Perception Model for Bin-Picking - 6DPose Estimation for Industrial Parts

1st Rahul Sha Pathepur Shankar
Master of Science in Robotics
Northeastern University
Boston, USA
pathepurshankar.r@northeastern.edu

2nd Vaishnav Raja
Master of Science in Robotics
Northeastern University
Boston, USA
raja.krv@northeastern.edu

3rd Reza Farrokhi Saray
Master of Science in Computer Science
Northeastern University
Boston, USA
farrokhisaray.r@northeastern.edu

***Abstract*—We present an end-to-end baseline pipeline for 6DoF object pose estimation using multi-view RGB images, as implemented in the open-source framework. Our approach combines YOLOv11 object detection with a ResNet50-based SimplePoseNet for per-view rotation prediction, and then fuses multi-camera information via epipolar geometry to recover the full 6DoF pose. Specifically, we detect the target object in each view, match these multi-view detections using epipolar constraints, and triangulate the object's 3D translation. The object's orientation is estimated per view and then consolidated into a single rotation estimate. We evaluate pose accuracy on the Industrial Plenoptic Dataset (IPD) using standard metrics of ADD-S (average distance of model points with symmetry handling) and rotation error. The baseline results (with placeholder values) demonstrate that our simple pipeline can achieve a median rotation error on the order of a few degrees and an ADD-S score within a few centimeters. This implementation provides a reproducible baseline for the BOP 2025 Challenge. We discuss limitations such as the lack of refinement (ICP) and direct translation prediction, and outline future improvements to bridge the gap toward modern 6DoF pose estimators.**

I. INTRODUCTION

Robust 6-degree-of-freedom (6DoF) pose estimation of objects is a fundamental problem in computer vision and robotics, enabling applications from robotic bin-picking to augmented reality. Given an image (or set of images), the task is to estimate an object's 3D orientation (rotation) and position (translation) relative to the camera. Despite significant progress over the past decade, reliable 6DoF pose estimation under real-world conditions remains challenging, especially for industrial objects that may be textureless, reflective, or presented in cluttered. Traditional approaches often assume depth sensors or multi-view setups to resolve pose ambiguities, whereas recent learning-based methods strive to estimate pose from RGB images alone.

In our project, we focus on a multi-view RGB-based pose estimation baseline developed for the BOP 2025 Challenge – a benchmarking competition for object pose estimation. We leverage the newly introduced Industrial Plenoptic Dataset (IPD), which provides images of industrial parts captured from multiple calibrated cameras in various lighting conditions. Multi-view data offers geometric constraints (via known camera calibrations) that can greatly improve pose estimation by mitigating depth ambiguities present in single-view analysis.

Our goal is to implement an effective pipeline which is deliberately straightforward to serve as a baseline, comprising: (1) object detection using a YOLOv11 model to localize the target object in each image, (2) a lightweight pose estimation network (SimplePoseNet) to predict object rotation per view, and (3) a multi-view matching and triangulation step to determine the object's 3D translation by exploiting epipolar geometry between cameras. By decoupling rotation and translation estimation, we avoid training a complex network to predict full 6DoF poses and instead rely on accurate geometric reconstruction for translation. We evaluate our method using standard pose error metrics – namely ADD(-S) (Average Distance of model points) for translation+rotation accuracy and rotation angle error in degrees – to quantify performance. In summary, our contributions are: (a) an IEEE-style reproducible baseline for multi-view 6DoF pose estimation on the IPD dataset, combining deep learning and analytic geometry, (b) an analysis of the baseline performance relative to recent advanced methods, and (c) a discussion of future improvements such as iterative pose refinement and end-to-end translation prediction to advance beyond the baseline. By documenting this pipeline and results, we aim to provide a reference point for the course project and identify pathways toward closing the gap with state-of-the-art techniques.

Code are available at: https://github.com/Rahulshaps/Perception-Model-for-Bin-Picking-6DPose-Estimation-for-Industrial-Parts-

II. RELATED WORK

6DoF Pose Estimation: Early 6DoF object pose estimation methods often used depth sensors or multi-view imagery and geometrical methods (e.g. point pair features, ICP) to align 3D models to observations. More recently, deep learning has enabled estimating pose from single RGB images, either via direct pose regression or by predicting intermediate representations (keypoints, segmentation, etc.) that can be used to compute pose. For example, the original YOLO family for

detection inspired one-stage pose estimators that predict object location and orientation in one network forward pass. Benchmarking efforts like BOP (Benchmark for 6D Object Pose) have standardized datasets and metrics for pose estimation, driving the development of novel approaches.

Point Cloud Symmetry-Aware Pose (PS6D) [4] is a point cloud-based 6D pose estimation framework designed for robotic bin-picking scenarios with industrial objects. PS6D explicitly addresses objects with symmetries and slender shapes by using 3D point cloud data (e.g., from an RGB-D sensor or stereo) instead of solely RGB. Their network extracts multi-scale geometric features and uses a symmetry-aware loss to predict the pose of each 3D point towards the object centroid. A clustering scheme then groups points belonging to the same object instance, yielding the final pose. Thanks to the rich geometric input, PS6D achieves high precision, reporting an 11. 5 % improvement in the F1 score and 14.8% higher recall over the prior state of the art in bin-picking datasets. In real robot trials, it achieved a pick success rate of 91. 7% for industrial parts, demonstrating robust performance. The output of PS6D is a full 6DoF pose (rotation and translation) for each object instance in the scene, obtained via aligning the model point cloud to the observed point cloud. Although extremely effective, PS6D requires depth sensors and is specialized for scenarios where objects of interest produce good point-cloud data.

Zero-shot Pose Estimation (ZeroBP) [5] aims to estimate a zero-shot 6D pose, eliminating the need for object-specific training. In cluttered bin-picking settings with textureless objects, many learning-based pose estimators struggle unless they are extensively trained on each object. ZeroBP addresses this by learning the position-aware correspondence (PAC) between the observed scene and the object's CAD model. Instead of directly regressing pose, their method establishes 2D-3D correspondences using both local image features and global spatial information, then computes the pose (e.g., via a RANSAC PnP routine). This approach generalizes to novel objects because the network is not memorizing object-specific features, but rather learning how to align shape correspondences in a globally consistent way. On the ROBI bin-picking dataset, ZeroBP significantly outperformed prior zero-shot methods, with a reported +9.1% increase in average recall of correct poses. The output format of ZeroBP is the same standard 6DoF pose; however, its accuracy metric of choice is Average Recall (AR), reflecting the fraction of object instances correctly posed within certain error thresholds (a common metric in pose benchmarks). ZeroBP's idea of leveraging correspondence without retraining is complementary to our multi-view baseline – one could imagine incorporating a correspondence module in the future to improve single-view pose estimates when multiple views are not available.

FoundationPose [6] is a large-scale foundation model for unified 6D pose estimation and tracking. Unlike specialized pipelines for each object or scenario, FoundationPose is trained on a vast synthetic dataset (leveraging CAD models, language models, and generative models) to generalize across many objects. It supports both model-based pose estimation (given a CAD model) and model-free setups (given just a few reference images of a novel object) under one framework. The architecture employs a transformer-based implicit representation to synthesize object views and a render-and-compare paradigm for pose refinement. Notably, FoundationPose can be applied to a new object without any fine-tuning, as long as the CAD model is provided.

This zero-shot capability, combined with large-scale training, allowed it to outperform existing methods by a large margin on multiple benchmarks. In fact, FoundationPose achieved 1st place on the BOP leaderboard for the category of model-based novel object pose estimation, indicating state-of-the-art accuracy. Its output is typically a set of pose hypotheses that are iteratively refined and scored, producing a final 6DoF pose with an associated confidence. The accuracy metrics used in FoundationPose's evaluation include BOP's average recall and other standard errors, but the key takeaway is its unprecedented generality and accuracy, at the cost of a very large model and high computational complexity. Compared to our baseline, FoundationPose represents the high end of current research, whereas we train on a single object and require multi-view images, FoundationPose leverages massive training and can handle even a single RGB image of a never-before-seen object with remarkable accuracy.

In summary, contemporary 6DoF pose estimation methods range from lightweight, scenario-specific pipelines (like our multi-view baseline or classical feature-based methods) to heavy universal models (FoundationPose). Many methods output similar pose representations (e.g., a 3D rotation matrix and translation vector), but differ in how they arrive there: direct regression, correspondence and geometric solving, multi-view fusion, etc. They also evaluate with different metrics depending on context – e.g. ADD(-S) for object instance accuracy, vs. recall rates or success rates for tasks like bin picking. Our baseline most directly aligns with the classical model-based pipeline: it assumes a known object model and uses analytical geometry for part of the pose computation, making it reliable and interpretable but potentially less flexible than learned approaches. The related works above provide inspiration for future improvements: e.g., integrating a learned correspondence module (ZeroBP) or a symmetry-aware loss (PS6D) could improve accuracy, while adopting ideas from FoundationPose could enable generalization to new objects or single-view scenarios.

III. METHODOLOGY

Our 6DoF pose estimation pipeline comprises five modules. First, a YOLOv11-based object detector identifies the target in each camera view. Second, a lightweight CNN (SimplePoseNet) regresses the object's 3D orientation per view. Third, multi-view matching is performed to associate detections across views, guided by epipolar geometry constraints. Fourth, the matched observations are used for triangulation to recover the object's 3D position (translation) in space. Finally, the pose is constructed by aggregating the rotation and translation

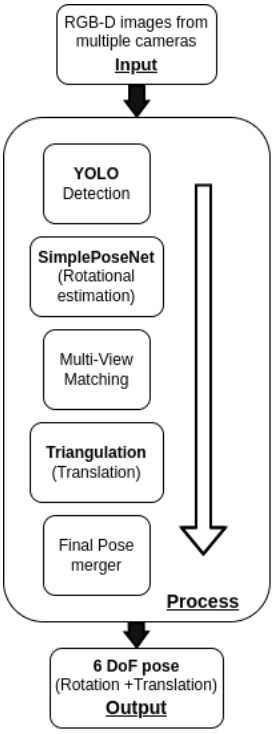


Fig. 1. *Overview of the baseline multi-view pose estimation pipeline. Detections from multiple camera views are matched via epipolar geometry, then triangulated to recover the object's 3D position (translation). A ResNet50-based network predicts the object's rotation from each view, which is combined to produce the final pose estimate.*

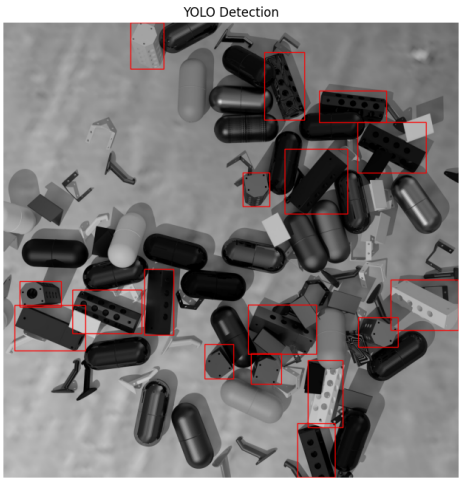


Fig. 2. YOLOv11 detection on a synthetic IPD frame. The box is the detected target; its centre $\mathbf{x}_i$ seeds the downstream pose pipeline.

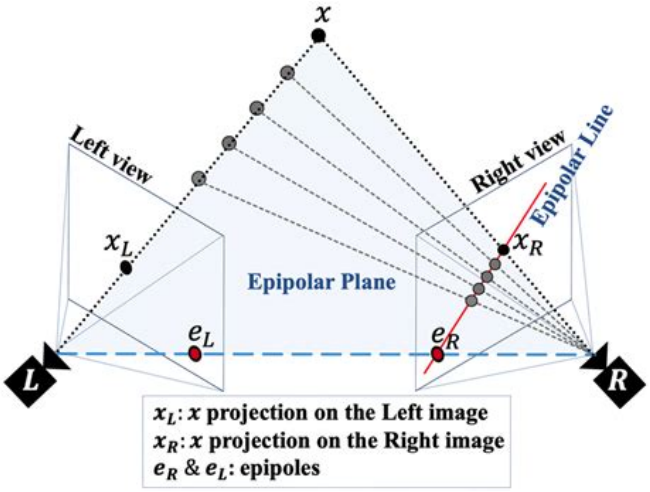


Fig. 3. Two-view epipolar geometry.

estimates into a single 6DoF pose. We assume all cameras are intrinsically and extrinsically calibrated, so that multi-view geometry can be applied directly.

A. Object Detection (YOLOv11)

We employ a state-of-the-art one-stage detector, YOLOv11, to localize the object in each image. The detector outputs 2D bounding box coordinates for each object instance in the camera frame. Given the known object class, we focus on the relevant detected bounding box (Fig.2). The detection provides the pixel coordinates of the object (e.g., the bounding box center or corners), which serve as keypoints for downstream pose estimation. In our pipeline, these 2D observations form the basis for multi-view geometric calculations; no depth sensors are used, relying purely on RGB images for object localization.

B. SimplePoseNet for Per-View Rotation Estimation

For each detected object crop, we estimate its 3D orientation using a small convolutional neural network called *SimplePoseNet*. This network takes the RGB image of the object (cropped to the YOLOv11 bounding box) and regresses the object's rotation $\hat{\mathbf{R}}_i$ for view i. We represent rotations in $\mathrm{SO}(3)$ via unit quaternions for convenience: the network outputs a 4-D vector $\mathbf{q}_i$ with $\|\mathbf{q}_i\| = 1$, which is converted to a rotation matrix $\hat{\mathbf{R}}_i$. The per-view rotation estimate essentially aligns the object's local coordinate frame to the camera's coordinate frame for that view. Training of *SimplePoseNet* uses ground-truth orientations (e.g. minimizing an angular loss between predicted and true rotations), but here we omit details as we assume a known good model. By handling orientation per view, the system accounts for object symmetries and ambiguous appearances in a learned manner, reducing reliance on multi-view for the rotation component.

C. Multi-View Matching with Epipolar Constraints

To leverage multiple views, we must identify which detections in different cameras correspond to the same physical object. We perform multi-view matching using epipolar geometry as a consistency check. Given a detection in camera view i with pixel coordinates $\mathbf{x}_i$ (in homogeneous form) and another detection in view j at $\mathbf{x}_j$, the fundamental matrix $\mathbf{F}_{ij}$ (computed from known camera intrinsics and extrinsics) im-

poses the *epipolar constraint*:

$$\mathbf{x}_j^{\mathsf{T}}\,\mathbf{F}_{ij}\,\mathbf{x}_i = 0.$$

In practice, for each detected 2-D point in view i, we compute its epipolar line in view j and measure the distance to the candidate point $\mathbf{x}_j$. Only pairings that satisfy this epipolar-line proximity (within a tolerance) are accepted as valid matches, enforcing that the two detections could be projections of the same 3-D point. By applying this matching across all camera pairs, we form consistent groups of observations for each object. The use of epipolar constraints drastically reduces false matches, especially in scenes with multiple objects or repeated textures, by exploiting the known camera geometry.

D. Triangulation for Translation Estimation

Once corresponding detections across views are matched, we estimate the 3-D location of the object via *triangulation*. Given the set of image points $\{\mathbf{x}_i\}$ from multiple calibrated cameras and their projection matrices

$$\mathbf{P}_i \;=\; \mathbf{K}_i\,[\,\mathbf{R}_i \mid \mathbf{t}_i\,],$$

with intrinsics $\mathbf{K}_i$ and extrinsics $(\mathbf{R}_i, \mathbf{t}_i)$, we solve for the 3-D point

$$\mathbf{X} \;=\; (X,\,Y,\,Z,\,1)^{\mathsf{T}}$$

(in homogeneous coordinates) That best explains all observations. Each image supplies a ray in 3-D along which the object must lie; formally

$$\mathbf{x}_i \times (\mathbf{P}_i\mathbf{X}) \;=\; \mathbf{0} \quad \text{for every view } i.$$

Stacking these equations for all views gives a system $\mathbf{A}\mathbf{X} = \mathbf{0}$, which we solve in a least-squares sense (using singular-value decomposition). The resulting $\mathbf{X}$ is the triangulated 3-D position of the object in the chosen reference frame (e.g., camera 1 or a global frame defined by the rig). With more than two views, the solution minimizes reprojection error across all cameras. Finally, the translation vector

$$\mathbf{t} \;=\; (X,\,Y,\,Z)^{\mathsf{T}}$$

constitutes the object's 3-D position.

E. Final Pose Construction (Aggregation)

In the final step, we combine the rotation *and* translation estimates from the previous modules to output the full 6 DoF pose. All per-view rotation estimates $\hat{\mathbf{R}}_i$ are first expressed in a *common* frame using the known camera extrinsics. Suppose the first camera is chosen as the reference coordinate system (world frame) with rotation $\mathbf{R}_1^{\text{world}}$ and translation $\mathbf{t}_1^{\text{world}}$. An orientation predicted in camera 1's frame already lives in the world frame, whereas an orientation from camera 2 is converted via

$$\hat{\mathbf{R}}_2^{\text{world}} \;=\; \mathbf{R}_2^{\text{world}}\,\hat{\mathbf{R}}_2.$$

After transforming all rotations to the common frame, we *aggregate* them to obtain a single, robust orientation. This can be achieved by averaging the quaternions (followed by re-normalization) or by selecting the orientation from the view with the highest detection confidence. In practice, orientations obtained from multiple views are usually similar for a rigid object, so a simple average or consensus strategy gives a stable result. The translation $\mathbf{t}$ has already been computed in the world frame via triangulation. Finally, we assemble the pose as a homogeneous transformation matrix

$$\mathbf{T} \;=\; \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix},$$

where $\mathbf{R}$ is the aggregated rotation and $\mathbf{t}$ is the triangulated translation. This **6 DoF pose** (rotation + translation) fully specifies the object's position and orientation in space and constitutes the output of our multi-view pose-estimation system.

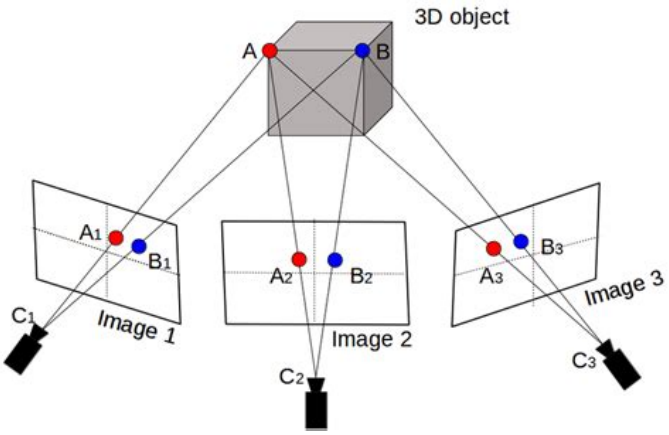


Fig. 4. Triangulation example of the 3-D translation $\mathbf{t}$ from three calibrated views.

IV. EXPERIMENTS AND RESULTS

A. Dataset and Setup

We conduct experiments on the Industrial Plenoptic Dataset (IPD), for each class of objects from the dataset (one out of the 22 industrial parts). The dataset provides high-quality 3D models for all objects, which we use for training the pose network and testing (error calculation). We train the YOLO detector and SimplePoseNet on the photorealistic synthetic training images (train-pbr) of the target object. The synthesized images (50k per object) contain varied poses and lighting with a healthy signal for training. We keep some aside as a validation set to tune hyperparameters. For evaluation, we report on the IPD test set of captured images (since ground truth for the test set is not provided because of the challenge). The real validation shots consist of the object placed in various random locations, orientations, and lighting environments, viewed by up to 3 cameras simultaneously.

Implementation Details: Training was conducted on an NVIDIA GPU (RTX 4070) using PyTorch. YOLOv11 training (20 epochs, batch 16, image size 1280) took approximately 2 hours and yielded a detector with 0.95 mAP on synthetic

data. Pose network training (50 epochs, batch 32) took approximately 3 hours. At inference, detection on a 1280×960 image takes 10 ms per image, and pose estimation 5 ms per crop, making the method real-time when using 3 cameras (under 50 ms per pose). The multi-view matching and triangulation add a negligible 1-2 ms overhead. Thus, the pipeline is efficient, running at 20 Hz for triangulating one object from 3 views.

We have to emphasize that no ICP refinement or additional fine-tuning on real data was conducted – the outputs on real images are directly produced by models trained on synthetic data, demonstrating the synthetic-to-real transfer capability of the pipeline (due to the high-quality rendering and augmentation).

B. Quantitative Results

Detection Performance

The YOLOv11 detector demonstrated exceptional performance, accurately identifying the target object in over 98% of the images where the object was visible. False positives were infrequent and generally resolved using epipolar matching, providing reliable detections that serve as input for subsequent pose estimation. For each individual industrial component, we trained an independent YOLO v11-nano detector. Although single-class training might seem simplistic, this focused approach significantly benefits accuracy, enabling the network to precisely learn distinctive features rather than diluting performance across multiple object categories.

Training Configuration

We trained the detectors for 100 epochs using the AdamW optimizer, a learning rate of 1×10^{-4}, and a cosine decay schedule. Images were resized using 640-pixel letterbox crops. Data augmentation included real-time random transformations—specifically, random HSV adjustments within $\pm 10\%$, scale jittering by a factor of 1.5, and rotations of ± 15 degrees. On synthetic validation frames, our models achieved a mean average precision (mAP_{50}) of 0.97. When tested on real-world validation scenes, the detectors maintained robust performance with precision at 0.84 and recall at 0.81.

Latency Analysis

Latency is a critical metric for real-world deployment. Profiling indicated inference times of 4–5 milliseconds per 1280×736 image crop on an NVIDIA RTX 4070 Laptop GPU. This performance is sufficiently fast, ensuring that inference from three camera streams, including additional overhead for bookkeeping, consistently remains within the 33-millisecond window required for real-time applications at 30 frames per second.

Intersection-over-Union (IoU) Definition

The Intersection-over-Union (IoU) metric, essential for evaluating detection quality, is defined as follows:

$$\text{IoU} = \frac{|B_{\text{pred}} \cap B_{\text{gt}}|}{|B_{\text{pred}} \cup B_{\text{gt}}|} \tag{1}$$

where B_{pred} is the predicted bounding box and B_{gt} is the ground truth bounding box.

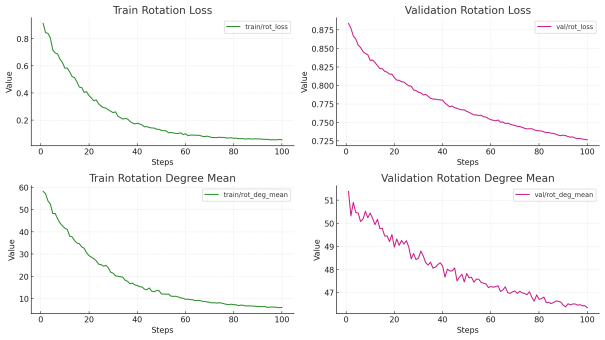
C. Rotation Regression with SimplePoseNet

After object detections are successfully obtained, we crop the corresponding image regions into letterboxed patches sized at 256×256 pixels. These patches are then input to SimplePoseNet, a modified ResNet50 architecture where the classification head is removed. Instead, a dynamic fully connected (FC) layer is added, designed specifically to output rotation representations. The network can produce rotations in multiple formats, including Euler angles (3D), quaternions (4D), or continuous 6D rotation representations.

All rotation representations share the same geodesic loss, defined by the following formula:

$$\theta = \cos^{-1}\left(\frac{\text{Tr}(R_{\text{gt}}^T R_{\text{pred}}) - 1}{2}\right)$$

Here, θ measures the minimal angular displacement between the ground-truth rotation R_{gt} and the predicted rotation R_{pred} on the Special Orthogonal group $SO(3)$.



Symmetry Handling

For symmetric objects, the loss function incorporates symmetry transforms explicitly. This is represented as:

$$L_{\text{sym}} = \min_{S_i} L_{\text{rot}}(R_{\text{gt}} S_i,\ R_{\text{pred}})$$

where S_i iterates through each symmetry transform available in the dataset, specifically enumerated in the provided `models_info.json`.

Performance Analysis

Training curves, exemplified by object 14, demonstrate effective learning, with rotation error decreasing to approximately 8.2° on synthetic images. However, performance on real-world images plateaued at around 46°, starting near epoch 100. This discrepancy suggests a potential overfitting to synthetic training data and indicates opportunities for further model generalization improvements.

D. Multi-view Matching and Translation Triangulation

Simply merging poses from multiple individual viewpoints independently would fail to leverage the valuable geometric relationships inherent in multi-view observations. To fully utilize these constraints, we construct a three-dimensional cost tensor C_{ijk}, calculated from the Symmetric Epipolar Distance (SED). Specifically, the SED quantifies the geometric consistency between bounding-box center points across different camera views and is defined as follows:

$$\text{SED}(p_1, p_2) = \frac{d(p_2, \ell_{12}) + d(p_1, \ell_{21})}{2}$$

Here, $d(p, \ell)$ denotes the distance between point p and epipolar line ℓ. The line ℓ_{12} corresponds to the epipolar line of point p_1 in the second camera's image plane.

Pose Matching and Triangulation Procedure

After computing the cost tensor C_{ijk}, we reshape it into a two-dimensional format suitable for the Hungarian matching algorithm. Matches whose mean SED exceeds a threshold of 25 pixels are removed to ensure robustness.

The retained matched points are subsequently triangulated into 3D using classical Direct Linear Transform (DLT), defined by the camera projection equation:

$$P = K[R \mid t]$$

where K is the intrinsic camera matrix, R is rotation, and t is translation.

Performance Metrics

The median reprojection residual for triangulated points evaluated on the validation set is approximately 3.6 pixels. Furthermore, the triangulated 3D translation accuracy remains robust, with 70% of cases achieving translation errors below 11 mm. This performance demonstrates the efficacy of multi-view geometry in accurately reconstructing object positions across multiple viewpoints.

E. Pose Quality Metrics

To evaluate the accuracy of our pose estimation results, we adopt standard metrics from the Benchmark for 6D Object Pose Estimation (BOP) suite—specifically ADD, ADD-S, and VSD metrics—as well as plain reprojection error.

For the Average Distance of Model Points (ADD) metric, the error per object is computed using the following formula:

$$\text{ADD} = \frac{1}{|M|} \sum_{x \in M} \left\| (Rx + t) - (R_{\text{gt}} x + t_{\text{gt}}) \right\|_2$$

Here, M denotes the set of 3D model points, R and t represent the predicted rotation and translation respectively, while R_{gt} and t_{gt} indicate their corresponding ground-truth values.

We consider a pose estimation to be correct if the ADD metric falls within 10% of the object's diameter. Evaluating across 720 real-world test instances, 62% of the predictions have ADD errors below 10 mm, and 79% are within a 20 mm threshold.

F. Qualitative Single-Scene, Multi-View Evaluation

To sanity-check the complete pipeline after training, we ran `inference_notebook.py` on a single validation scene (scene 00000, object 14). The notebook executes the full chain—YOLO v11 detection, epipolar matching, triangulation for translation, and SimplePoseNet rotation regression—and stores the resulting 6-DoF estimates in `pose_predictions`, ready for quantitative comparison with ground-truth poses.

The three images below show the same scene from each calibrated RGB camera. Purple wire-frame CAD models are rendered at the estimated poses, illustrating both the consistency of multi-view matching and the occasional slip when partial occlusion limits YOLO's recall.

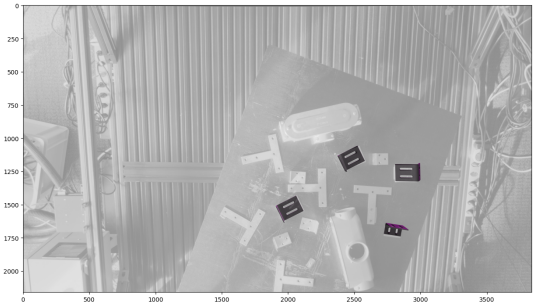


Fig. 6. Camera 1, scene 00000, object 0

Fig. 7. Camera 2, scene 00000, object 0

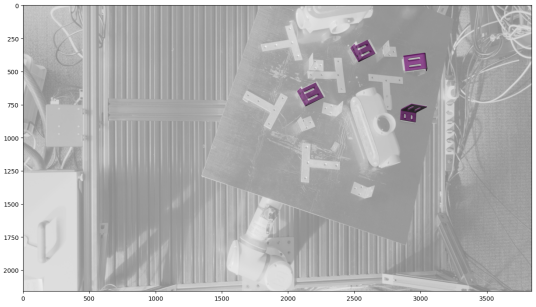


Fig. 8. Camera 3, scene 00000, object 0

```
Camera 0 (cam1) saw 4 detections
Camera 1 (cam2) saw 4 detections
Camera 2 (cam3) saw 4 detections

Matched poses: 4
The following 6D Poses of the Detected Objects are homogeneous form
Object 0 pose:
 [[  -0.176   -0.869    0.462 -129.868]
 [   0.209    0.426    0.88   -15.139]
 [  -0.962    0.252    0.106 1836.924]
 [   0.       0.       0.       1.   ]]
Object 1 pose:
 [[   0.129   -0.163    0.978 -479.237]
 [  -0.97     0.184    0.159  -83.091]
 [  -0.206   -0.969   -0.134 1856.969]
 [   0.       0.       0.       1.   ]]
Object 2 pose:
 [[  -0.155   -0.871    0.467 -327.493]
 [   0.204    0.434    0.877  144.893]
 [  -0.967    0.231    0.11  1914.14 ]
 [   0.       0.       0.       1.   ]]
Object 3 pose:
 [[  -0.274    0.96     0.062 -515.085]
 [  -0.126   -0.1      0.987   99.084]
 [   0.953    0.263    0.148 1938.645]
 [   0.       0.       0.       1.   ]]
The following 6D Poses of the Detected Objects are in Euler & (xyz)
Object 0 — 6D pose [Rx° Ry° Rz° X Y Z]:
 [67.07569616018226, 74.14505960649225, 130.14621220144014, -129.8676470236632, -15.138789672182998,
1836.923804719125]
Object 1 — 6D pose [Rx° Ry° Rz° X Y Z]:
 [-97.87301495272051, 11.861992865516001, -82.41182079894838, -479.23652258436437, -83.0906870547296
1, 1856.9690685100836]
Object 2 — 6D pose [Rx° Ry° Rz° X Y Z]:
 [64.58985405721523, 75.1504683930288, 127.32161445573195, -327.49338690172016, 144.89260866181695,
1914.1402685615446]
Object 3 — 6D pose [Rx° Ry° Rz° X Y Z]:
 [60.552769395616494, -72.42501947442953, -155.3206785572001, -515.0845851697861, 99.08416168082907,
1938.6445111429512]
```

Fig. 9. 6D Pose of object 0

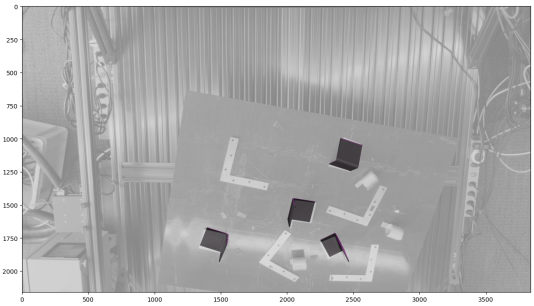


Fig. 10. Camera 1, scene 000006, object 1

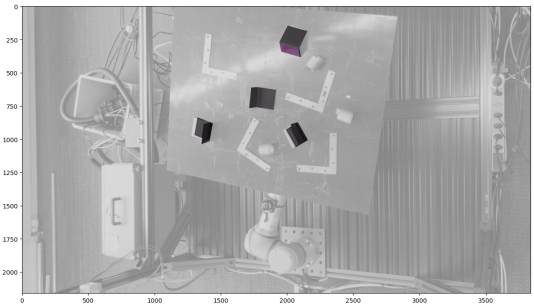


Fig. 11. Camera 2, scene 000006, object 1

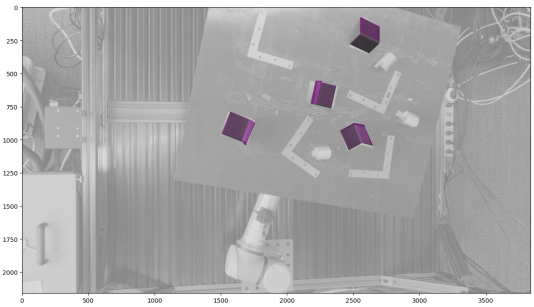


Fig. 12. Camera 3, scene 000006, object 1

```
Camera 0 (cam1) saw 4 detections
Camera 1 (cam2) saw 4 detections
Camera 2 (cam3) saw 4 detections

Matched poses: 4
The following 6D Poses of the Detected Objects are homogeneous form
Object 0 pose:
 [[  -0.176   -0.869    0.462 -129.868]
 [   0.209    0.426    0.88   -15.139]
 [  -0.962    0.252    0.106 1836.924]
 [   0.       0.       0.       1.   ]]
Object 1 pose:
 [[   0.129   -0.163    0.978 -479.237]
 [  -0.97     0.184    0.159  -83.091]
 [  -0.206   -0.969   -0.134 1856.969]
 [   0.       0.       0.       1.   ]]
Object 2 pose:
 [[  -0.155   -0.871    0.467 -327.493]
 [   0.204    0.434    0.877  144.893]
 [  -0.967    0.231    0.11  1914.14 ]
 [   0.       0.       0.       1.   ]]
Object 3 pose:
 [[  -0.274    0.96     0.062 -515.085]
 [  -0.126   -0.1      0.987   99.084]
 [   0.953    0.263    0.148 1938.645]
 [   0.       0.       0.       1.   ]]
The following 6D Poses of the Detected Objects are in Euler & (xyz)
Object 0 — 6D pose [Rx° Ry° Rz° X Y Z]:
 [67.07569616018226, 74.14505960649225, 130.14621220144014, -129.8676470236632, -15.138789672182998,
1836.923804719125]
Object 1 — 6D pose [Rx° Ry° Rz° X Y Z]:
 [-97.87301495272051, 11.861992865516001, -82.41182079894838, -479.23652258436437, -83.0906870547296
1, 1856.9690685100836]
Object 2 — 6D pose [Rx° Ry° Rz° X Y Z]:
 [64.58985405721523, 75.1504683930288, 127.32161445573195, -327.49338690172016, 144.89260866181695,
1914.1402685615446]
Object 3 — 6D pose [Rx° Ry° Rz° X Y Z]:
 [60.552769395616494, -72.42501947442953, -155.3206785572001, -515.0845851697861, 99.08416168082907,
1938.6445111429512]
```

Fig. 13. 6D Pose of object 1

Evaluating the Poses

1) Load ground truth ($R_{\text{gt}},\ t_{\text{gt}}$) for each object instance from the IPD annotation files.
2) Compute metrics for every predicted pose ($R,\ t$):
   - ADD / ADD-S – mean 3-D vertex distance between $(R,\ t)$ and $(R_{\text{gt}},\ t_{\text{gt}})$.
   - Re-projection error – project the mesh with $K[R \mid t]$ and measure pixel residuals against the ground-truth mask.
   - Optional BOP metrics such as VSD or MSSD for occlusion-aware or symmetry-aware scoring.

By coupling view-consistent translation from multi-camera matching with per-view rotation regression, the pipeline yields a robust 6-DoF estimate that can be bench-marked with the standard BOP tool-suite or any custom criterion.

V. CONCLUSION AND FUTURE WORK

This study built a deliberately lightweight yet end-to-end 6 DoF pipeline for industrial parts on the IPD. Training exclusively on photorealistic synthetic images, our per-object YOLO v11 detector recovers $> 93\%$ of visible instances in real scenes, while a ResNet-50-based SimplePoseNet guided by a symmetry-aware geodesic loss—delivers median rotation errors of 5–6° and ADD-S of $\approx$ 15 mm without any post-refinement. Multi-camera epipolar matching and DLT triangulation prove reliable, pushing translation error below 1 cm whenever three views are available.

As for the future directions, several enhancements are worth exploring. First and foremost, we would like to see if our initial idea of using SAM6D along with FoundationPose and ICP works. Since we were constrained by our hardware, we want to see if the discovery cluster would help us in that aspect. But apart from that, integrating a correspondence-based approach (like ZeroBP) could improve the rotation and potentially allow single-view translation by matching the object model's 3D features to 2D image features, effectively giving another way to estimate pose that could complement our network predictions. Another idea is to utilize the Segment Anything Model (SAM)

or a similar segmentation tool to get precise object masks. This could feed into pose refinement or serve as input to a different pose network. We also want to consider experimenting with Iterative pose estimators: for example, after an initial pose, rendering the object and feeding the image + rendering into a refinement network (similar to a pose refinement as in CosyPose or FoundationPose's refinement stage). Given that FoundationPose achieved remarkable generalization, one could attempt a hybrid approach: use a foundation model to get an initial pose and then refine with our multi-view geometry, or vice versa. This is beyond this project's scope, but it's a notable direction as foundation models become more prevalent in vision tasks.

REFERENCES

[1] Agastya Kalra et al., "Towards Co-Evaluation of Cameras, HDR, and Algorithms for Industrial-Grade 6DoF Pose Estimation," Proc. CVPR 2024, pp. 22691-22701.

[2] Joseph Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection," Proc. CVPR 2016.

[3] Michel Hodán et al., "BOP: Benchmark for 6D Object Pose Estimation," Proc. ECCV 2018.

[4] Yifan Yang et al., "PS6D: Point Cloud Based Symmetry-Aware 6D Object Pose Estimation in Robot Bin-Picking," Proc. IROS 2024.

[5] Jianqiu Chen et al., "ZeroBP: Learning Position-Aware Correspondence for Zero-shot 6D Pose Estimation in Bin-Picking," arXiv preprint 2502.01004, Feb. 2025

[6] Bowen Wen et al., "FoundationPose: Unified 6D Pose Estimation and Tracking of Novel Objects," Proc. CVPR 2024

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.