6D Pose Estimation for Glass and Transparent Objects

Samuel Klemic

Abstract – **This work investigates the task of 6D pose estimation for transparent and metallic objects. Current methods often fail due to the material properties of these objects, such as transparency, reflection, and distortion. Building on recent advances by ContourPose [7], this work extends its strategy of using a CNN to regress 2D key points and contours for metal object pose estimation, by applying it to a new dataset of transparent glass objects called ClearPose [4]. The proposed method involves generating ground truth contours using Blender, then training a CNN to predict object key points and contours from RGB images. By iteratively comparing and refining a predicted pose against the object contour, the approach demonstrates its potential in overcoming the limitations of conventional pose estimation techniques.**

I. Introduction:

6D Pose estimation is an increasingly developing task, providing crucial and accurate information to a variety of fields, including robotic grasping, manufacturing, augmented reality and more. Throughout recent years, a number of application areas where lightweight, accurate, and efficient 6D object pose estimation have become a necessity. As such, there has been much development in the field investigating various strategies for robust pose estimation. However, one particular challenge is accurately estimating the pose of objects with outlying material properties like transparency and reflection [6].

As discussed by [7], many current successful pose estimation methods rely on surface and texture features, or depth imaging in order to establish where and how an object is positioned. In the case of metallic and transparent objects, material properties such as distortion, transparency, and specular reflection pose complications with using these methods. Particularly in the case of depth imaging, these properties lead to fragmented depth data, not suitable for pose predictions [12]. As such, attempting pose estimation of non-Lambertian objects is limited to only use RGB-based methods. Utilizing known texture features also pose issues for metallic and transparent objects. These objects tend to lack distinctive surface and texture features, often dependent on the environment around them due to reflection and/or transparency. As such, strategies which identify key points based on known-features are prone to inaccuracy when tested on metallic or transparent objects.

Recent success has been found by ContourPose [7] in surpassing these challenges, by utilizing a convolutional neural network (CNN) to regress both 2D key points as well as the contour outline on an object. Given these references, an initial pose can be estimated by solving the Perspective-n-Point (PnP) using the regressed keypoints, and iteratively comparing that pose against the regressed contour until an optimal set of keypoints and subsequently, an optimal pose prediction is found. This process is coined by [7] as Pose Evaluation using Contour as a Priori (PECP).

An object contour provides a stable geometric reference, because it is consistent regardless of lighting conditions or background objects, so distortion, reflection, and transparency are less likely to affect the final prediction. ContourPose [7] was applied to a dataset of industrial metallic parts, however this work investigates the effectiveness of its strategy on a dataset of glass and transparent objects; ClearPose [4].

II. Related Work:

Modern pose estimation strategies tend to fall into two categories: Direct, learning-based approaches, or indirect, non-learning-based approaches [6][10]. The latter often establish 2D-3D correspondences, either from only RGB input or also using depth data. These correspondences are used to estimate a 6D pose using a RANSAC-based PnP algorithm. With indirect methods, such as this, the specific correspondences chosen greatly impact the final pose estimation, and they tend to vary in effectiveness depending on the pose [7]. An alternative indirect method is template matching, which discretizes rotation into known poses, and finds a best match based on a pre-existing database.

Direct methods, on the other hand, tend to use CNN and regression to directly output the pose prediction given an input image, such as GDR-Net [10]. However, some approaches, like the one utilized in ContourPose [7] and this work, regress 2D-3D correspondences and then solve the PnP. Completely direct approaches are limited to the feature extraction abilities of CNNs, which, without masking or other constraints, tend to be less effective towards transparent and metal parts. Alternatively, incorporating depth and point cloud data into the input for direct methods increase their effectiveness, but the material properties of non-Laberian objects prevent accurate depth data from being collected [1][6]. Some implementations, such as Pix2Pose [8], regress the position of each individual pixel within the object's segmentation. However transparency and reflection with glass and metallic objects lead to variability in the specific color of a given pixel for the object, and inaccuracies with this method.

ContourPose's approach to regress sparse keypoints and a contour prediction based on a single RGB image demonstrated promising results for estimating the pose of reflective objects.

III. Proposed Approach:

In this work, the 6D pose estimation strategy utilized in ContourPose [7] is applied to a dataset of everyday, transparent, glass objects [4]. 6D pose estimation seeks to predict the 3D rotation and translation of an object in physical space. The final output of 6D pose estimation is a transformation matrix, $[R|t]$, which transforms the object's coordinate system to the camera's coordinate system.

Firstly, ground truth contours were rendered using Blender and an annotation tool, Progress Labeller [3]. Key points for each object within the ClearPose [4] dataset were determined using a farthest point sampling (FPS) method. Following the strategy of [7], for a specific object in the dataset, a CNN was trained on the RGB images from Clearpose to output both a heatmap predicting the location of the object's key points, and the contour of the object. Following the output of the CNN, the pose of the object can be iteratively estimated using PECP. A summary of this process can be seen in Fig. 1.

A. Annotation Process

ClearPose [4] provides a number of scenes with a variety of household objects, along with details about which objects are visible in which images, object poses, and camera intrinsics. In order to train the contour decoder end of the network, ground truth contours for the objects needed to be generated. In order to do so, the same annotation tool used to annotate Clearpose [4] was employed. Progress Labeller

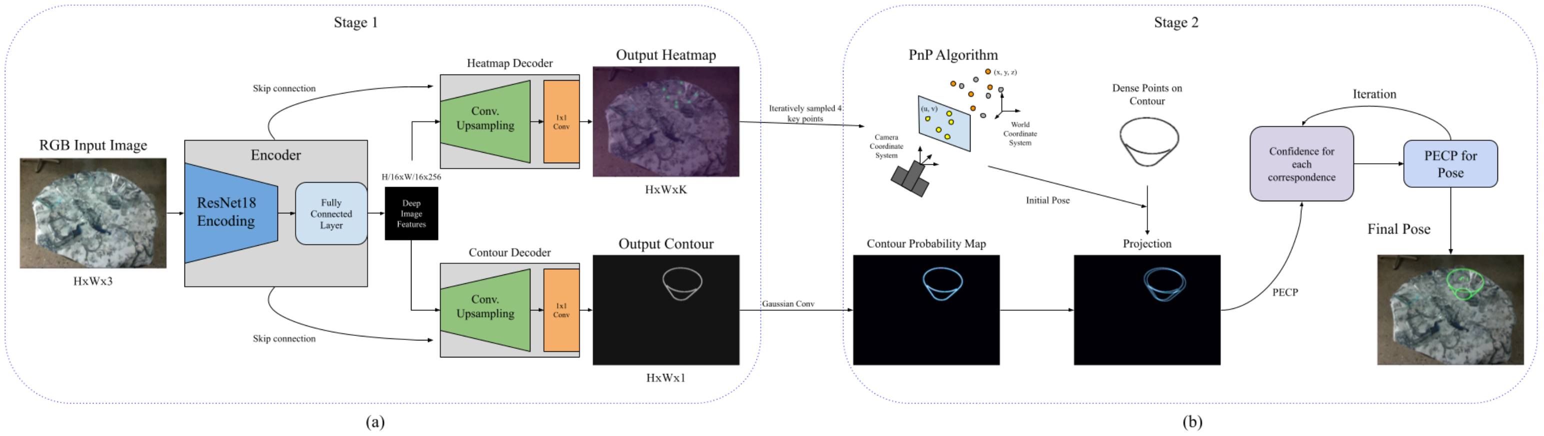


Fig 1. Overview of methodology, using a two stage process. (a) Keypoints and object contour are predicted from single RGB image using dual-encoder CNN. (b) Pose is estimated using PECP, iteratively comparing a predicted pose against the contour generated in Stage 1.

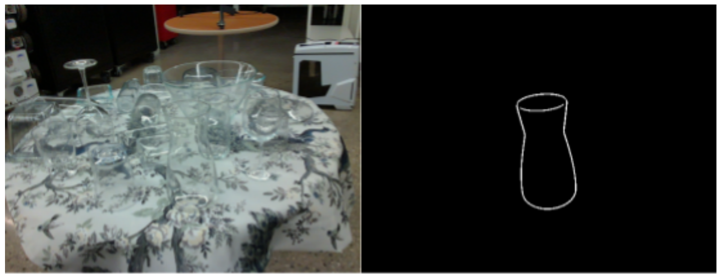


Fig 2. Example contour rendered in Blender using ProgressLabeller[3] and custom contour material

[3] is an addon for Blender which utilizes ORB-SLAM reconstruction to create a 3D space based on a dataset of RGB-D images. It also places cameras in the appropriate perspectives corresponding to each of the input images from the dataset. This was employed on ClearPose in order to annotate the scenes used in experimentation, placing each object model in the correct spot within 3D space. Once a scene was annotated, a custom material was applied to each object such that only the contour would be output in a render, thus creating a ground truth contour with which to regress the CNN.

B. Keypoint and Contour Prediction Network

Following the setup of ContourPose [7], the network features a modified encoder/decoder structure, in which a single encoder feeds its output into two separate decoders. One decoder outputs a heatmap predicting the location of K keypoints, and the other outputs a single channel image predicting the location of the object contour. [7] opted to incorporate this dual-decoder architecture in order to segregate the keypoint contour prediction. Their ablation studies found that outputting both predictions from a single decoder leads to the prediction of keypoints being heavily constrained by the success of the contour prediction, and vice versa. Comparatively, separating them prevents erroneous dependance between the predictions. Additionally, utilizing a single encoder for both decoders creates an implicit relationship between the two decoders' output without their predictions being directly entangled. In this way, [7] does not need a predetector to find the object in an input image, like [10], but instead cotrains the encoder with both decoders' output. As a result, the deep image features encoded implicitly provide the location of the object as is necessary to generate the contour. Likewise, the keypoints are restrained to a similar location.

Regarding the specifics of the network, the encoder downsamples an input H x W x 3 RGB image to an H/16 x W/16 x 256 feature map. The encoder uses a ResNet-18 architecture, and as such, features skip connections connecting corresponding down and upsampling layers between itself and both of the decoders. After downsampling, a layer of dilated convolutions is applied. Upsampling occurs in both of the decoders, the heatmap decoder outputs a H x W x K heatmap of the predicted location for K keypoints, while the contour decoder outputs a H x W x 1 prediction of the object contour.

Because of the dual decoder design of the network, there are two regressions with which to train the model: the regression of the heatmap, denoted by **Φ**, and the regression of the contour, denoted by **Γ**. For the regression of the heatmap a mean square error loss function is employed.

$$l(w_E, w_{HD}) = \sum_{k=1}^{K} l_2(H_k(w_E, w_{HD}) - \overline{H}_k) \qquad (1)$$

$$H_k = \Phi_{w_E, w_{HD}}(I) \qquad (2)$$

Where w_E represents the parameters of the encoder, and w_{HD} represents the parameters of the heatmap decoder. I represents the input image, K represents the total number of keypoints for the particular object being trained. $\overline{H}_k$ represents the ground truth heatmap for the kth keypoint and H_k represents the predicted heatmap.

For the regression of the contour prediction, ContourPose [7] discusses the issue of category imbalance between the positive and negative points in the output tensor. Particularly, because the actual contour makes up for so little relative area in the H x W output, they utilized a weighted cross entropy function.

$$l(w_E, w_{CD}) =- \beta \sum_{p \in \overline{Y}^+} log(Y_p = 1; w_E, w_{CD}) - (1 - \beta) \sum_{p \in \overline{Y}^-} log(Y_p = 0; w_E, w_{CD}) \qquad (3)$$

$$Y = \Gamma_{w_E, w_{CD}}(I) \qquad (4)$$

Where w_{CD} represents the parameters of the contour decoder, Y is the predicted contour, and p denotes each pixel in the H x W output. β represents the ratio of negative points in the ground truth contour, $\beta = |\overline{Y}^-| / |\overline{Y}^+ + \overline{Y}^-|$ and thus $1 - \beta = |\overline{Y}^+| / |\overline{Y}^+ + \overline{Y}^-|$. Given these two loss functions, the final loss of the network can be expressed in (5).

$$l(w) = l(w_{E'}, w_{HD}) + \rho l(w_{E'}, w_{CD}) \tag{5}$$

With w as all parameters of the network, and hyperparameter ρ to balance the magnitude of the two loss functions.

C. Keypoint Selection

Unlike the metal industrial parts of ContourPose [7], the models within ClearPose [4] do not feature defined edges or many circle centers. As such, using semantic point detection algorithms would not prove effective towards the largely curved and smooth glass objects of this work's dataset. In lieu of these sharp features, key points were selected semantically using an FPS method on the point clouds of each object, similar to PVNet [9].

D. Pose Estimation Using Contour as Geometric Priors

Once keypoints and a contour are predicted by the network in stage 1, the pose of the object can be estimated in stage 2. ContourPose [7] outlines a novel approach to this, PECP. Firstly, an initial pose T is calculated by solving the PnP with 4 randomly sampled points from the K key points predicted in stage 1. Previously, dense points along the contour of each model were sampled. These points, S, are then transformed by the initial pose T and the camera intrinsic matrix k in order to obtain the projection, P, of the dense contour points within the camera coordinate system:

$$P = k[R|t]S \tag{6}$$

$$\kappa = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

Once this projection is calculated, the accuracy of the pose can be determined by finding its overlap with the predicted contour of the object. Specifically, a Gaussian convolution function is applied to the predicted contour and the projection, where $\sigma = 9$ and the kernel size is 3. This creates a probability map, scoring each projected point with how close it lies to the contour. The final score is then the sum of all projected points.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{8}$$

$$H(u, v) = \sum_x \sum_y G(x, y) Y(u - x, v - y) \tag{9}$$

Once a numerical score is found for the pose, this score is then assigned to the 4 key points sampled to calculate the initial pose T. This process is repeated for I iterations, overwriting previous confidence scores assigned to a keypoint, if a higher one is found in a future iteration. After I iterations, the 4 highest scoring keypoints are used to calculate a new pose, which is scored again using PECP, denoted as M. The next highest scoring point is then added to the PnP calculation, with the resulting pose's PECP score denoted as Q. If $Q > M$, then this point is added alongside the initial 4 to the final set of keypoints used in pose calculation. Otherwise this point is discarded. This process continues for all key points predicted for stage 1, until an optimal set of keypoints to use is determined, and a final pose is calculated.

E. Dense Contour Point Sampling

In order to obtain the dense points along the contour of each object, each point within the point cloud was scored for how often it lies on the edge of that object's projection within the dataset. For each image pose, the object's point cloud is projected, and the Canny edge detection algorithm is used to determine the edges of this projection. The number of times a particular point in the point cloud is projected onto this edge is counted. If this count exceeds a certain threshold, then that associated point is considered part of the contour and is used within the set of dense contour points. Generally, a threshold of 1/8th the total number of images sampled lead to a successful number and distribution of dense points, although manual review modified this threshold depending on the object.

IV. Experimental Setup:

A. Dataset

The dataset used in this work's experiments is ClearPose [4] along with the custom rendered contour images. ClearPose was selected because of its rich and well annotated data, along with a variety of glass and transparent objects to evaluate with. Specifically, scenes 1 and 3 from set 2 of Clearpose were annotated. This set contained a total of 27 transparent objects arranged on a table with a variable table cloth, depending on the scene. Scene 1 was annotated first as a proof of concept, and was sampled by ProgressLabeller at a rate of 0.1, meaning 1/10th of the RGB images have associated ground truth contours. Scene 3 on the other hand, was sampled at a rate of 1.0, meaning all images were annotated. For each object in the set, 821 ground truth contours rendered for scene 1 and 7,241 were rendered for scene 2, meaning a total of 8,062 images were used in training for each object. It's worth noting that this number varies slightly between each object, as some of the RGB images don't feature the object in frame, or it is too occluded to be visible. In these cases, this work referred to ClearPose's annotation to determine which RGB images actually feature which objects. 1/10th of these frames were stored as test data, while the remaining 9/10th were used from training.

Additionally, as a result of time constraints, only 3 of the 27 objects featured in the set were trained. In particular, "bottle_1," "bottle_4," and "bowl_5".

“Bottle_1” was selected because of its unique shape and the fact that it was subject to occlusion throughout both scenes, whilst remaining visible in frame for nearly every image. In this way, results could be obtained for how well the network can predict contours with only a section of the object being visible.

“Bottle_4” was selected because of its relatively sharp edges, and rectangular shape. Because of these features, the contour is more consistently defined across different camera perspectives, ideally enabling the PECP process to more accurately estimate the pose.

“Bowl_5” was selected because it is the most prominent, least occluded object in both scenes, and as such could serve as a reference for the general performance of both the network and PECP process on a transparent object.

B. Training Strategy

Following the procedure of ContourPose [7], an initial learning rate of 0.1 was used, halving it at every 20 epochs. An individual model was trained for each of the three objects selected. The models were trained for 250 epochs total, using an AdamW optimizer, and trained on a single Nvidia A40 GPU. Weight decay was set to 0.1 and the hyperparameter ρ in the combined loss function was set to 100.

C. Metrics

The effectiveness of the pose estimation was evaluated using three metrics: a mean 2D projection metric, the average 3D distance of model points (ADD) metric, and the average rotation and translation error of the predicted pose against the ground truth.

2D Projection Metric: This metric finds the mean distance between the projection of the object’s 3D mesh in both the ground truth pose and the predicted pose. In particular if this distance is less than 5 pixels, the pose is considered correct.

ADD Metric: This metric finds the mean distance between the point cloud of the model at the ground truth and predicted poses. If this distance is less than 10% of the diameter of the object, then the pose is considered correct. The diameter of each object by finding the maximum distance between any two key points selected earlier using FPS.

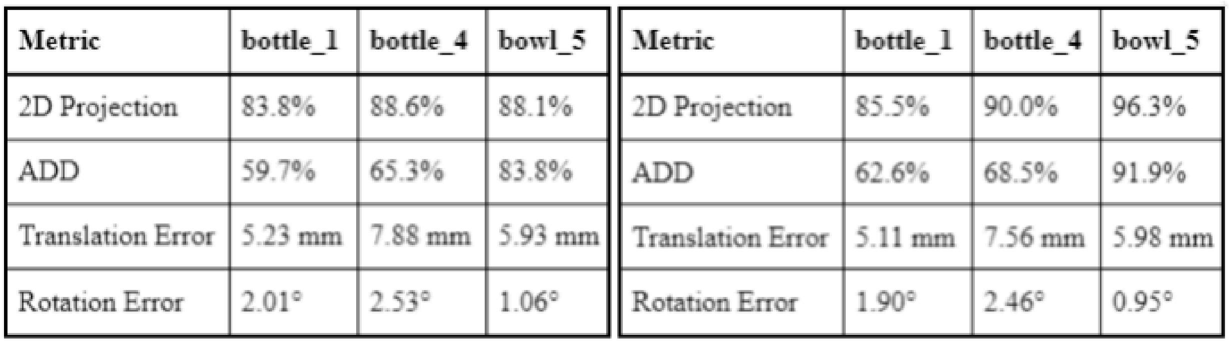
Rotation/Translation Error: Provided that a pose is correct according to the ADD metric, the rotation and translation error denotes the difference between the ground truth and predicted pose in terms of α, β, γ, in 3D rotation and x, y, z, in 3D translation.

V. Results:

A. Initial Results

Following training, the three models were evaluated using the discussed metrics on 1/10th of the images from the annotated ClearPose dataset. These results are visible in the Table I section (a). Despite the relative success of the “bowl_5” model, these results are generally suboptimal. One potential source of this accuracy is translational error of camera perspectives in the ORB-SLAM reconstruction of the

TABLE I.
Performance of models trained on data rendered by Blender
(a) Results from evaluating using both scenes. (b) Results from evaluating using only scene 3



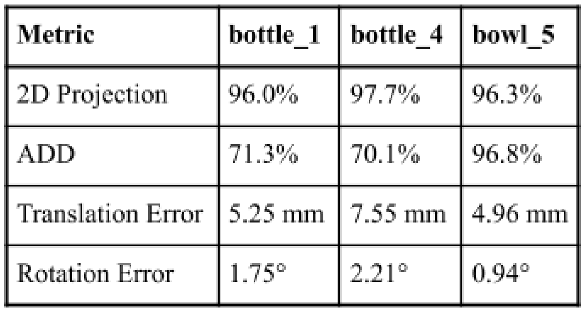
Metric	bottle_1	bottle_4	bowl_5
2D Projection	83.8%	88.6%	88.1%
ADD	59.7%	65.3%	83.8%
Translation Error	5.23 mm	7.88 mm	5.93 mm
Rotation Error	2.01°	2.53°	1.06°

(a)

Metric	bottle_1	bottle_4	bowl_5
2D Projection	85.5%	90.0%	96.3%
ADD	62.6%	68.5%	91.9%
Translation Error	5.11 mm	7.56 mm	5.98 mm
Rotation Error	1.90°	2.46°	0.95°

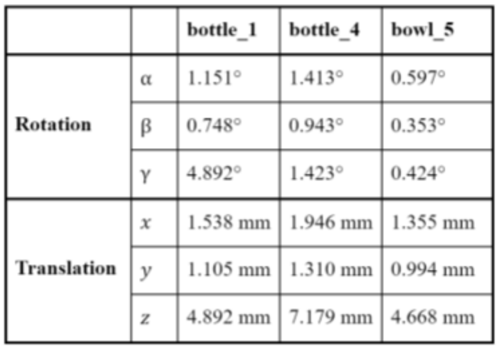
(b)

TABLE II.
Performance of models trained on projected densely sampled contour points



Metric	bottle_1	bottle_4	bowl_5
2D Projection	96.0%	97.7%	96.3%
ADD	71.3%	70.1%	96.8%
Translation Error	5.25 mm	7.55 mm	4.96 mm
Rotation Error	1.75°	2.21°	0.94°

TABLE III.
Error of each model with respect to each degree of freedom



		bottle_1	bottle_4	bowl_5
Rotation	α	1.151°	1.413°	0.597°
	β	0.748°	0.943°	0.353°
	γ	4.892°	1.423°	0.424°
Translation	x	1.538 mm	1.946 mm	1.355 mm
	y	1.105 mm	1.310 mm	0.994 mm
	z	4.892 mm	7.179 mm	4.668 mm

dataset while annotating in Blender. It was observed that not every camera was positioned correctly within the 3D environment. As a result, a significant portion of the rendered ground truth contours were inaccurate to the actual position and rotation of the object. “Bowl_5” being the most prominent and central object in both scenes, was least affected by these erroneous annotations, however, its model’s performance still suffered.

Another potential source for error is overtraining towards scene 3. Given that there are approximately 8.8 times more annotated images for scene 3, it is noteworthy that all models performed marginally better when evaluated on only test data from scene 3. Results from evaluating exclusively on scene 3 can be seen in Table I. section (b).

B. Results from Training on Densely Sampled Contours

In order to achieve better results, a different strategy was employed for generating the ground truth contours. Densely sampled points along the edges of ClearPose’s 3D models were already established in order to evaluate the accuracy of the pose estimation using PECP. These points projected in the correct

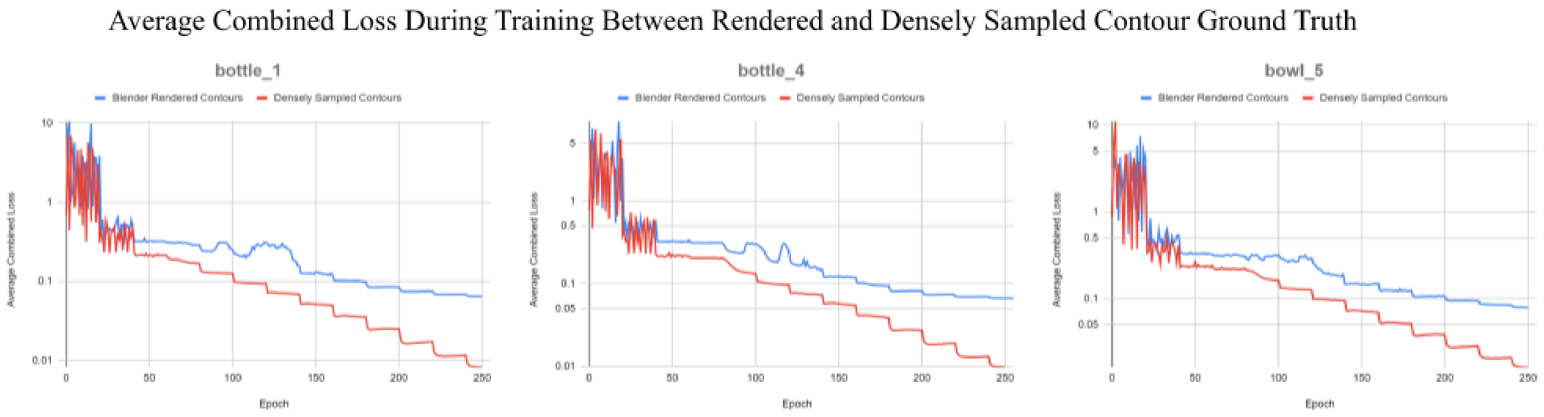


Fig 3. Average combined loss of each model per epoch during training. Blue line indicates the average loss of the models trained on contours rendered using Blender, while the Red line indicates the average loss of models trained using projected densely sampled contour points.

pose based on ClearPose's annotations, combined with the Canny detected edge of the overall point cloud projection can produce significantly more accurate contour ground truth images and the results can be seen in Table II.

Retraining each of the models on this new contour data produced significantly better results, in terms of the 2D project metric and ADD. One noteworthy aspect of this data is only a marginal improvement to the rotational and translational error in the predicted pose. This is likely because these errors were only measured in the case of a pose being correct according to the ADD metric. Thus, the poses which contributed to the average of these errors were ones actually correct in the first place, leading to a smaller discrepancy between the two sets of trained models.

C. Rotation and Translational Errors

Looking into the positional errors with respect to each degree of freedom (Table III), we see that the largest contributor to translational error is the error in the z direction. This direction corresponds to the distance away from the camera. This is likely a limitation of the monocular nature of this approach, as only a single perspective of the scene makes determining depth difficult. *ContourPose*'s results both with their own model, and testing other state-of-the-art methods demonstrate a similar trend, with the z direction having the largest error.

D. Training Results

One final noteworthy aspect of the compared results between the two annotation strategies, is that the models trained on densely sampled contour points experienced a more rapid decline in loss over their epochs compared to the original models. (Fig. 3) This is most likely because of a stronger correlation between the RGB inputs and the location of the ground truth contour. Comparatively, the contours rendered using blender had positional discrepancies to the actual object in the RGB input. As a result,

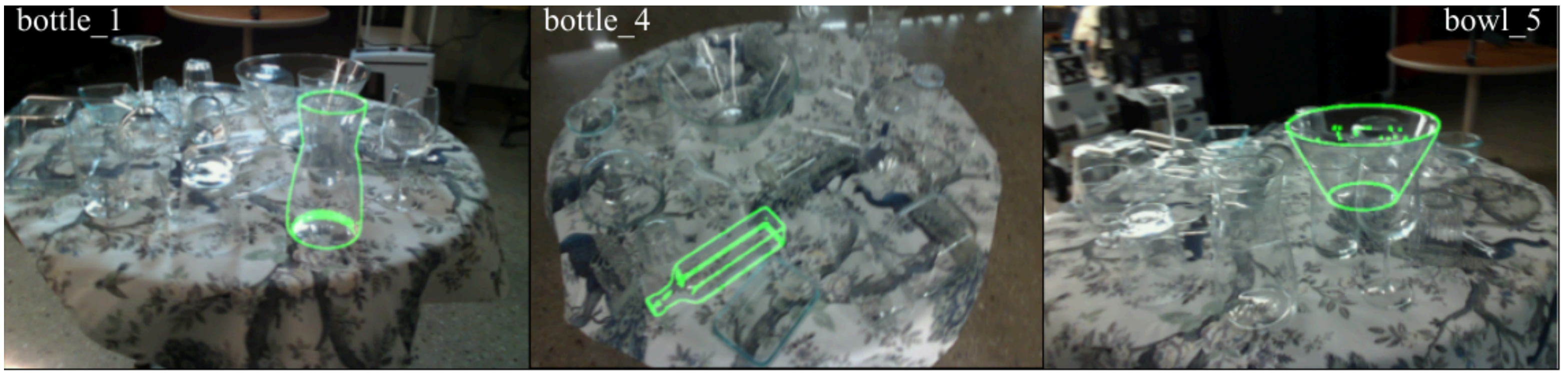


Fig 4. Qualitative results of the three models trained on densely sampled contour data. The green lines visualize the predicted pose of the object projected onto the input image

training was more difficult. Overall, learning both the contour, as well as the associated keypoints was more successful and efficient using densely sampled contour points rather than the rendered contour.

VI. Conclusions and Future Work:

Overall, the models trained in this work were successful at replicating accurate contour and keypoint predictions of transparent objects. However the pose estimation process utilized by ContourPose [7] did not produce exemplary results. The objects trained on by ContourPose [7] featured sharp edges, drilled holes, and far more defined features than that of ClearPose [4]. As a result, there is a far more positive correlation between a correct pose and the contour of these objects, compared to glass curved objects. Additionally, ContourPose [7] employed a randomized data strategy. In it, masked out objects were placed in a random background, sampled from SUN397 [11], at random scaling, rotation, and translation, in order to prevent overfitting.

Considering incorporating this randomized data strategy into this work's model, there is the limitation in the transparency of glass objects, as well as properly simulating reflection. As such, directly pasting a masked glass object onto a background may not account for how these properties affect the object's surface features, and hence the performance of the network. Ideally, for the task of a randomized dataset, synthetic data rendered by Blender or similar software could be used. Contemporary work on transparent object pose estimation [1] has found success using synthetic data, so testing the PECP method on it could lead to interesting results.

Further exploration should also include continuing experimentation on the remaining 24 objects from the annotated scenes. This could provide more comprehensive results and help determine specific use cases where the methodology of ContourPose [7] could be applied. Furthermore, rigorously testing alternative state-of-the-art strategies on ClearPose [4] would be necessary to properly gauge the success of this work's results. Recreating the ablation studies of *ContourPose* would also aid in drawing comprehensive conclusions about the effectiveness of the network structure. Regarding training, the results from the models trained on densely sampled points demonstrate that a more rapid learning rate decay – perhaps every 10 epochs instead of 20 – could decrease the amount of epochs necessary to achieve satisfactory results.

VII. References:

[1] Byambaa, M., Koutaki, G., & Choimaa, L. (2022). 6D Pose Estimation of Transparent Object From Single RGB Image for Robotic Manipulation. IEEE Access, 10, 114897–114906. https://doi.org/10.1109/ACCESS.2022.3217811

[2] Chen, C., & Jiang, X. (2024). Multi-View Metal Parts Pose Estimation Based on a Single Camera. Sensors, 24(11). https://doi.org/10.3390/s24113408

[3] Chen, X., Zhang, H., Yu, Z., Lewis, S., & Jenkins, O. C. (2022). ProgressLabeller: Visual Data Stream Annotation for Training Object-Centric 3D Perception. ArXiv Preprint ArXiv:2203.00283.

[4] Chen, X., Zhang, H., Yu, Z., Opipari, A., & Jenkins, O. C. (2022). ClearPose: Large-scale Transparent Object Dataset and Benchmark. European Conference on Computer Vision.

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385

[6] He, Z., Feng, W., Zhao, X., & Lv, Y. (2021). 6D Pose Estimation of Objects: Recent Technologies and Challenges. Applied Sciences, 11(1). https://doi.org/10.3390/app11010228

[7] He, Z., Li, Q., Zhao, X., Wang, J., Shen, H., Zhang, S., & Tan, J. (2023). ContourPose: Monocular 6-D Pose Estimation Method for Reflective Textureless Metal Parts. IEEE Transactions on Robotics, 39(5), 4037–4050. https://doi.org/10.1109/TRO.2023.3290300

[8] Park, K., Patten, T., & Vincze, M. (2019). Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation. CoRR, abs/1908.07433. http://arxiv.org/abs/1908.07433

[9] Peng, S., Liu, Y., Huang, Q., Bao, H., & Zhou, X. (2018). PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation. https://arxiv.org/abs/1812.11788

[10] Wang, G., Manhardt, F., Tombari, F., & Ji, X. (2021). GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation. CoRR, abs/2102.12145. https://arxiv.org/abs/2102.12145

[11] Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., & A. Torralba. (2010). SUN database: Large-scale scene recognition from abbey to zoo. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 3485–3492. https://doi.org/10.1109/CVPR.2010.5539970

[12] Zhang, H., & Cao, Q. (2019). Detect in RGB, Optimize in Edge: Accurate 6D Pose Estimation for Texture-less Industrial Parts. 2019 International Conference on Robotics and Automation (ICRA), 3486–3492. https://api.semanticscholar.org/CorpusID:199541985