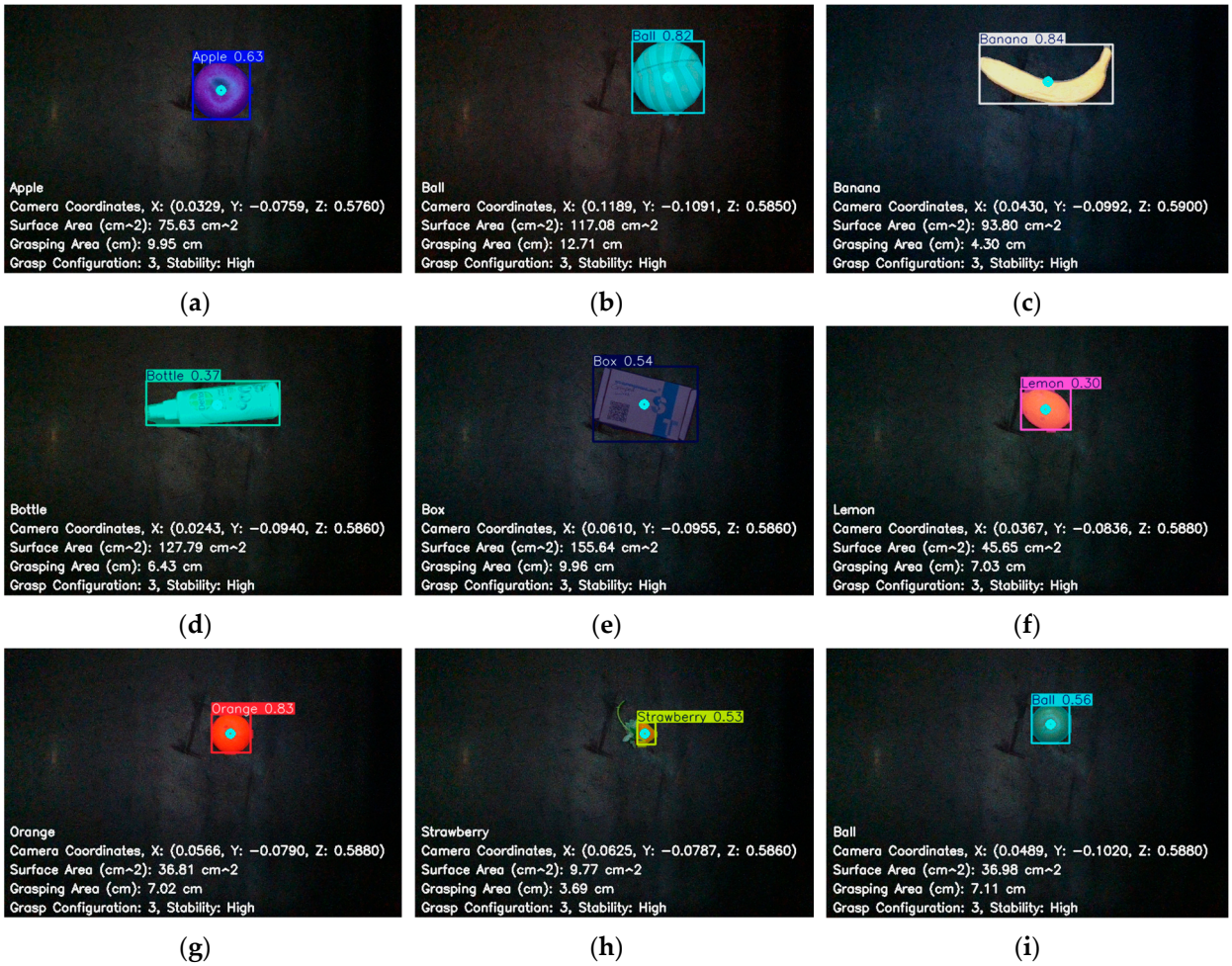


Figure 9. Real-time detection and segmentation for three-finger grasp configuration: (**a**) Apple, (**b**) Ball, (**c**) Banana, (**d**) Bottle, (**e**) Box, (**f**) Lemon, (**g**) Orange, (**h**) Strawberry, (**i**) Tomato misclassified as Ball.

3.2. Dexterous Grasping

The grasping configuration results, as illustrated in Table 2, showcase the performance of LSTM, Gaussian Process (GP), and Transformer models across various objects using two-, three-, and five-finger grasps. All configurations demonstrated adaptability for stable objects like the Apple, Ball, and Box, which were successfully grasped by each model. Figure 10 shows the three-finger grasping of a Ball, Banana, Orange, and Tomato, highlighting the models' ability to adjust grasp configurations dynamically.

Figure 10. Real-time grasping results for three-finger grasp configuration: (**a**) Ball, (**b**) Banana, (**c**) Orange, (**d**) Tomato.

Table 2. Average force on all fingers based on grasp configuration for LSTM, GP, and Transformer models for the detected objects. (GREEN—successful grasps; RED—Unsuccessful grasps).

Object Class	Average Force (N) on All Fingers Based on Grasp Configuration								
	LSTM			GP			Transformers		
	2 Fingers	3 Fingers	5 Fingers	2 Fingers	3 Fingers	5 Fingers	2 Fingers	3 Fingers	5 Fingers
Apple	4.1	3.1	2.1	1.6	2.9	1.9	3.2	2.4	2.2
Ball	2.7	2.4	1.75	1.3	1.9	1.63	2.4	1.8	1.9
Banana	3.6	2.1	1.6	1.8	2.2	1.7	3.6	2.9	2.1
Bottle	2.9	2.6	1.9	1.7	2.7	1.2	3	2.8	2.3
Box	3.8	3.1	2.3	2.9	1.9	2	2.4	2.7	2.1
Lemon	1.7	1.8	-	2.3	1.7	-	2.8	1.9	-
Orange	2.7	2.1	-	1.7	1.7	-	2	2.2	-
Strawberry	3.5	3.3	-	1.6	1.4	-	3.6	2.9	-
Tomato	2.7	1.6	-	1.7	1.7	-	3.1	2.7	-

However, challenges arose with objects such as the Banana and Bottle when using the five-finger grasp, resulting in instability and suboptimal performance across all models. The LSTM model exhibited force fluctuations, struggling to maintain consistent force application, particularly when grasping objects with irregular shapes. The GP model, although effective in providing uncertainty estimates, tended to under-predict the required grasp force, resulting in occasional slippage. The Transformer model outperformed both LSTM and GP by dynamically adjusting grasp forces based on contextual relationships rather than relying solely on past states. This adaptability was evident in the three-finger grasp configuration, where the Transformer model maintained a 95.6% efficiency rate compared to 91.3% for LSTM and GP models.

For more delicate items like Lemon, Orange, Strawberry, and Tomato, the models consistently opted for two- or three-finger configurations instead of five-finger grasps, as these proved to be more stable and effective. The grasp configuration system dynamically predicted these results based on object-specific characteristics, including geometry, grasping area, and stability levels, thereby enhancing the precision of the outcomes. Despite some limitations, the combination of grasp prediction with the advanced models ensured a balanced approach, achieving stable grasps for a wide variety of objects.

3.2.1. LSTM

The Long Short-Term Memory (LSTM) model served as a baseline for predicting adaptive finger positions during grasping tasks. With an average Mean Squared Error (MSE) of 0.00085, the LSTM exhibited moderate accuracy in predicting grasp forces. However, it demonstrated instability when dealing with certain objects. For instance, while the grasp forces for the Ball (2.7 N in a two-finger grasp) and Bottle (2.9 N in a two-finger grasp) were reasonably precise, the model struggled to maintain consistency for items that required more complex grasp configurations, particularly the Banana and Bottle during five-finger grasps, as shown in Table 2. Figure 11 illustrates the force profiles for the three-finger grasp configuration of the Ball, Banana, Orange, and Tomato classes.

In three-finger grasp configurations, the average forces recorded were 2.4 N, 2.1 N, 2.1 N, and 1.6 N for the Ball, Banana, Orange, and Tomato, respectively. Overall, the LSTM model achieved an efficiency of 91.3% for grasping tasks. Although the vision model misclassified the objects, it successfully detected and facilitated grasping. This limitation can be attributed to LSTM's challenges in effectively generalizing sequential dependencies for such tasks. Despite these issues, LSTM provided a robust baseline for comparison, demonstrating moderate accuracy and computational efficiency.

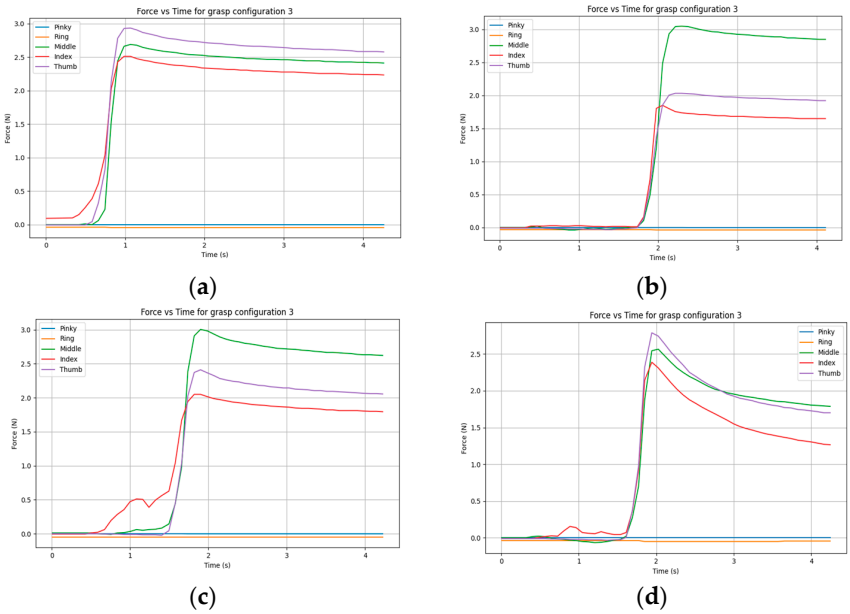


Figure 11. Force profile for three-finger grasp configuration for the LSTM model: (**a**) Ball, (**b**) Banana, (**c**) Orange, (**d**) Tomato.

3.2.2. Gaussian Process

The Gaussian Process (GP) model demonstrated reliable predictions and offered the crucial benefit of uncertainty quantification, which is essential for handling fragile and deformable objects. The model achieved an average Mean Squared Error (MSE) of 0.00833, slightly higher than that of LSTM, yet it provided more conservative and stable predictions. For delicate items such as Strawberries (1.6 N for a two-finger grasp) and Lemon (1.7 N for a two-finger grasp), the GP model effectively minimized the risk of damage by applying lower forces. Additionally, it achieved average forces of 1.9 N, 2.2 N, 1.7 N, and 1.7 N for three-finger configurations for the Ball, Banana, Orange, and Tomato, respectively, as shown in Figure 12.

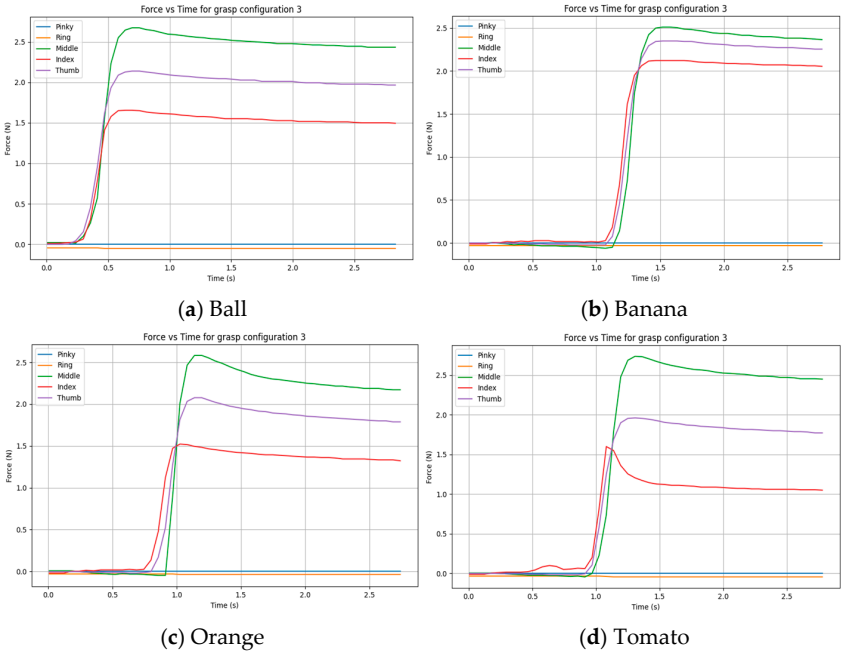
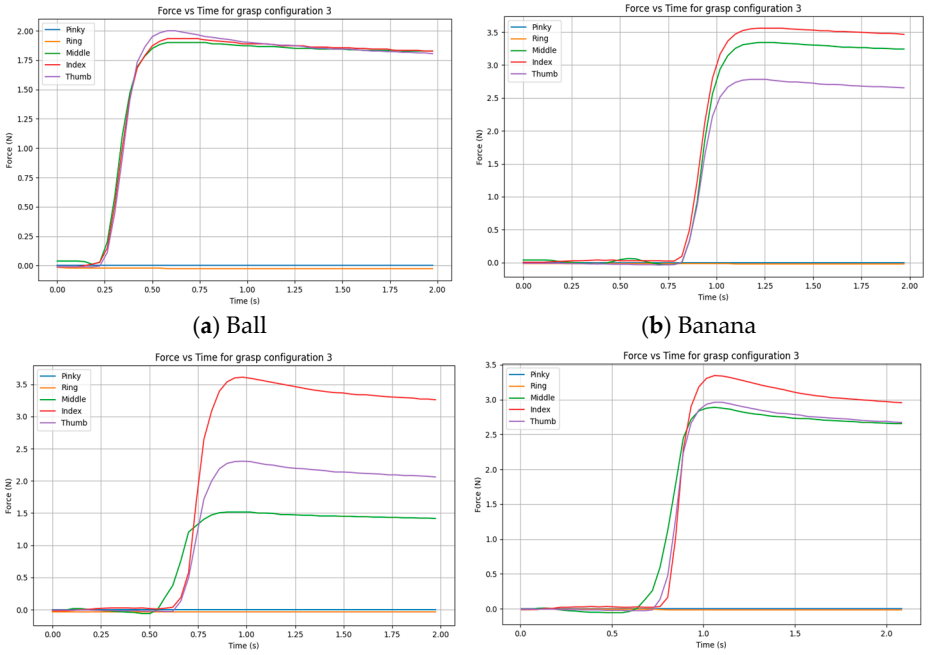


Figure 12. Force profile for three-finger grasp configuration for the GP model: (**a**) Ball, (**b**) Banana, (**c**) Orange, (**d**) Tomato.

The model maintained an overall efficiency of 91.3%, comparable to that of the LSTM model. This conservative methodology minimizes damage risks, making it particularly valuable in safety-critical applications. For heavier objects like boxes, the GP exhibited reasonable force estimations but occasionally under-predicted optimal grasp forces, reflecting its cautious nature. Overall, GP's probabilistic approach proved advantageous for safety-critical scenarios, although it faced challenges in adapting to varying grasp configurations.

3.2.3. Transformers

The Transformer model exhibited superior performance in predicting grasp forces and finger positions across a variety of objects, effectively overcoming the limitations seen in LSTM and GP models. Unlike sequential architectures, which process data step by step, Transformers leverage a self-attention mechanism to analyze the entire sequence at once. This ability enables them to efficiently capture long-term dependencies and contextual relationships, leading to more accurate and adaptive force modulation. In the experiments, the Transformer model demonstrated improved force modulation across different grasp configurations. For instance, the forces applied to an apple were recorded at 3.2 N for a two-finger grasp and 2.4 N for a three-finger grasp as shown in Table 2. The model achieved an overall efficiency of 95.6%. In the three-finger grasp configuration, the average forces recorded were 1.8 N, 2.9 N, 2.2 N, and 2.7 N, for the Ball, Banana, Orange, and Tomato, respectively, as shown in Figure 13.



(**a**) Ball (**b**) Banana

(**c**) Orange (**d**) Tomato

Figure 13. Force profile for three-finger grasp configuration for the Transformer model: (**a**) Ball, (**b**) Banana, (**c**) Orange, (**d**) Tomato.

A key advantage of the Transformer model is its ability to process multiple dependencies within an input sequence simultaneously, allowing it to refine force predictions more effectively than LSTM, which relies on sequential memory. By considering the broader context of the grasping scenario, the Transformer model was able to adjust forces dynamically, making it more adaptable to varying object shapes and textures. However, similar to LSTM and GP models, the Transformer encountered stability issues with the ball during five-finger grasps, as the grasp configuration struggled to maintain a constant force once grasped. These findings indicate that, while the Transformer excels in force modulation

and generalization, certain object geometries and grasp configurations will require further refinement. These findings highlight the potential of Transformers in robotic grasping applications, demonstrating their ability to enhance force adaptation and grasp precision across a wide range of objects while maintaining computational efficiency and robustness.

3.3. Failure Analysis

The failure cases for the five-finger grasp show the difficulties with complex shapes and how forces are applied. The system worked well with two- and three-finger grips but struggled with objects like the Banana, Bottle, and Ball, as seen in Table 2. These challenges primarily stem from model limitations and hardware constraints, affecting force distribution and grasp consistency.

Each model exhibited specific weaknesses when dealing with certain object geometries. The LSTM model struggled with sudden force spikes, particularly in the Index and Middle fingers when grasping the Banana. Figure 14a shows the corresponding force profile. As the Banana is long and prone to bending, stable grasping required continuous fine-tuned force adjustments. However, the sequential nature of LSTM limited its ability to dynamically regulate forces in real-time, resulting in frequent overcompensation. The GP model, while effective in providing uncertainty-aware predictions, tended to underestimate required forces, leading to instabilities when grasping rigid objects such, as the Bottle. Repeated force fluctuations were observed in the Middle and Ring fingers due to the model's conservative force estimates, which led to unintended shifts in grasp stability. Figure 14b shows the corresponding force profile. The Transformer model demonstrated superior adaptability but encountered difficulties with the Ball, where its symmetrical surface required a precise balance of forces across all fingers. Figure 14c shows the corresponding force profile. The model occasionally struggled to maintain uniform force distribution, leading to slow adjustments and inconsistent grasp stability.

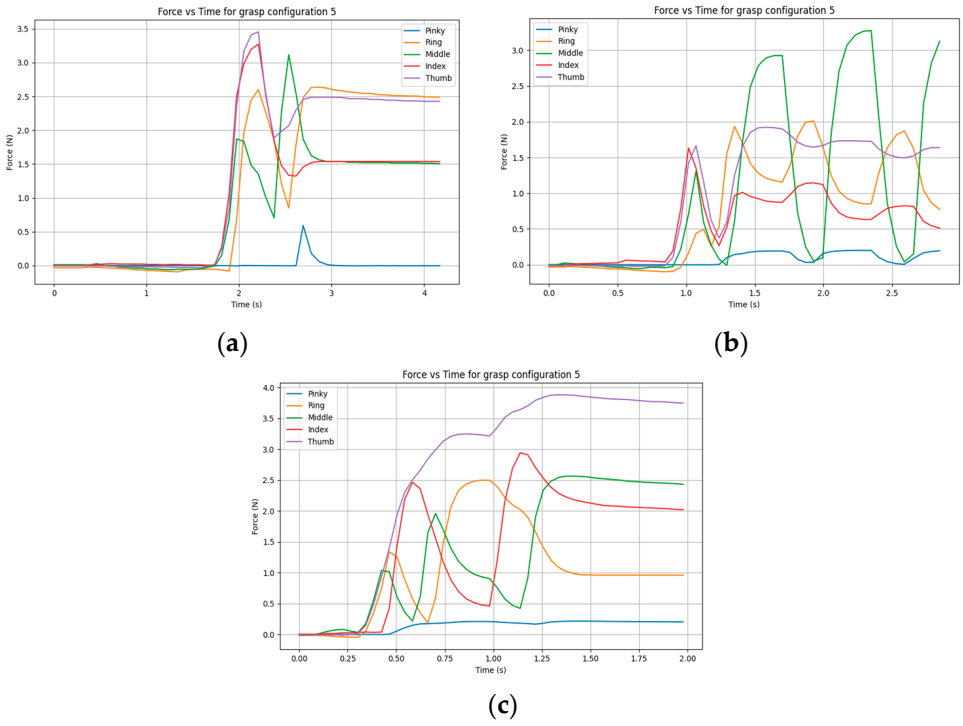


Figure 14. Force profile for the failure cases of five-finger grasp configuration: (**a**) Banana with LSTM, (**b**) Bottle with GP, (**c**) Ball with Transformer model.

Beyond model-specific limitations, hardware constraints also contributed to grasp failures. The robotic hand used in this study relies on embedded force sensors for force estimation but lacks distributed tactile sensing across the fingers. This limitation was particularly evident in objects such as the Tomato and Strawberry, where minor force inconsistencies caused unexpected shifts in object position. The absence of fine-grained tactile sensing restricted real-time corrective adjustments, impacting grasp stability for deformable and delicate objects. Additionally, actuation delays in the robotic fingers introduced minor inconsistencies in force application. Although the Transformer model effectively adjusted grasp forces, the hardware's response lag influenced grasp precision, especially in five-finger configurations where force needed to be dynamically redistributed.

Addressing these challenges requires improvements in both model robustness and hardware capabilities. Enhancing the Transformer model with stability-aware grasp predictions could mitigate force inconsistencies for symmetrical objects, such as the Ball. Incorporating additional tactile sensors would improve real-time force adaptation, particularly for fragile and deformable objects, where subtle force variations significantly affect grasp stability. Moreover, optimizing grasp configuration predictions to incorporate real-time adjustments based on force feedback could further enhance grasp performance. These findings emphasize the importance of integrating advanced force prediction models with improved sensing and actuation mechanisms to achieve stable and adaptive robotic grasping.

4. Discussion and Future Work

This study presents a vision-guided deep learning framework for dexterous robotic grasping that integrates YOLOv8-n-seg for object detection, Gaussian Processes (GP) for uncertainty-aware force predictions, and Transformer models for sequential data processing. The grasp configuration prediction system enhances adaptability by selecting optimal finger configurations based on object-specific characteristics.

While the framework demonstrates improved efficiency, several challenges remain. The vision model exhibited misclassification errors for objects with subtle textures and irregular shapes, particularly for the Bottle in two- and five-finger configurations, the Lemon in two-finger configurations, and the Tomato in two- and three-finger configurations. Additionally, instability persisted in five-finger grasps for the Banana, Bottle, and Ball due to their irregular geometries and complex force modulation demands. These factors resulted in inconsistent force distribution across the fingers, leading to grasp failures in certain scenarios.

Future research will focus on enhancing the vision system through improved object classification techniques and real-time tracking to mitigate misclassification issues. Five-finger grasp control strategies will be refined by incorporating advanced grasp stability metrics and force redistribution techniques. Further optimizations in stability-aware predictions using the Transformer model will be explored to improve force consistency, particularly for objects requiring fine-tuned adjustments. Additionally, integrating tactile sensing capabilities will provide real-time feedback on grasp stability, enabling precise and adaptive force modulation.

5. Conclusions

The proposed framework significantly enhances dexterous robotic grasping by integrating advanced vision and deep learning techniques. The results indicate a substantial improvement over the previous LSTM-based system, which achieved only 60% efficiency across five object classes. In this study, with nine object classes [Apple, Ball, Banana, Bottle,

Box, Lemon, Orange, Strawberry, Tomato], the LSTM and GP models achieved an efficiency of 91.3%, while the Transformer model outperformed with an efficiency of 95.6%.

The findings highlight the potential of combining probabilistic modelling, self-attention mechanisms, and multi-modal sensing to improve grasp adaptability and precision. By addressing current limitations, future advancements will contribute to enhancing robotic dexterity in unstructured environments, facilitating more reliable automation in complex grasping tasks. The proposed framework has practical applications in industrial assembly, logistics, and agricultural harvesting. By improving grasp stability and adaptability, this approach reduces reliance on pre-programmed grasp strategies, increasing operational efficiency and flexibility. In agriculture, it can be applied to delicate produce handling, minimizing damage and optimizing sorting efficiency.

Author Contributions: Conceptualization, methodology, software, writing and editing, S.K.S.; writing—review, supervision N.W., M.P., C.Y., H.W. and C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: Author Howard Wu was employed by the company Antobot Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A

Appendix A.1. Coordinate Mapping

During operation, the RGB-D camera (Intel RealSense D435i) captures RGB images and depth data of the scene. The YOLOv8-n-segmentation network processes the RGB images and outputs the bounding boxes, and segmentation masks of the detected objects, along with their class labels, confidence scores, surface area and grasping area. The depth data is then used to estimate the 3D coordinates of the object's centroid within the camera frame. A calibrated transformation matrix then converts these coordinates from the camera frame to the robotic arm's frame, ensuring precise grasping. Initially, chessboard calibration was used to obtain (X, Y, Z) position data, but it lacked rotation information, leading to grasping inaccuracies. To improve mapping, an ArUco-based calibration method was implemented, incorporating both position and orientation for better alignment between the camera and the AR4 robotic arm.

The coordinate transformation follows a two-step process. First, the centroid of the detected bounding box is computed as:

$$C_x = \frac{x_{min} + x_{max}}{2},\ C_y = \frac{y_{min} + y_{max}}{2}$$

The corresponding depth value $D\left(C_x,\ C_y\right)$ is retrieved from the depth map. Using the camera's intrinsic parameters, the 2D image coordinates are projected into 3D camera frame coordinates as follows:

$$X_c = \frac{(C_x - c_x)\cdot D\left(C_x,\ C_y\right)}{f_x},\ \ Y_c = \frac{\left(C_y - c_y\right)\cdot D\left(C_x,\ C_y\right)}{f_y},\ \ Z_c = D\left(C_x,\ C_y\right),$$

where $(c_x,\ c_y)$ is the principal point, $(f_x,\ f_y)$ are the focal lengths, and $D\ (C_x,\ C_y)$ is the depth at the centroid.

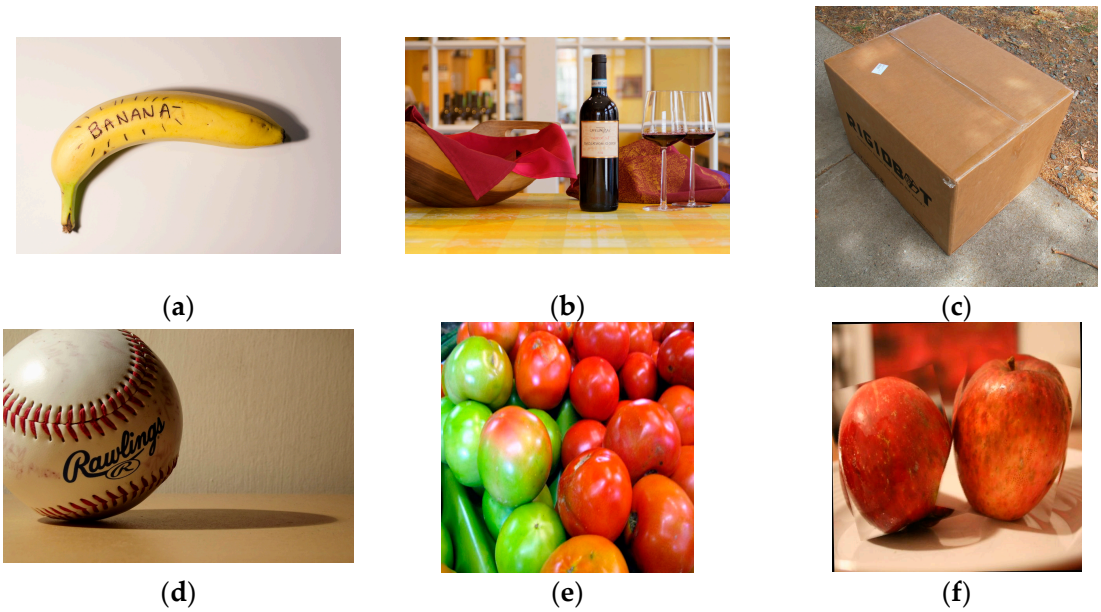
The second step involves transforming the extracted camera frame coordinates into the robot's coordinate system using the extrinsic transformation matrix $T_{camera \rightarrow Robot}$ obtained through ArUco-based calibration:

$$\begin{bmatrix} X_r \\ Y_r \\ Z_r \\ 1 \end{bmatrix} = T_{camera \rightarrow Robot} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

where $(X_r,\ Y_r, Z_r)$ represents the final object position in the robotic coordinate frame. This transformation ensures accurate grasp execution by compensating for the camera's position and orientation relative to the robot. ArUco-based calibration significantly improved the accuracy by including rotation corrections, making object localization more reliable compared to previous methods.

Appendix A.2. Vision

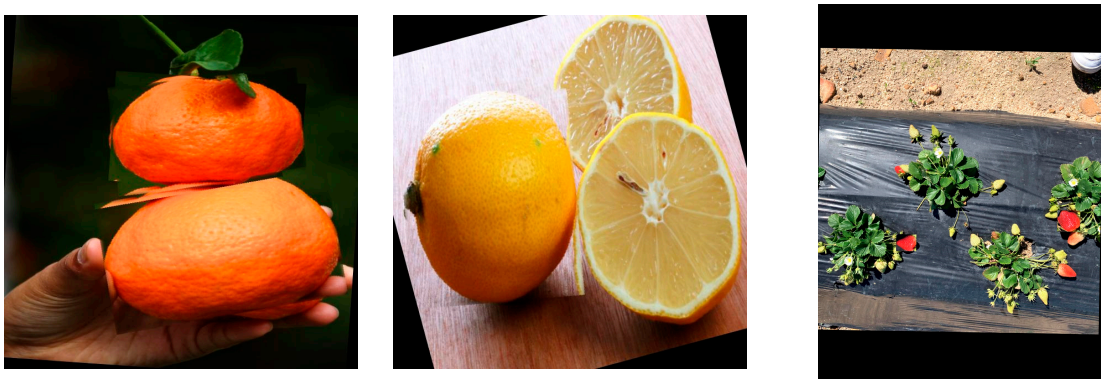
The YOLOv8-n-seg model by Ultralytics was trained using a dataset processed in Python 3.9, comprising 5551 images for training and 781 images for validation and testing. The dataset was collected from Google Open Images V7, ensuring diverse object representation. Images were annotated using the Roboflow tool to generate segmentation masks and bounding boxes. While the original images had varying resolutions, they were resized to 640 pixels before being input into the model. A standard 80–20 split was applied to maintain a fair distribution between training and validation. Figure A1 shows the sample dataset.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure A1. *Cont.*



(**g**) (**h**) (**i**)

Figure A1. Vision dataset: (**a**) Banana, (**b**) Bottle, (**c**) Box, (**d**) Ball, (**e**) Tomato, (**f**) Apple, (**g**) Orange, (**h**) Lemon, (**i**) Strawberry.

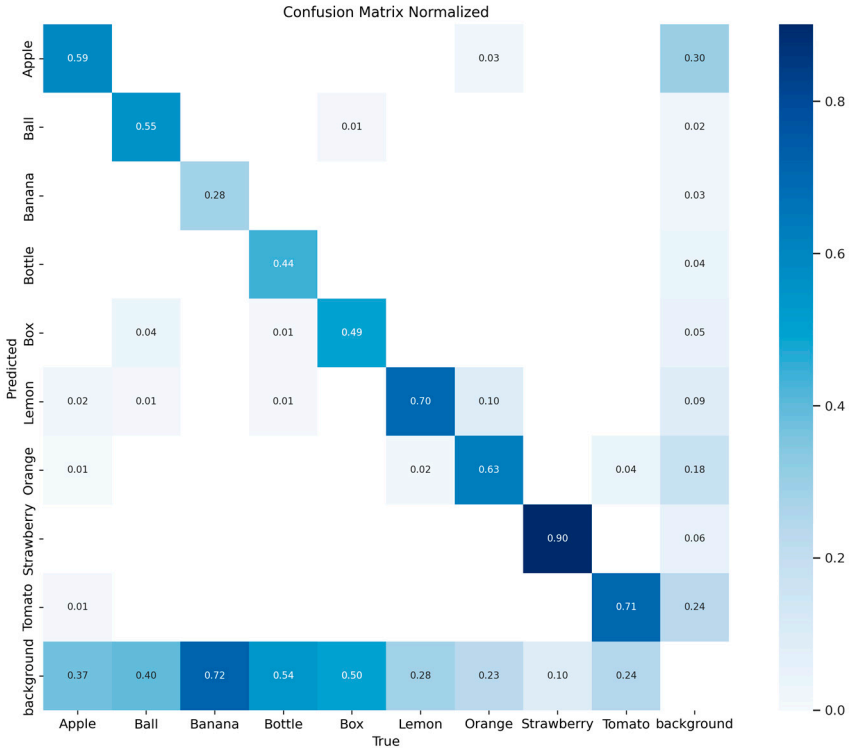


Figure A2. Confusion matrix for YOLOv8-n-seg model.

(**a**) (**b**) (**c**)

Figure A3. *Cont.*

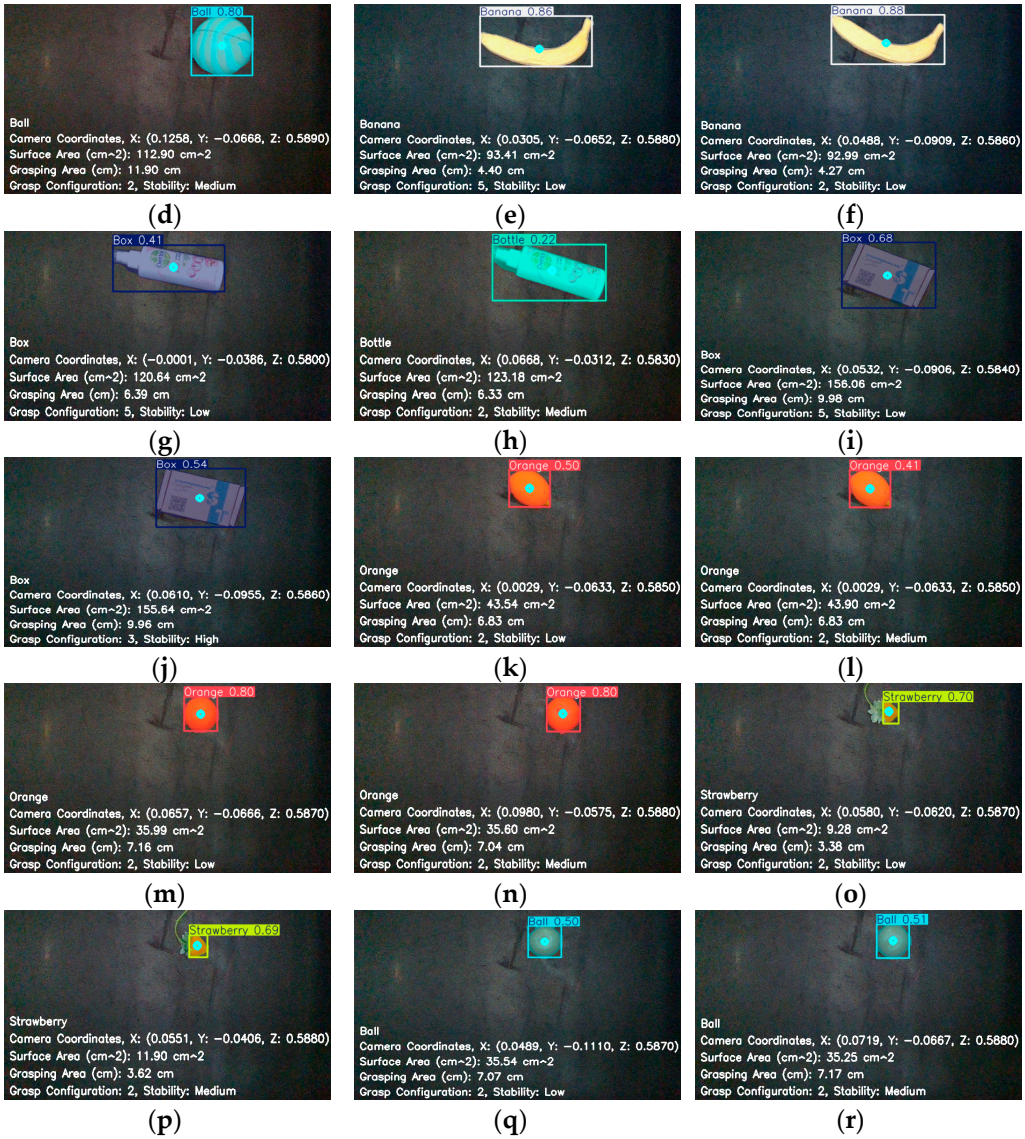


Figure A3. Real-time detection and segmentation for two- and three-finger grasp configurations: (**a**) Apple (5 finger), (**b**) Apple (2 finger), (**c**) Ball (5 finger), (**d**) Ball (2 finger), (**e**) Banana (5 finger), (**f**) Banana (2 finger), (**g**) Bottle misclassified as Box (5 finger), (**h**) Bottle (2 finger), (**i**) Box (5 finger), (**j**) Box (2 finger), (**k**) Lemon misclassified as Orange (2 finger—low stability), (**l**) Lemon misclassified as Orange (2 finger—medium stability), (**m**) Orange (2 finger—low stability), (**n**) Orange (2 finger—medium stability), (**o**) Strawberry (2 finger—low stability), (**p**) Strawberry (2 finger—medium stability), (**q**) Tomato misclassified as Ball (2 finger—low stability), (**r**) Tomato misclassified as Ball (2 finger—medium stability).

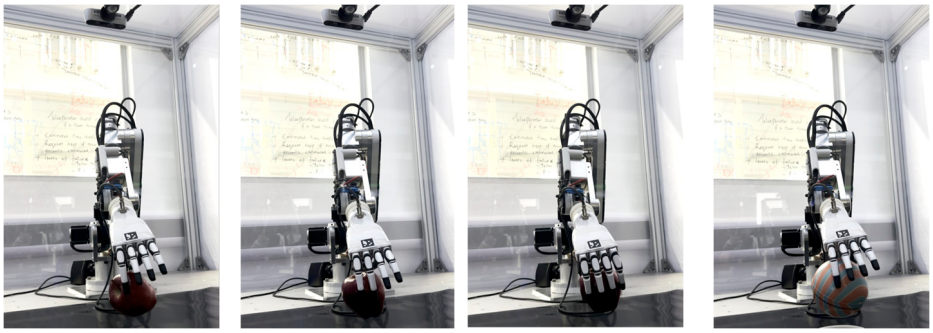


Figure A4. *Cont.*

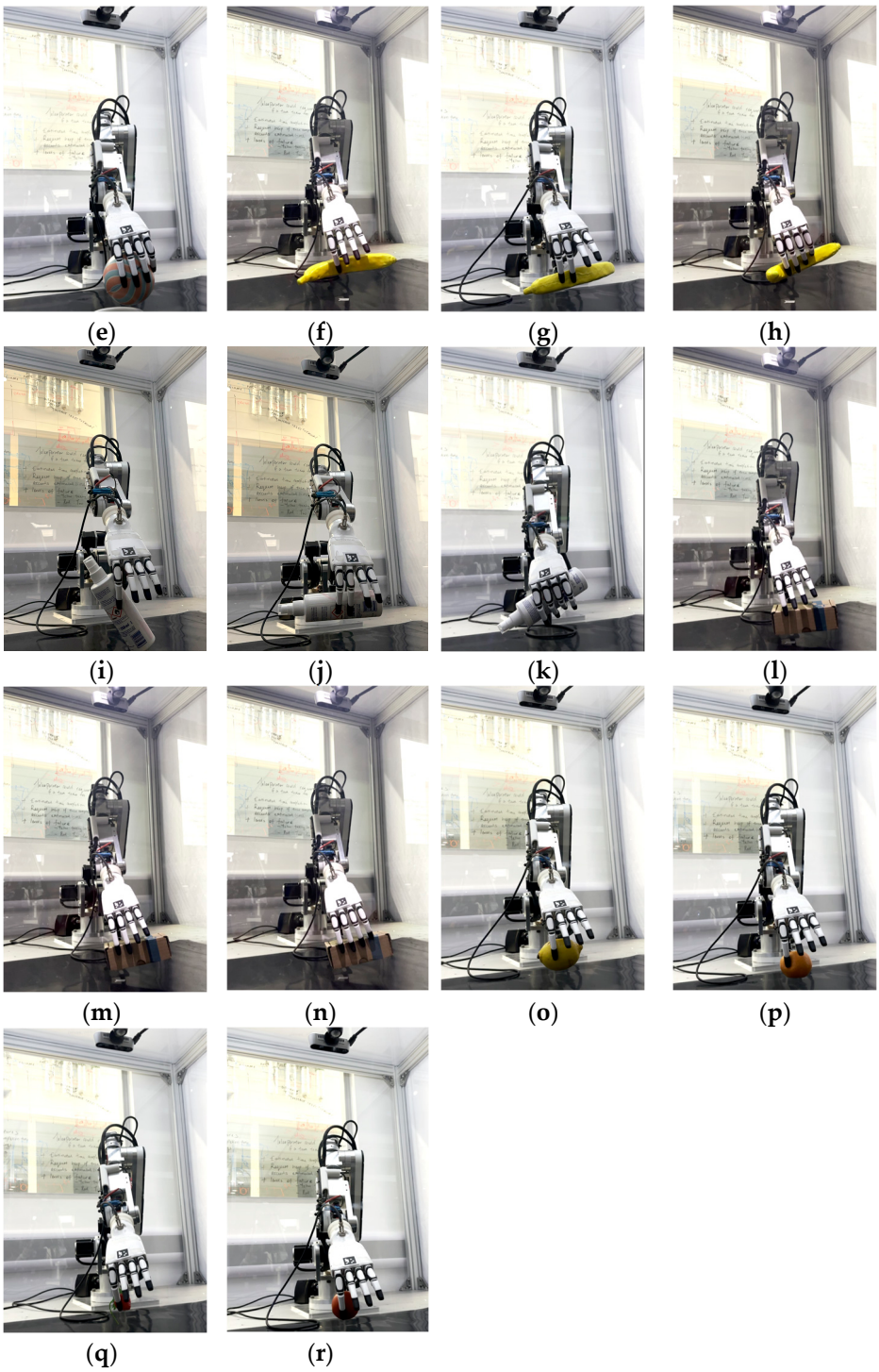


Figure A4. Real-time grasping results: (**a**) Apple (2 finger), (**b**) Apple (3 finger), (**c**) Apple (5 finger), (**d**) Ball (2 finger), (**e**) Ball (5 finger), (**f**) Banana (2 finger), (**g**) Banana (3 finger), (**h**) Banana (5 finger), (**i**) Bottle (2 finger), (**j**) Bottle (3 finger), (**k**) Bottle (5 finger), (**l**) Box (2 finger), (**m**) Box (3 finger), (**n**) Box (5 finger), (**o**) Lemon (2 finger), (**p**) Orange (2 fingers), (**q**) Strawberry (2 fingers), (**r**) Tomato (2 fingers).

Appendix A.3. Grasp Configuration

The grasp configuration system relies on detailed feature extraction and preprocessing steps for effective prediction. Key features include:

- Object Features

The system extracts the object category (e.g., apple, bottle), grasping area, and surface area from the object detection pipeline. Figure A5 shows the grasping area (red line) of the detected objects. These features provide a foundation for predicting the grasp configuration. The shortest segment of the object to be grasped is also identified to ensure precision in manipulation.

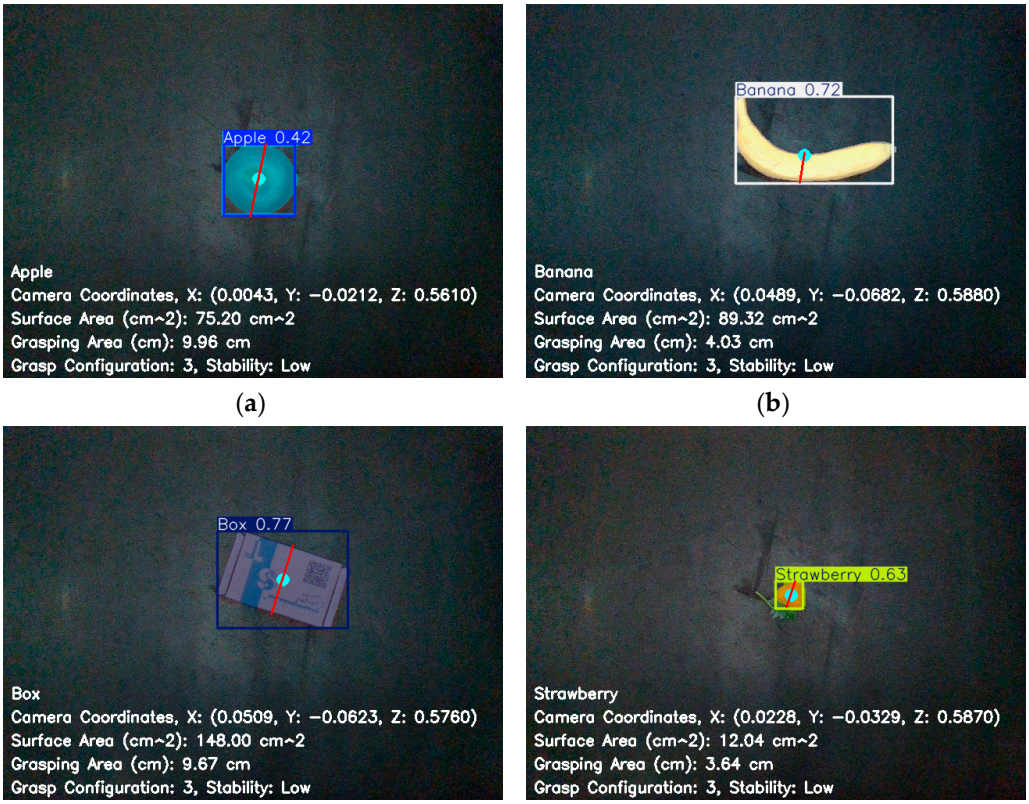


Figure A5. Real-time object detection with grasping area (red line) (**a**) Apple (3 fingers), (**b**) Banana (3 fingers), (**c**) Box (3 fingers), (**d**) Strawberry (3 fingers).

- Grasping Ratio

  To quantify the ease of grasp, the grasping ratio is calculated as:

$$\text{Ratio} = \frac{\text{Grasping Area}}{\text{Surface Area}} \tag{A1}$$

where Surface Area > 0. This metric captures the proportional relationship between the graspable area and the overall surface area, offering insights into the grasping ability of the object.

- Stability Parameter

  A stability feature is introduced to classify objects into one of three discrete stability levels—low, medium, or high. These levels are assigned numerical values of 0, 1, and 2, respectively, to facilitate machine-learning compatibility.

- Feature Normalization

  Numerical features, such as grasping area, surface area, and grasping ratio, are normalized using standard scaling techniques to ensure uniformity across feature magnitudes, preventing any one feature from disproportionately influencing the model. The object

category is label-encoded into numerical values to enable its use within the machine learning framework.

The predictive model is a pre-trained Random Forest classifier designed to output the grasp configuration $g_{conf} \in \{2, 3, 5\}$ representing the number of fingers required for grasping. The model uses a feature vector combining the encoded object category, normalized numerical features, and mapped stability values. Trained on a labelled dataset of diverse objects, the Random Forest classifier ensures robustness and adaptability to various geometries and sizes. Once an object is detected by the robotic vision system, the extracted features are processed and fed into the trained model. The system predicts the optimal grasp configuration, directing the robotic manipulator to execute the most appropriate grasp. Users can adjust the stability feature to achieve grasps with low, medium, or high stability. If stability data is unavailable, the system randomly assigns stability levels before making predictions. This integration into the perception-action pipeline ensures seamless and efficient robotic operation. Below Figure A6 shows the training results of the Random Forest Classifier Model.

```
Classification Report:
              precision    recall  f1-score   support

           2       0.89      0.73      0.80        11
           3       0.67      0.80      0.73         5
           5       0.75      1.00      0.86         3

    accuracy                           0.79        19
   macro avg       0.77      0.84      0.79        19
weighted avg       0.81      0.79      0.79        19


Confusion Matrix:
[[8 2 1]
 [1 4 0]
 [0 0 3]]
```

Figure A6. Grasp Prediction Model performance.

References

1. Hou, Z.; Li, Z.; Fadiji, T.; Fu, J. Soft Grasping Mechanism of Human Fingers for Tomato-Picking Bionic Robots. *Comput. Electron. Agric.* **2021**, *182*, 106010. [CrossRef]
2. Zheng, W.; Xie, Y.; Zhang, B.; Zhou, J.; Zhang, J. Dexterous Robotic Grasping of Delicate Fruits Aided with a Multi-Sensory e-Glove and Manual Grasping Analysis for Damage-Free Manipulation. *Comput. Electron. Agric.* **2021**, *190*, 106472. [CrossRef]
3. Brown, E.; Rodenberg, N.; Amend, J.; Mozeika, A.; Steltz, E.; Zakin, M.R.; Lipson, H.; Jaeger, H.M. Universal Robotic Gripper Based on the Jamming of Granular Material. *Proc. Natl. Acad. Sci. USA* **2010**, *107*, 18809–18814. [CrossRef]
4. Ireri, D.; Belal, E.; Okinda, C.; Makange, N.; Ji, C. A Computer Vision System for Defect Discrimination and Grading in Tomatoes Using Machine Learning and Image Processing. *Artif. Intell. Agric.* **2019**, *2*, 28–37. [CrossRef]
5. Dimeas, F.; Sako, D.V.; Moulianitis, V.C.; Aspragathos, N.A. Design and Fuzzy Control of a Robotic Gripper for Efficient Strawberry Harvesting. *Robotica* **2015**, *33*, 1085–1098. [CrossRef]
6. Hayashi, S.; Shigematsu, K.; Yamamoto, S.; Kobayashi, K.; Kohno, Y.; Kamata, J.; Kurita, M. Evaluation of a Strawberry-Harvesting Robot in a Field Test. *Biosyst. Eng.* **2010**, *105*, 160–171. [CrossRef]
7. Morales, A.; Prats, M.; Felip, J. Sensors and Methods for the Evaluation of Grasping. In *Grasping in Robotics*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 10, pp. 77–104.
8. Zhuoling, H.; Sam, W.; Simon, P. Towards Automated Strawberry Harvesting: Identifying the Picking Point. In Proceedings of the Towards Autonomous Robotic Systems, TAROS 2017, Guildford, UK, 19–21 July 2017; Gao, Y., Fallah, S., Jin, Y., Lekakou, C., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2017; Volume 10454, pp. 222–236.
9. Kim, U.; Jung, D.; Jeong, H.; Park, J.; Jung, H.M.; Cheong, J.; Choi, H.R.; Do, H.; Park, C. Integrated Linkage-Driven Dexterous Anthropomorphic Robotic Hand. *Nat. Commun.* **2021**, *12*, 7177. [CrossRef] [PubMed]
10. Berkenkamp, F.; Schoellig, A.P. Safe and Robust Learning Control with Gaussian Processes. In Proceedings of the 2015 European Control Conference (ECC), Linz, Austria, 15–17 July 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 2496–2501.
11. Rasmussen, C.E. *Gaussian Processes in Machine Learning*; Springer: Berlin/Heidelberg, Germany, 2003.
12. Schulz, E.; Speekenbrink, M.; Krause, A. A Tutorial on Gaussian Process Regression: Modelling, Exploring, and Exploiting Functions. *J. Math. Psychol.* **2018**, *85*, 1–16. [CrossRef]

13. Garcia-Sillas, D.; Gorrostieta-Hurtado, E.; Soto-Vargas, E.; Diaz-Delgado, G.; Rodriguez-Rivero, C. Learning from Demonstration with Gaussian Processes. In Proceedings of the 2016 IEEE Conference on Mechatronics, Adaptive and Intelligent Systems (MAIS), Hermosillo, Mexico, 20–22 October 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–6.
14. Garcia, D.; Gorrostieta, E.; Vargas Soto, E.; Rodriguez Rivero, C.; Diaz Delgado, G. Learning from Demonstration with Gaussian Process Approach for an Omni-Directional Mobile Robot. *IEEE Lat. Am. Trans.* **2018**, *16*, 1250–1255. [CrossRef]
15. Choi, S.; Lee, K.; Oh, S. Robust Learning from Demonstration Using Leveraged Gaussian Processes and Sparse-Constrained Optimization. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 470–475.
16. Choi, S.; Lee, K.; Oh, S. Robust Learning From Demonstrations With Mixed Qualities Using Leveraged Gaussian Processes. *IEEE Trans. Robot.* **2019**, *35*, 564–576. [CrossRef]
17. Deisenroth, M.P.; Fox, D.; Rasmussen, C.E. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 408–423. [CrossRef] [PubMed]
18. Graeve, K.; Stueckler, J.; Behnke, S. Learning Motion Skills from Expert Demonstrations and Own Experience Using Gaussian Process Regression. In Proceedings of the ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics), Munich, Germany, 7–9 June 2010; pp. 1–8.
19. Rahmatizadeh, R.; Abolghasemi, P.; Bölöni, L.; Levine, S. Vision-Based Multi-Task Manipulation for Inexpensive Robots Using End-To-End Learning from Demonstration. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation, Brisbane, QLD, Australia, 21–25 May 2018.
20. Correia, A.; Alexandre, L.A. Hierarchical Decision Transformer. In Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 1–5 October 2023.
21. Dalal, M.; Mandlekar, A.; Garrett, C.; Handa, A.; Salakhutdinov, R.; Fox, D. Imitating Task and Motion Planning with Visuomotor Transformers. *arXiv* **2023**, arXiv:2305.16309.
22. George, A.; Farimani, A.B. One ACT Play: Single Demonstration Behavior Cloning with Action Chunking Transformers. *arXiv* **2023**, arXiv:2309.10175.
23. Tianci, G. Transformer-XL for Long Sequence Tasks in Robotic Learning from Demonstration. *arXiv* **2024**, arXiv:2405.15562.
24. Trinh, M.; Behery, M.; Emara, M.; Lakemeyer, G.; Storms, S.; Brecher, C. Dynamics Modeling of Industrial Robots Using Transformer Networks. In Proceedings of the 2022 Sixth IEEE International Conference on Robotic Computing (IRC), Naples, Italy, 5–7 December 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 164–171.
25. Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.-X.; Yan, X. Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019.
26. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 12. [CrossRef]
27. Han, Y.; Yu, K.; Batra, R.; Boyd, N.; Mehta, C.; Zhao, T.; She, Y.; Hutchinson, S.; Zhao, Y. Learning Generalizable Vision-Tactile Robotic Grasping Strategy for Deformable Objects via Transformer. *IEEE/ASME Trans. Mechatron.* **2021**, *30*, 554–566. [CrossRef]
28. Jiang, J.; Cao, G.; Butterworth, A.; Do, T.-T.; Luo, S. Where Shall I Touch? Vision-Guided Tactile Poking for Transparent Object Grasping. *IEEE/ASME Trans. Mechatron.* **2023**, *28*, 233–244. [CrossRef]
29. Fu, J.; Yu, Z.; Guo, Q.; Zheng, L.; Gan, D. A Variable Stiffness Robotic Gripper Based on Parallel Beam with Vision-Based Force Sensing for Flexible Grasping. *Robotica* **2023**, 1–19. [CrossRef]
30. Verotti, M.; Masarati, P.; Morandini, M.; Belfiore, N.P. Active Isotropic Compliance in Redundant Manipulators. *Multibody Syst. Dyn.* **2020**, *49*, 421–445. [CrossRef]
31. Zhao, C.; Liu, J.; Ma, D. IFEM2.0: Dense 3-D Contact Force Field Reconstruction and Assessment for Vision-Based Tactile Sensors. *IEEE Trans. Robot.* **2025**, *41*, 289–305. [CrossRef]
32. Kadalagere Sampath, S.; Wang, N.; Wu, H.; Liu, C.; Jiang, J.; Yang, C. Integrating Vision and Learning-Based Method for Dexterous Grasping. In Proceedings of the 2024 29th International Conference on Automation and Computing (ICAC), Sunderland, UK, 28 August 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 1–6.
33. Annin, C. Annin Robotics. AR4 Robotic Arm. Available online: https://www.anninrobotics.com/ (accessed on 2 November 2024).
34. Inspire Robots Dexterous Hand. Available online: https://en.inspire-robots.com/product-category/the-dexterous-hands (accessed on 4 March 2022).
35. Kadalagere Sampath, S.; Wang, N.; Wu, H.; Yang, C. Review on Human-like Robot Manipulation Using Dexterous Hands. *Cogn. Comput. Syst.* **2023**, *5*, 14–29. [CrossRef]
36. Ultralytics (2023) YOLOv8. Available online: https://github.com/ultralytics/ultralytics (accessed on 25 April 2024).
37. Open Images Dataset V7. Available online: https://storage.googleapis.com/openimages/web/index.html (accessed on 18 March 2024).

38. Dwyer, B.; Nelson, J.; Hansen, T. Roboflow (Version 1.0) [Software]. Available online: https://roboflow.com/ (accessed on 17 December 2024).
39. Krishna Origanti, V. Recognition and Reproduction of Force-Based Robot Skills via Learning from Demonstration. Ph.D. Thesis, RWTH Aachen University, Aachen, Germany, 2021.
40. Leap Motion Controller. Available online: https://www.ultraleap.com/leap-motion-controller-overview/ (accessed on 19 April 2024).
41. Zeng, C.; Li, S.; Member, S.; Chen, Z.; Yang, C.; Member, S.; Sun, F.; Zhang, J. Multi-Fingered Robot Hand Compliant Manipulation Based on Vision-Based Demonstration and Adaptive Force Control. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *34*, 5452–5463. [CrossRef] [PubMed]
42. Handa, A.; Van Wyk, K.; Yang, W.; Liang, J.; Chao, Y.-W.; Wan, Q.; Birchfield, S.; Ratliff, N.; Fox, D. DexPilot: Vision Based Teleoperation of Dexterous Robotic Hand-Arm System. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 9164–9170.