6D Strawberry Pose Estimation: Real-time and Edge AI Solutions Using Purely Synthetic Training Data

Saptarshi Neil Sinha[1], Paul Julius Kühn[2], Mika Silvan Goschke[3], Michael Weinmann[4]

[1] Fraunhofer IGD, Fraunhoferstr. 5, 64283 Darmstadt, Germany - saptarshi.neil.sinha@igd.fraunhofer.de
[2] Fraunhofer IGD, Fraunhoferstr. 5, 64283 Darmstadt, Germany - julius.kuehn@igd.fraunhofer.de
[3] Fraunhofer IGD, Fraunhoferstr. 5, 64283 Darmstadt, Germany - mika.silvan.goschke@igd.fraunhofer.de
[4] Delft university of technology, Van Mourik Broekmanweg 6, 2628 XE Delft, The Netherlands - m.weinmann@tudelft.nl

Keywords: 6D pose estimation, Synthetic data generation, Deep learning, Edge AI, Agricultural robotics, Selective harvesting

Abstract

Automated and selective harvesting of fruits is increasingly vital due to high costs and seasonal labor shortages especially in advanced economies. This paper focuses on 6D pose estimation of strawberries using purely synthetic data generated through a procedural pipeline for photorealistic rendering. We employ the single-shot YOLOX-6D-Pose algorithm that leverages the YOLOX backbone (i.e., a specific deep convolution network that extracts hierarchical image features used for object detection), known for its balance between speed and accuracy, and support for edge inference. To address the lacking availability of training data, we present a flexible pipeline for generating realistic synthetic data from various 3D strawberry models via the procedural Blender pipeline, enhancing its value for training pose estimation algorithms. Quantitative evaluations show YOLOX-6D-Pose algorithm achieve comparable accuracy on both the NVIDIA RTX 3090 and Jetson Orin Nano, measured by several ADD-S metrics, which measure 6D object pose estimation accuracy by computing the average closest-point distance between model points under predicted and ground-truth poses (for symmetric objects) and evaluating it against chosen thresholds. The RTX 3090 offers superior processing speed, while the Jetson Orin Nano is ideal for resource-constrained environments, suitable for agricultural robotics. Qualitative results confirm the model’s ability to accurately infer poses of ripe and partially ripe strawberries, though challenges remain with unripe specimens. This indicates potential for future enhancements, particularly in detecting unripe strawberries by exploring color variations. The methodology can also be adapted for other fruits like apples, peaches, and plums, broadening its impact in agricultural automation.

1. Introduction

In the rapidly advancing world of smart farming, the use of robotic systems is revolutionizing fruit harvesting (Duckett et al., 2018). These cutting-edge technologies enhance production quality by automating various steps during harvesting. With their ability to perform selective color picking, robots can efficiently identify and harvest only the ripest fruits, ensuring that consumers receive fruits of better quality. Operating continuously, these systems boost efficiency and reduce reliance on seasonal labor, leading to significant cost savings for agricultural operations. Their agile design and user-friendly interfaces make them easy to operate, allowing farmers to integrate them smoothly into existing workflows. Hence, the selective harvesting of fruits through robotic technology offers a promising solution to the societal and economic issues related to agricultural labor shortages especially in advanced economies, supporting the long-term viability and resilience of farming.

Strawberries rank among the most popular fruits globally, with the strawberry industry’s annual retail value surpassing $17 billion (Parsa et al., 2023). However, the economic sustainability of this sector is jeopardized by substantial labor costs, which exceed $1 billion (Parsa et al., 2023) dedicated solely to the selective harvesting process each year. Thereby, strawberries offer significant commercial value in the agricultural sector. However, their harvesting remains a labor-intensive process, with labor costs accounting for approximately 40% of total production expenses (Li and Kasaei, 2024). The reliance on seasonal labor in strawberry harvesting leads to increased costs and challenges, especially during peak seasons when labor shortages are common. The COVID-19 pandemic has further highlighted the urgent need for automated solutions that can improve efficiency and reduce costs in the strawberry market. Hence, there is a need for automated harvesting of strawberries. Companies like Organifarms (Organifarms, 2025) and Tevel (Tevel, 2025) are actively developing automated technologies for picking various fruits, including strawberries, plums, apricots, etc. Most existing robotic vision methods for strawberry picking utilize a 2D to 3D transformation approach (Organifarms, 2025, Tevel, 2025, Montoya-Cavero et al., 2022). These methods typically employ either traditional image processing algorithms or machine learning techniques to first determine the 2D coordinates of strawberries in an image. This data is then correlated with depth information from specialized depth sensors to create approximate 3D coordinates. However, despite advancements in designing fault-tolerant end-effectors, the overall performance of these robotic systems remains suboptimal due to the incomplete 3D pose information of the target strawberries. A comprehensive understanding of the 6 degrees of freedom (6DoF) pose of each strawberry is crucial for a robotic arm to effectively and safely separate the target fruit especially in a highly cluttered environment. The detection and picking of strawberries is challenging due to their varying shapes and colors. Additionally, the assessment of their ripeness can be inaccurate and many ripe strawberries may not become correctly detected as ripe fruits. This variability complicates the harvesting process, making it difficult for robotic systems to effectively identify and select the optimal fruit. Synthetic data generation can help us simulate these environments for effective pose estimation.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-2-2026-751-2026

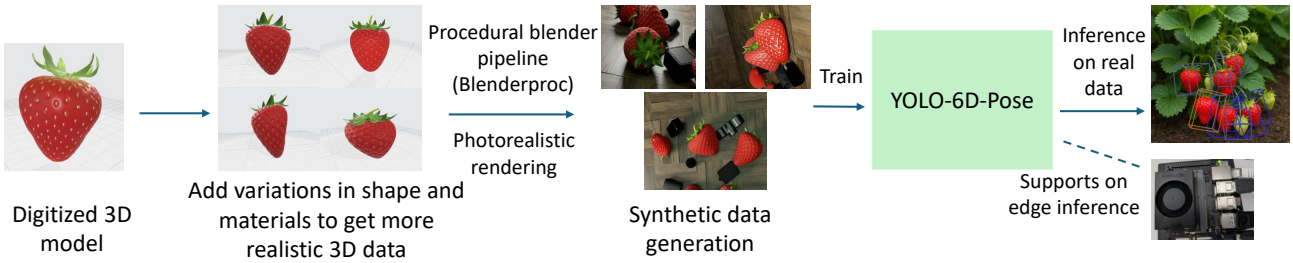

Figure 1. Pipeline for generating synthetic data and then training on it for 6D pose estimation (utilizing the YOLO-6D-Pose (Maji et al., 2024) model) of strawberries. The trained model can also be used for inference on edge devices.

A recent approach using synthetic data generation (Li and Kasaei, 2024) enables the detection of poses through simulation software that explicitly models strawberry scenes. However, the dataset generation process does not simulate different backgrounds or distractors in a scene limiting robustness in real-world, heterogeneous settings. It also omits any exploration of edge-AI optimizations and evaluations—crucial for resource-constrained robotic platforms. Inspired by this approach, we utilize 3D strawberry models with various shapes and colors, placing them in the scene using BlenderProc (Denninger et al., 2023), a procedural pipeline that is free and accessible to the community. This choice enhances our ability to simulate realistic strawberry appearance. Furthermore building on physics-based body simulation, we can accurately position the objects within the scene, allowing for more effective training of our model, while also introducing scene complexity in terms of adding occluding distractor objects. Additionally, we integrate the recent YOLOX 6D pose single-shot (Maji et al., 2024) model, which improves the detection capabilities for our specific application. Furthermore, we support inference on edge devices, specifically the Jetson Orin Nano, and rigorously evaluate our results on these platforms. This aspect sets our approach apart, as to the best of our knowledge no previous studies utilizing synthetic data for pose estimation of strawberries have addressed the performance of their models on edge devices. By demonstrating the feasibility of running our model on such hardware, we enhance the practicality and accessibility of our solution in real-world agricultural settings. This focus on edge device compatibility helps in developing efficient and deployable technologies for strawberry harvesting on robotic system which often have limited GPU resources. In light of these advancements, the main contributions of this paper are as follows:

- We present a collection of digitized 3D models of strawberries that are specifically designed (various colors and shapes) for synthetic data generation. These models can serve as a foundational resource for developing more accurate and realistic datasets for training pose estimation algorithms.
- We introduce a procedural pipeline (see Figure 1) utilizing BlenderProc (Denninger et al., 2023) to generate synthetic data based solely on various strawberry models. This approach not only facilitates the creation of diverse datasets but also allows for easy extensions, such as incorporating different types of distractors and varying environmental conditions to enhance the realism of the simulations. Our 6D strawberry pose synthetic dataset can be accessed under following link: strawberry-6D-pose-synthetic-dataset
- Our methodology includes support for inference on edge devices, specifically the Jetson Orin Nano. We provide a thorough evaluation of the model’s performance on these platforms, demonstrating its practicality and efficiency for real-time applications in agricultural settings. Inference on edge devices enables deployment across a variety of robotic systems, which typically operate with limited computational resources.

2. Related works

Monocular 6D pose estimation can be categorized into two primary approaches. One approach involves directly regressing the final 6D pose, while the other relies on establishing 2D-3D correspondences using techniques such as onthe Random Sample Consensus (RANSAC) based Perspective-n-Point (PnP) algorithm. Both methodologies can incorporate refinement techniques to enhance the accuracy of the initially estimated pose.

Indirect pose estimation approaches: The most widely adopted approach for 6D pose estimation involves establishing 2D-3D correspondences prior to applying the RANSAC-based PnP algorithm to solve for the pose (Rad and Lepetit, 2017, Tekin et al., 2018, Peng et al., 2019, Hu et al., 2019). Initial methods (Rad and Lepetit, 2017, Tekin et al., 2018) focused on computing 2D projections of the corners of a 3D bounding box, which serves as a foundational step in the pose estimation process. Subsequently, Peng et al. (Peng et al., 2019) highlighted an important aspect of pose estimation by demonstrating that utilizing keypoints positioned away from the object’s surface can lead to significant errors in the pose estimation results. To address this issue, they proposed a method that samples multiple keypoints directly on the object model, thereby improving the accuracy of the estimates. To further enhance the robustness of the pose estimation process segmentation techniques combined with a voting mechanism for each correspondence (Peng et al., 2019, Hu et al., 2019). This approach helps ensure that the estimated pose is less sensitive to noise and inaccuracies in the detected keypoints.

Direct pose estimation approaches: Indirect pose estimation approaches cannot be applied for many tasks which require the pose estimation to be differentiable (Wang et al., 2020b) and this issue is addressed by employing direct pose estimation approaches. Direct pose estimation methods focus on directly regressing a representation of the 6D pose. The single Shot Detector (SSD) framework by Liu et al. (Liu et al., 2016) was extended by Kehl et al. (Kehl et al., 2017) to estimate the 6D object pose by discretizations of the pose space and employing a classification approach instead of traditional regression methods. PoseCNN (Xiang et al., 2018) exemplifies a straightforward approach to 6D pose estimation using a convolutional neural network (CNN). It takes a different approach by regressing depth information, the projected 2D center, and a quaternion

for each region of interest within a custom detection pipeline. This method incorporates a Hough voting layer to accurately localize the object’s center within the image. DeepIM (Li et al., 2018) introduced an innovative iterative refinement technique that regresses the difference between the pose hypothesis rendered image and the actual input image. This approach aims to improve the accuracy of pose estimation by continually adjusting the pose based on the discrepancies observed. CosyPose (Labbé et al., 2020) further enhances the DeepIM framework by incorporating a continuous rotation parameterization and leveraging more modern neural network architectures, thereby improving the robustness and accuracy of pose estimation. EfficientPose (Bukschat and Vetter, 2020) proposed an enhancement to the EfficientDet (Tan et al., 2020) object detection model to facilitate pose estimation. GDR-Net (Wang et al., 2021) further enhances 6D pose estimation by addressing the limitations of both indirect and direct regression methods, ultimately proposing the Geometry-guided Direct Regression Network (GDR-Net) for improved performance. However, this method faces challenges due to the lack of proper parameterization for the pose parameters. Additionally, the reliance on the vanilla ADD(S) loss function, combined with aggressive scale augmentation techniques, can lead to instability during training, particularly when applied to large datasets such as YCB-V (Xiang et al., 2018). This challenge is effectively addressed by the YOLO-6D-Pose (Maji et al., 2024), which not only improves robustness but also supports inference on edge devices through the use of the small or tiny backbone of YOLOX. Consequently, we have chosen this method for estimating the 6D pose of strawberries.

Pose estimation for agricultural robotics: In the realm of agricultural robotics, particularly in fruit picking, various 2D detection methods have been deployed effectively. For instance, the YOLO-v4 model has been applied to detect cherry fruits (Gai et al., 2023), successfully addressing challenges posed by environmental factors such as shadows and achieving superior performance compared to the standard YOLO-v4 model. Additionally, the YOLO-v4 model has proven effective for detecting oranges (Mirhaji et al., 2021) and multiple fruit types simultaneously (Dexiao et al., 2021). Recently, advancements have been made with the introduction of YOLOv9 and YOLOv10 models for real-time detection of strawberry stalks (Meng et al., 2025). However, all of these methods rely on 2D detection techniques, which often necessitate a 3D transformation scheme. A more recent approach has employed synthetic data alongside a YOLO-based architecture for the pose estimation of strawberries (Li and Kasaei, 2024). However, this method is focused on strawberries positioned within a specific scene and the provided data does not contain 3D data and respective annotations. To effectively simulate diverse real-world environments, it is essential to utilize physics-based body simulations in conjunction with various distractors, backgrounds, and scenes. To overcome these challenges, we employ different strawberry models along with a procedural rendering pipeline from BlenderProc. This pipeline not only facilitates the creation of realistic simulations but can also be utilized by the community for further development. Additionally, we plan to release our models and the dataset used for this study to to contribute to ongoing research and advancements in this field.

3. Methodology

In the following sections, we will detail the methodology employed for synthetic data generation and pose estimation (see Figure 1). This includes an overview of the techniques used to create realistic strawberry models and the implementation of the YOLO-6D-Pose network (Maji et al., 2024) for accurate pose estimation from RGB images.

Figure 2. Blending the noise texture with a gradient texture along with slight tint of yellow enhances the strawberry’s color by emphasizing variations at the top (left) and transitioning from red to a subtle yellow hue (middle). Finally, some subsurface scattering is applied to get the final output (right). The leaves were modified and a stem was also added (bottom).

3.1 Synthetic data generation

In this section, we describe our workflow for synthetic data generation utilizing 3D digitization techniques and simulation using procedural pipelines.

3D digitization: We used a base model created by a designer and freely available under a CC-BY license from Sketchfab (de Arte, 2025). In our approach, we initiated various modifications to make the rendering of the strawberries more realistic to bridge the reality gap with real strawberries and allow a better generalization to data captured in the wild. First, we utilized Blender’s built-in procedural Perlin Noise texture to introduce a less uniform appearance to the red colour of the strawberry’s surface. This effect was achieved by first subtracting the Noise texture from a Gradient Texture. The blending was designed such that the noise is more pronounced at the upper portion of the strawberry, simulating the natural variations found in real fruit. Finally, this modified texture was employed to blend the vibrant red hue with a subtle yellow (see Figure 2) tint to have more variations in the hue using the “add” blend mode. Furthermore, we also modified the leaves to better reflect the variance encountered for real-world strawberries by manually changing the mesh giving them more random directions. We did this by using Blenders proportional editing, which allows one to edit meshes or polygons so that nearby ones are also affected. This effect tapers off with the distance to the original selections which makes our leaves more natural. Additionally, we added a stem, which is just a simple cylinder with a natural curve and some added irregularities (see Figure 2). It uses the same texture as the leaves. To create different strawberries with variations in shape, we used proportional editing again to slightly change the shape of each strawberry (see Figure 3).

Simulation for dataset generation: The six strawberry models (see Figure 3) were utilized in a simulation process using Blender’s procedural pipeline, known as BlenderProc (Denninger et al., 2023). This method enabled the generation of realistic synthetic strawberry data under a variety of lighting conditions and material settings. The final output consisted of a comprehensive pose estimation dataset, which included RGB im-

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-2-2026-751-2026 | © Author(s) 2026. CC BY 4.0 License.

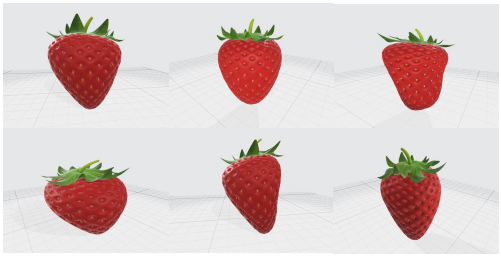

Figure 3. Variations in different shapes of the strawberries used for creating the synthetic data

ages, depth maps, segmentation masks, and annotations formatted according to the BOP standard (Hodan et al., 2018).

To enhance realism, we built our simulation on physically-based rendering (PBR) techniques, which involved modifying the materials, colors, and lighting of the objects within the scene (see Figure 4). Each strawberry was assigned unique textures and colors to capture the variety as encountered in real-world environments. Cameras were strategically positioned around the scene, sampling multiple viewpoints that focused on the strawberries while maintaining a safe distance to avoid being thrown out of the scene. Additionally, the use of a physics simulation allowed both the strawberries and distractor objects to settle realistically on a surface, mimicking their natural interactions (see Figure 4). In this context, distractors refer to additional 3D objects included in the scene to create a more complex and realistic environment. These objects help simulate a cluttered setting, which is commonly encountered in real-world scenarios, and they serve to model occlusion effects (see Figure 4e). By incorporating distractors, the generated dataset becomes more challenging and diverse, thereby enhancing the robustness of learning-based approaches trained on it. For this dataset, distractors were sourced from the YCB-Video (YCBV) dataset (Xiang et al., 2018), which contains a wide variety of 3D models. By incorporating a wide range of distractor models—not just leaves—we force the detector to learn more discriminative features, which improves its generalizability and leads to more reliable strawberry detection across diverse, real-world scenes.

Diverse backgrounds were also simulated by utilizing various textures from the BlenderProc utility (Denninger et al., 2023) known as CCTextures (see Figure 4e and Figure 4f). This resource provided a wide range of high-quality, realistic textures that were applied to the environment, enhancing the overall visual diversity of the scenes. By incorporating different backgrounds, the synthetic dataset better mimicked real-world scenarios, where objects are often placed against a variety of surfaces and settings.

3.2 Pose estimation

The objective of our approach is to estimate the 6D pose $P = [R|t]$ for each object O in an RGB image I, where R represents 3D rotation and t denotes 3D translation. We utilize the YOLO-6D-Pose network (Maji et al., 2024) to perform end-to-end pose-estimation as it supports edge inference, making it suitable for deployment in resource-constrained environments. YOLOX, in particular, is a premier algorithm that balances both efficiency and accuracy, allowing for real-time object detection and pose estimation. This makes it an ideal choice for our 6D pose estimation framework, as we aim to detect strawberries in real time using a robotic arm that operates under limited computational resources.

Pose Parameterization: YOLO-6D-Pose (Maji et al., 2024) utilizes a 6-dimensional representation R_{6d} for rotation, defined as the first two columns of the rotation matrix R. This avoids discontinuities present in traditional representations such as quaternions. For translation, it decouples the 3D translation into 2D projected coordinates and depth, ensuring scale and location invariance.

Network Architecture: The YOLOX-6D-Pose architecture (Maji et al., 2024) extends the YOLOX (Ge et al., 2021) base model by integrating object detection with pose estimation. The entire network is built upon the CSPDarknet53 (Wang et al., 2020a) backbone coupled with PANet-based feature aggregation (Liu et al., 2018). It consists of a rotation head that predicts the R_{6d} representation and a translation head (t) that predicts the normalized translation parameters. This unification allows the model to learn both tasks simultaneously, enabling efficient pose estimation from a single forward pass.

Augmentation Techniques: A variety of augmentation strategies are employed by the YOLOX-6D-Pose model (Maji et al., 2024) to improve the robustness and accuracy of the model. These techniques include end-to-end 6D augmentation, where images are transformed alongside the corresponding poses of all objects. This approach ensures that any modifications applied to the image accurately reflect the changes in the 3D poses of the objects, facilitating better model training.

The YOLOX-6D-Pose architecture (Maji et al., 2024) integrates translation, rotation, and scale augmentations into its data augmentation pipeline, applying the same geometric transformations to both the input image and the 6D poses of all objects present in the image. These augmentations helps maintain alignment between the projected CAD models and the real objects in the image, allowing the model to learn from varied orientations effectively. In addition, the architecture also utilizes scale augmentation, which rescales the images while proportionally adjusting the translation component of the object poses. This aspect is managed carefully to mitigate potential misalignments caused by occlusion or significant rotation angles. By ensuring that scaling is applied judiciously, we maintain the integrity of the pose predictions.

Furthermore, the approach (Maji et al., 2024) applies translation augmentation, where images are randomly translated horizontally and vertically. This adjustment is coupled with corresponding modifications to the pose parameters, allowing the model to learn how to handle variations in object positioning within the scene. The final augmentation strategy systematically combines various transformations: rotation randomly selected from $(0, 10)$ degrees, translation randomly selected from $(0, 10)\%$, and scaling with a factor chosen from $(0.9, 1.1)$. By refining the scale augmentation based on observed discrepancies, the model remains robust against misalignments. Additionally, random color-space augmentations are applied, which do not require any transformations of pose parameters, further enhancing the diversity of our training dataset.

Loss Function: To optimize the pose parameters, a combination of various loss functions is used by the YOLOX-6D-Pose model (Maji et al., 2024), focusing on the ADD(-S) metric (Hinterstoisser et al., 2013), which couples rotation and translation. This approach not only aims to enhance the overall

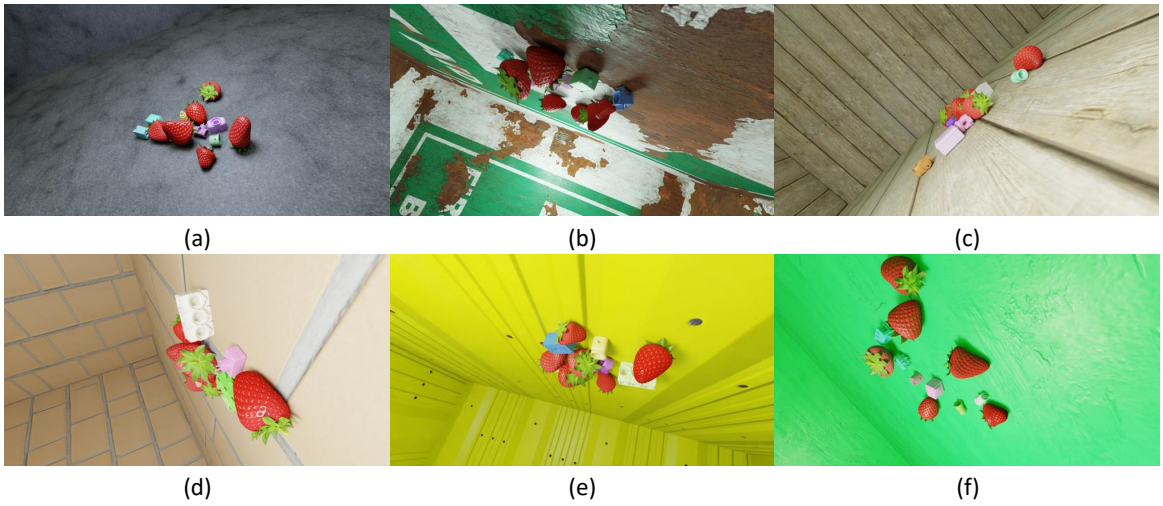

Figure 4. Examples of the synthetic dataset generated using Blender’s procedural pipeline (Denninger et al., 2023). The images illustrate variations in materials and lighting conditions (a and b), along with occlusion effects (c). Additionally, the dataset features occlusions created by distractors (d, e, and f) and showcases diverse backgrounds (e and f).

pose estimation accuracy but also to optimize each component of the pose independently to address the limitations of the ADD(-S) loss.

ADD(-S) Loss: The ADD(-S) loss is designed to evaluate the accuracy of the estimated poses based on the distance between projected points of the predicted and ground-truth poses. It is defined as follows:

$$L_{ADD(-S)} = \begin{cases} L_{asym} = \frac{1}{m}\sum_{x\in m} \|(R_p x + t_p) - (R_g x + t_g)\|_d^2 \\ L_{sym} = \frac{1}{m}\sum_{x_1\in m} \min_{x_2\in m} \|(R_p x_1 + t_p) - (R_g x_2 + t_g)\|_d^2 \end{cases} \tag{1}$$

where, R_g and t_g denote the ground-truth pose, while R_p and t_p represent the predicted pose. The set M consists of the object’s 3D model points, m is the number of points in the model, and d_m is the object diameter. While L_{asym} computes the distance directly between corresponding points, L_{sym} considers the minimum distance from each point in the predicted set to any point in the ground-truth set. However, L_{sym} can be relaxed, particularly for symmetric objects.

Translation Loss: The network predicts the projection of the translation components $[t_x, t_y]$ in the image space, an additional loss term is used to enforce the accuracy of these 2D predictions. This is achieved by using a simplified version of the object keypoint similarity (OKS) loss (Maji et al., 2022) defined as:

$$L_{OKS} = 1 - \text{OKS} = 1 - \exp\left(-\frac{d^2}{2s^2}\frac{1}{b_k^2}\right) \tag{2}$$

where, d is the Euclidean distance between the predicted and ground-truth centroids, s_b is the area of the object, and k is a keypoint-specific weight set empirically to 0.1. For the t_z component of the loss, we employ the Absolute Relative Difference (ARD) loss, defined as:

$$L_{ARD} = 1 - \frac{t_{zp}}{t_{zg}} \tag{3}$$

Here, t_{zp} and t_{zg} are the predicted and ground-truth values for t_z, respectively.

Rotation Loss: The rotation loss follows the YOLO-6D (Maji et al., 2024) approach, first introduced in (Zhou et al., 2018) where 3D rotations are represented using a continuous 6D representation derived from the first two columns of the rotation matrix. We compute the L1 loss between the predicted and ground truth 6D rotation vectors, which provides a smooth, discontinuity-free optimization landscape compared to alternative rotation parameterizations such as Euler angles or quaternions. Finally, the overall pose loss is formulated as follows:

$$L_{pose} = \lambda_{ADD(-S)}L_{ADD(-S)} + \lambda_{rot}L_{rot} + \lambda_{OKS}L_{OKS} + \lambda_{ARD}L_{ARD} \tag{4}$$

where, $\lambda_{ADD(-S)}$, λ_{rot}, λ_{OKS}, and λ_{ARD} are empirically chosen weights for each respective loss term, allowing for balanced optimization across all components of the pose.

Implementation details: The weights $\lambda_{ADD(-S)}$, λ_{rot}, λ_{OKS}, and λ_{ARD} (see Equation 4) were all set to 1. The training of our model was conducted using four NVIDIA A100 GPUs with a batch size of 32. We employed the Stochastic Gradient Descent (SGD) optimizer, utilizing a cosine learning rate scheduler. The base learning rate was set to 1×10^{-2}, and the model was trained for a total of 300 epochs. We utilized the YOLOX-S backbone for training, as it supports inference on edge devices and all results are presented using this backbone. Additionally, the training process took into account the symmetry of the strawberries.

4. Evaluation

In this section, we will provide a detailed evaluation of our 6D pose inference method. We will start with a quantitative assessment, presenting performance metrics such as average processing times for both high-performance and edge devices, along with comparisons of ADD-S metric, rotation and translation errors. Following this, we will showcase qualitative results from our model applied to synthetic test data and real-world strawberries from a dataset. Through this evaluation, we illustrate both the strengths and areas for improvement in our approach.

4.1 Quantitative evaluation

Metrics: We evaluate our method based on the BOP challenge (Hodan et al., 2018) and the ADD(-S) 0.1d metrics (Hinterstoisser et al., 2013). The ADD metric evaluates the average

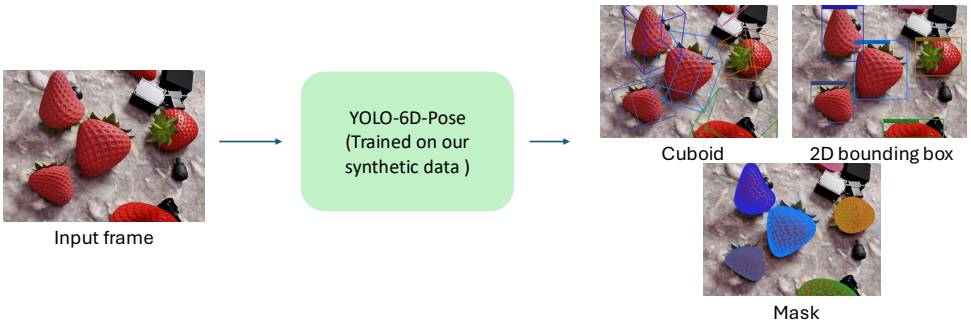

Figure 5. The figure illustrates the output of the YOLO-6D-Pose (Maji et al., 2024) trained using our synthetic data. It outputs the pose, the 2D detections and the corresponding mask.

Table 1. Quantitative analysis of pose inference performance: average forward time, non-maximum suppression (NMS) time, and total average inference time

Metric	NVIDIA RTX 3090	Jetson Orin Nano
Average Forward Time	22.30 ms	35.57 ms
Average NMS Time	2.21 ms	14.02 ms
Average Inference Time	24.51 ms	49.59 ms

distance of 3D model points between the ground truth pose and the predicted pose, with a pose deemed correct if this average distance is less than 10% of the object's diameter. In contrast, the ADD-S metric computes the average distance from the predicted pose to the nearest points of the ground truth. For symmetric objects, the ADD-S metric is employed, while for non-symmetric objects, the standard ADD metric is used. Additionally, we compute the rotation error using the formula *rotation error* = $\arccos\left(\frac{\text{trace}(R^T\hat{R})-1}{2}\right)$ (in degrees), where R is the actual rotation and $\hat{R}$ is the predicted rotation. The translation error is calculated as *translation error* = $||t-\hat{t}||$, where t is the actual translation and $\hat{t}$ is the predicted translation. Furthermore, we conduct a performance analysis based on the average non-maximum suppression time, average forward time, and total average time of the network on both the RTX 3090 and the Jetson Orin Nano. This analysis compares edge devices, such as the Jetson Orin Nano, and non-edge devices, like the RTX 3090, highlighting the performance characteristics relevant for various application scenarios. The Jetson Orin Nano was chosen for its excellent performance-to-power efficiency, compatibility with NVIDIA's ecosystem, and cost-effectiveness for edge AI applications, surpassing other edge devices.

Performance analysis: The quantitative analysis of pose inference performance shows that there is no significant differences between the NVIDIA RTX 3090 and the Jetson Orin Nano (see Table 1). The RTX 3090 demonstrates a faster average forward time and non-maximum suppression (NMS) time compared to the Jetson Orin Nano (see Table 1). This difference is expected, as the Jetson Orin Nano, being an edge device, has lower computational performance than the high-performance RTX 3090. Additionally, the Jetson Orin Nano is much less costly, making it a more budget-friendly option for certain applications. However, its processing times are still acceptable for use in edge devices, which is crucial for robotic applications.

Accuracy of pose estimation: In terms of accuracy, both devices perform similarly across various ADD-S metrics (Hinterstoisser et al., 2013), which evaluate the precision of 6D object pose estimation (see Table 2). The metrics ADD-S_0p1, ADD-S_0p2, ADD-S_0p3, ADD-S_0p4, and ADD-S_0p5 represent the average distances between the estimated and ground truth poses, with thresholds of 0.1, 0.2, 0.3, 0.4, and 0.5 units, respectively. These thresholds indicate the maximum allowable distance for a successful pose estimation. Across these metrics, the results for the RTX 3090 and the Jetson Orin Nano are almost similar in most cases. Specifically, the average rotation error is slightly lower at 17.31° for the RTX 3090 compared to 17.70° for the Jetson Orin Nano. Similarly, the translation error (see Table 2) is comparable, with values of 23.10 mm for the RTX 3090 and 23.15 mm for the Jetson Orin Nano.

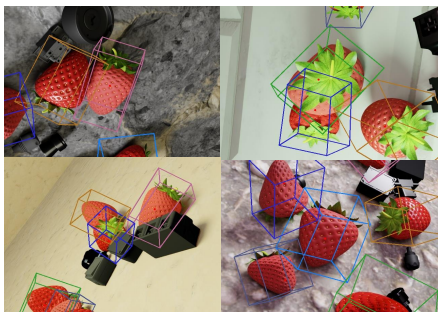

Figure 6. Qualitative analysis of the synthetic dataset, showcasing the accuracy of our model on unseen test data. The results indicate that the model is plausibly capable of inferring the 6D pose of the strawberries.

Table 2. Comparison of ADD-S metrics, rotation error, and translation error between high-performance and edge devices

Metric	NVIDIA RTX 3090	Jetson Orin Nano
ADD-S_0p1_avg	0.7228	0.7231
ADD-S_0p2_avg	0.7599	0.7594
ADD-S_0p3_avg	0.7716	0.7716
ADD-S_0p4_avg	0.7791	0.7791
ADD-S_0p5_avg	0.7831	0.7832
rotation_error_avg (in degrees)	17.31°	17.70°
translation_error_avg (in mm)	23.10	23.15

Overall, these results indicate that while both devices can achieve similar accuracy, the RTX 3090 excels in speed, making it the preferred choice for applications that require rapid and precise pose inference. In contrast, the Jetson Orin Nano is an excellent option for scenarios where inference is needed in resource-limited environments or on low-power devices.

4.2 Qualitative evaluation

Figure 5 shows the output of the YOLO-6D-Pose model (Maji et al., 2024) trained on our synthetic data, illustrating its ability to effectively generate pose estimations, 2D detections, and

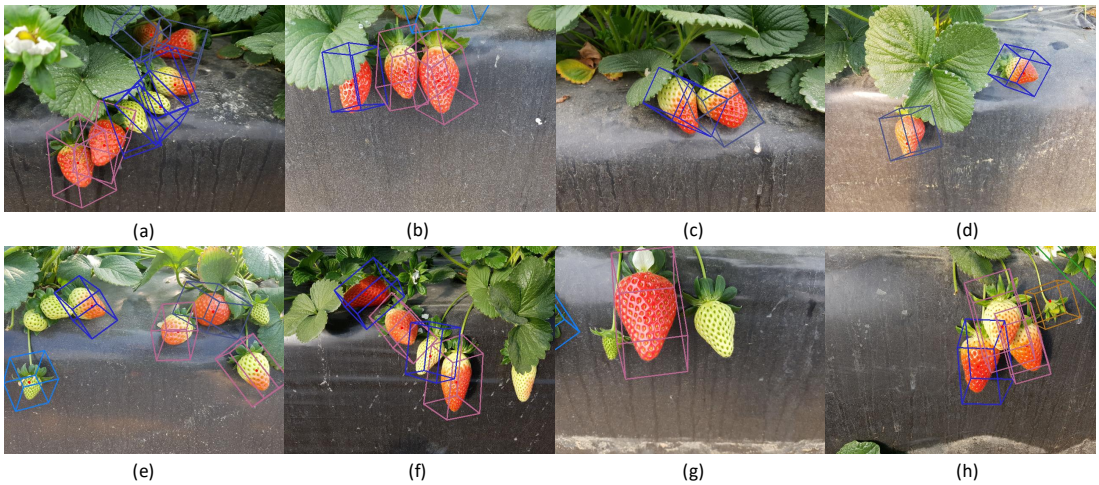

Figure 7. Qualitative analysis of 6D pose inference results on real strawberries from the Strawberry Pose Computer Vision Dataset (Strawberry, 2025). The results indicate effective detection of ripened and partially ripened strawberries (see (b), (d), and (h)). However, detection of completely unripe strawberries is inconsistent, as shown in images (e), (f), and (g).

corresponding masks. The qualitative analysis of our model's performance is presented through two datasets: our synthetic dataset (only test data that has not been used in training) and a real-world dataset from the Strawberry Pose Computer Vision Dataset (Strawberry, 2025). We trained the model with each variation in shape and color as individual classes and the different colors of the bounding boxes indicate the respectively detected class.

In Figure 6, we showcase the accuracy of our model on unseen synthetic test data. The results demonstrate the model's capability to reliably infer the 6D pose of strawberries, indicating its robust performance in controlled conditions.

Figure 7 presents the qualitative analysis of 6D pose inference results on real strawberries. The model effectively detects ripened and partially ripened strawberries, as evidenced in images (b), (d), and (h). However, the detection of completely unripe strawberries proves to be inconsistent, as illustrated in images (e), (f), and (g). This inconsistency highlights the challenges faced by the model when dealing with varying degrees of ripeness, suggesting areas for further improvement in detection accuracy for unripe strawberries.

5. Conclusion and Future Work

In this work, we successfully demonstrated the effectiveness of the YOLO-6D-Pose model for 6D pose inference of strawberries using both synthetic and real-world datasets. We created a flexible pipeline and approach for generating synthetic strawberry data from different 3D models using a procedural Blender pipeline. The quantitative evaluation revealed that the NVIDIA RTX 3090 and Jetson Orin Nano achieve virtually identical accuracy across various ADD-S metrics, with slight variations in rotation and translation errors. Notably, the RTX 3090 significantly outperforms the Jetson Orin Nano in terms of speed, making it the preferred choice for applications requiring rapid pose estimation. However, the Jetson Orin Nano remains an excellent option for use in resource-constrained environments, demonstrating that our model is well-suited for robotic applications across different performance tiers. Overall, our results indicate that the model offers flexibility for future implementations in agricultural settings. However, to achieve reliable pose estimation for industrial applications, our method requires improvements in rotation and translation accuracy. Currently, it can be used to support model-based tracking systems that need an initial approximate pose.

The qualitative analysis further supports our findings, showcasing the model's capability to accurately infer the poses of ripened and partially ripened strawberries while identifying areas for improvement in detecting unripe strawberries by considering more color variations which can be addressed in future work. However, if the use-case is to exclude detection of unripe strawberries, we can incorporate them as distractors to improve the model's robustness. Additionally, our approach can also be used for other fruits like apples, peaches, nectarines, apricots, plums, etc. Finally, our new dataset can be used for training diverse detectors to benchmark their performance.

References

Bukschat, A., Vetter, M., 2020. EfficientPose: An Efficient, Accurate and Scalable End-to-End 6D Multi-Object Pose Estimation Approach. *arXiv preprint arXiv:2011.04307*.

de Arte, G. E., 2025. 3d model of a strawberry created by a designer and available at sketchfab. `https://sketchfab.com/3d-models/strawberry-dd6a424807614544835c8cc4529d6f0d`. Author website: `https://www.gelmi.com.br/`. Accessed on: 2025-10-06.

Denninger, M., Winkelbauer, D., Sundermeyer, M., Boerdijk, W., Knauer, M., Strobl, K. H., Humt, M., Triebel, R., 2023. BlenderProc2: A Procedural Pipeline for Photorealistic Rendering. *Journal of Open Source Software*, 8(82), 4901. https://doi.org/10.21105/joss.04901.

Dexiao, K., Junqiu, L., Jiawen, Z., Jiale, X., Qinghui, Z., 2021. Research on fruit recognition and positioning based on you only look once version 4 (yolov4). *Journal of Physics: Conference Series*, 2005, IOP Publishing Ltd, Guilin, China, 012020.

Duckett, T., Pearson, S., Blackmore, S., Grieve, B., Chen, W.-H., Cielniak, G., Cleaversmith, J., Dai, J., Davis, S., Fox, C., From, P., Georgilas, I., Gill, R., Gould, I., Hanheide, M.,

Hunter, A., Iida, F., Mihalyova, L., Nefti-Meziani, S., Neumann, G., Paoletti, P., Pridmore, T., Ross, D., Smith, M., Stoelen, M., Swainson, M., Wane, S., Wilson, P., Wright, I., Yang, G.-Z., 2018. Agricultural robotics: The future of robotic agriculture.

Gai, R., Chen, N., Yuan, H., 2023. A Detection Algorithm for Cherry Fruits Based on the Improved YOLO-v4 Model. *Neural Computing and Applications*, 35(19), 13895–13906. https://doi.org/10.1007/s00521-021-06029-z.

Ge, Z., Liu, S., Wang, F., Li, Z., Sun, J., 2021. Yolox: Exceeding yolo series in 2021.

Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N., 2013. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. *Computer Vision – ACCV 2012*, Springer Berlin Heidelberg, Berlin, Heidelberg, 548–562.

Hodan, T., Michel, F., Brachmann, E., Kehl, W., Buch, A. G., Kraft, D., Drost, B., Vidal, J., Ihrke, S., Zabulis, X., Sahin, C., Manhardt, F., Tombari, F., Kim, T.-K., Matas, J., Rother, C., 2018. Bop: Benchmark for 6d object pose estimation.

Hu, J., Hugonot, P., Fua, M., 2019. Segmentation-driven 6d object pose estimation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, 3385–3394.

Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N., 2017. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, 1521–1529.

Labbé, Y., Carpentier, J., Aubry, M., Sivic, J., 2020. Cosypose: Consistent multi-view multi-object 6d pose estimation. *European Conference on Computer Vision*, Springer, 574–591.

Li, L., Kasaei, H., 2024. Single-shot 6dof pose and 3d size estimation for robotic strawberry harvesting.

Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D., 2018. Deepim: Deep iterative matching for 6d pose estimation. *Proceedings of the European Conference on Computer Vision (ECCV)*, 683–698.

Liu, S., Qi, L., Qin, H., Shi, J., Jia, J., 2018. Path Aggregation Network for Instance Segmentation . *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Computer Society, Los Alamitos, CA, USA, 8759–8768.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A. C., 2016. Ssd: Single shot multibox detector. *European Conference on Computer Vision*, Springer, 21–37.

Maji, D., Nagori, S., Mathew, M., Poddar, D., 2022. Yolo-pose: Enhancing yolo for multi-person pose estimation using object keypoint similarity loss. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, 2637–2646.

Maji, D., Nagori, S., Mathew, M., Poddar, D., 2024. Yolo-6d-pose: Enhancing yolo for single-stage monocular multi-object 6d pose estimation. *2024 International Conference on 3D Vision (3DV)*, 1616–1625.

Meng, Z., Du, X., Sapkota, R., Ma, Z., Cheng, H., 2025. YOLOv10-pose and YOLOv9-pose: Real-time strawberry stalk pose detection models. *Computers in Industry*, 165, 104231.

Mirhaji, H., Soleymani, M., Asakereh, A., Abdanan Mehdizadeh, S., 2021. Fruit detection and load estimation of an orange orchard using the YOLO models through simple approaches in different imaging and illumination conditions. *Computers and Electronics in Agriculture*, 191.

Montoya-Cavero, L.-E., de León Torres, R. D., Gómez-Espinosa, A., Cabello, J. A. E., 2022. Vision systems for harvesting robots: Produce detection and localization. *Computers and Electronics in Agriculture*, 192, 106562.

Organifarms, 2025. Harvesting the future one berry at a time. `https://www.organifarms.de/`. Accessed on: 20.10.2025.

Parsa, S., Debnath, B., Khan, M. A., E., A. G., 2023. Autonomous strawberry picking robotic system (robofruit).

Peng, S., Liu, Y., Huang, Q., Zhou, X., Bao, H., 2019. Pvnet: Pixel-wise voting network for 6dof pose estimation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, 4561–4570.

Rad, M., Lepetit, V., 2017. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, 3828–3836.

Strawberry, 2025. strawberry pose dataset. `https://universe.roboflow.com/strawberry-ind4b/strawberry-pose-7jxqn`. visited on 2025-10-20.

Tan, M., Pang, R., Le, Q. V., 2020. Efficientdet: Scalable and efficient object detection. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10781–10790.

Tekin, B., Sinha, S. N., Fua, P., 2018. Real-time seamless single shot 6d object pose prediction. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 292–301.

Tevel, 2025. A new era of harvesting with tevel’s flying autonomous robots. `https://www.tevel-tech.com/`. Accessed on: 25.10.2025.

Wang, C.-Y., Mark Liao, H.-Y., Wu, Y.-H., Chen, P.-Y., Hsieh, J.-W., Yeh, I.-H., 2020a. Cspnet: A new backbone that can enhance learning capability of cnn. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1571–1580.

Wang, G., Manhardt, F., Shao, J., Ji, X., Navab, N., Tombari, F., 2020b. Self6d: Self-supervised monocular 6d object pose estimation. *European Conference on Computer Vision (ECCV)*.

Wang, G., Manhardt, F., Tombari, F., Ji, X., 2021. Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 16611–16621.

Xiang, Y., Schmidt, T., Narayanan, V., Fox, D., 2018. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes.

Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H., 2018. On the Continuity of Rotation Representations in Neural Networks. *CoRR*, abs/1812.07035. http://arxiv.org/abs/1812.07035.