2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

3D-DAT: 3D-Dataset Annotation Toolkit for Robotic Vision

Markus Suchi[1], Bernhard Neuberger[1], Amanzhol Salykov[1],
Jean-Baptiste Weibel[1], Timothy Patten[1,2], and Markus Vincze[1]

***Abstract*— Robots operating in the real world are expected to detect, classify, segment, and estimate the pose of objects to accomplish their task. Modern approaches using deep learning not only require large volumes of data but also pixel-accurate annotations in order to evaluate the performance and therefore safety of these algorithms. At present, publicly available tools for annotating data are scarce and those that are available rely on depth sensors, which excludes their use for transparent, metallic, and general non-Lambertian objects. To address this issue, we present a novel method for creating valuable datasets that can be used in these more difficult cases. Our key contribution is a purely RGB-based scene-level annotation approach that uses a neural radiance field-based method to automatically align objects. A set of user studies demonstrates the accuracy and speed of our approach over a purely manual or depth sensor assisted pipeline. We provide an open-source implementation of each component and a ROS-based recorder for capturing data with a eye-in-hand robot system. Code will be made available at `https://github.com/markus-suchi/3D-DAT`.**

I. Introduction

With the rise of machine learning methods in computer vision comes the need for large annotated datasets. It is crucial to gather accurately annotated data from the sensory hardware used by the robot system to verify methods on relevant data and thus guarantee the safety of the system.

Robot vision comprises a diverse set of tasks such as object segmentation, classification, and pose estimation, which are necessary for robotic capabilities like grasping, manipulating, and placing objects. Despite the importance of the vision system, the tools to gather fully annotated data for direct and rapid evaluation are unfortunately scarce.

Although many publicly available datasets exist and have significantly advanced the field [1]–[5] , there is still a domain gap to overcome every time vision methods are deployed on a real robot. Due to the requirement of large annotated datasets for training modern deep learning approaches, researchers have turned to the generation of synthetic data that is highly realistic or includes mechanisms to overcome the synthetic-to-real domain gap. The Benchmark for 6D Object Pose Estimation (BOP) [6] is the standard for benchmarking object pose estimation and it provides numerous synthetic training datasets to evaluate pose estimators on real test data. Methods trained only with synthetic data achieve good performance on specific datasets but the best results [7][8] still require refinement using real data. Therefore, it is important to have access to annotations of real-world data not only to robustly evaluate methods before deployment but also to train them most effectively.

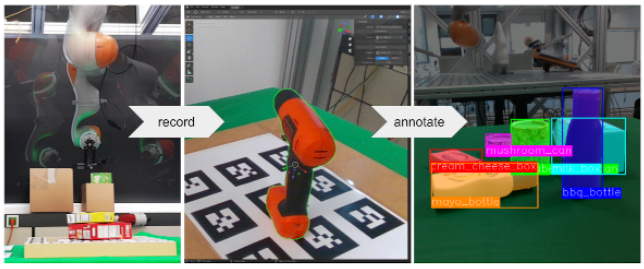


Fig. 1: 3D-DAT is a user-friendly semi-automatic annotation toolkit, utilizing NeRF-based scene reconstruction to enable efficient multi-view object pose and image labeling.

As such, annotating real data is unavoidable. Annotation is time-consuming for all vision tasks but is particularly difficult for the 6D object pose and there is a severe lack of easy-to-use tools. Furthermore, most popular and available tools [9][10] do not generalize to the case where objects return poor depth readings due to transparent or metallic surfaces [11][12] because they rely on 3D reconstructions made using depth sensors. Thus, these tools are unusable for a large variety of important real-world objects occurring in industrial [2] or household settings [13]

To overcome the limitations of prior work, we present 3D-DAT, a novel method for exact object pose annotation of RGB sequences. Our key innovation is the development of an automated annotation process with a fast NeRF-based (Neural Radiance Fields) alignment method that does not require a depth sensor (see Figure 1). 3D-DAT is implemented as a Blender[14] add-on, which enables a user to estimate the object pose by manipulating 3D template models to visually match them with their projection in recorded RGB images. After this quick manual coarse initialization of the object pose, it is refined automatically by applying the Iterative Closest Point (ICP) [15] algorithm with a sampled scene reconstruction and the 3D object model. In contrast to existing work that use the depth sensors for reconstruction [9][10], we produce reconstructions from RGB images only by employing NeRF [16] to generate depth images as in [12]. Consequently, this enables annotating difficult objects having transparent or shiny materials. Annotation is

[1]Vision for Robotics Laboratory, Automation and Control Institute, TU Wien, 1040 Vienna, Austria. {suchi, neuberger, weibel, patten, vincze}@acin.tuwien.ac.at, amanzhol.salykov@tuwien.ac.at

[2] Robotics Institute, University of Technology Sydney, Sydney, Australia. timothy.patten@uts.edu.au

This work is supported by the Austrian Science Foundation (FWF) and CHIST-ERA grant agreement No. I3967-N30 BURG, No. I3968-N30 HEAP, No. I3969-N30 InDex, and EC project No. 101017089 TraceBot. We are grateful for the financial support of Festo AG & Co. KG.

performed at the scene level so changes made in a single view are immediately propagated to the whole sequence of images and the user receives continuous visual feedback about the state of the current pose overlap. Once an object is aligned with the scene, pixel-wise segmentation annotation or bounding boxes are trivially obtained by projecting the object 3D models to the recorded views.

We demonstrate data collection using a robot arm with a mounted camera and use its kinematic chain to obtain the camera pose after calibration. This provides highly accurate camera poses similar to other works [4][17][18]. If data is available as ROS topics, our ROS recorder is employed to save the relevant data (i.e., RGB images, depth images, point clouds, camera poses). The resulting recordings are then immediately ready for processing with the annotation tools.

We perform user studies to demonstrate the accuracy and speed of the 3D-DAT annotation method. The studies include annotating synthetic scenes to provide accurate results using exact ground truth data. For real data, no ground truth exists, thus we calculate the deviation from the mean pose of all annotations to measure the similarity between them which is similar to the approach used by commercial products for bounding box detection annotation [19].

To summarize, our contributions are:

1) A GUI for manual annotation of multi-view image sequences using only RGB images.
2) A purely RGB-based semi-automatic alignment method using Neural Radiance Fields to refine coarse aligned objects to speed up the manual annotation process.
3) A set of user studies to evaluate the accuracy and speed of our annotation tool.
4) The implementation of the annotation approach as a set of open-source tools.

We demonstrate that our NeRF-based approach on real data results in an annotators' agreement level comparable to that obtained with the manual annotation of synthetic scenes with perfect camera poses and parameters. The results of our experiments also show that using NeRF outperforms using depth sensor readings for auto-aligning objects and has a significant positive impact on annotation speed.

II. Related Work

The availability of datasets and tools for creating labeled data has drastically improved many tasks in computer and robot vision. Object segmentation, classification and detection datasets such as COCO [20], PASCAL [21], and ImageNet [22] play a significant role in developing and benchmarking state-of-the-art algorithms. For annotating single images, tools like LabelMe [23], COCO-FiftyOne [24] , DEXTR [25] and PolygonRCNN [26] are freely available.

In robotics, 3D annotated data and especially annotation of 6D object poses are highly valuable to evaluate tasks involving grasping and manipulation. The OCRTOC [27] benchmark for robot grasping and manipulation provides labeled data for RGBD images for their cloud challenge contests. DEXYCB [28] provides annotations for objects and hands using a fixed camera setup and manual labeling of keypoints. The YCB-video dataset [5] provides annotated RGB and depth video sequences of scenes with YCB [29] objects. The pose of each object is manually annotated in the first depth frame, refined using the Signed Distance Function then tracked and propagated throughout the recording.

Monica et al. [30] develop an interactive editor for 3D point cloud labeling. SemanticPaint [31] enables a user to physically annotate 3D reconstructions through virtual reality. Our previous work, EasyLabel [32], uses depth differencing to retrieve object labels from point clouds. The downside of these methods is the lack of object pose annotation and the dependency on the quality of depth information.

SALT [33] uses 2D bounding boxes placed by the user to identify objects in a video stream. The boxes are propagated and 3D bounding boxes are derived from the manual input. Foreground and background are separated semi-automatically using Gaussian Mixture Models. Pixel-wise masks are refined using GrabCut [34], requiring additional manual annotations in the form of scribbles. LabelFusion [9] uses ElasticFusion [35] for camera pose estimation and reconstruction. The user manually annotates corresponding keypoints on 3D models and the reconstruction to retrieve an initial model alignment. ICP is then used to refine the alignment and object masks can be retrieved by back projecting the model to the RGB images. RapidPoseLabels [10] builds on the same reconstruction algorithm but uses 2D keypoint annotations that are tracked and propagated to all views.

Transparent and reflective object materials are notoriously difficult to annotate since most methods using depth data are unsuitable. The TOD [17] dataset of transparent objects addresses the shortcoming of depth-based methods by using only RGB images. It requires a stereo camera, the presence of fiducial markers for camera pose estimation, and manually provided keypoints to retrieve object poses. The StereOBJ-1M [36] approach is similar but uses a mix of fixed monocular cameras and a hand-held stereo camera.

Our method is similar to ProgressLabeller [37], which uses RGB images and manual multiview silhouette matching. Their results show that a purely RGB-based method is capable of handling transparent objects and provides better training data for training pose estimators than LabelFusion, which is based on depth sensor data. Our work also uses a projection-based manual annotation approach but additionally provides active support with automatic object alignment, resulting in both faster and more accurate annotation.

Recent advances in synthetic image creation based on NeRF [16] have been included in semantic scene annotation [38]. DexNeRF [12] showed that NeRF can be used to capture geometric information from transparent objects but optimization takes a significant amount of time. Instant-ngp [39] demonstrated that NeRF can be optimized in an efficient manner. Leveraging these ideas, 3D-DAT combines a quick manual annotation process with an efficient NeRF-based reconstruction method to automatically align manually placed objects using only RGB images. The user matches 3D template models to sequences of images with the help of a GUI and after a coarse alignment of an object, an auto-

alignment function refines the initialization to complete the process. The resulting precise 6D pose annotation is used to create pixel-wise annotation of images by the projection of 3D models to all camera views of a recording.

III. 3D-DAT Alignment Method

In this section, we describe the data annotation process shown in Figure 2. We introduce our manual multi-view alignment approach, which is supported by an auto-align procedure to increase annotation speed and accuracy.

A. Multiview Projection-Based Manual Alignment

We create a digital replica of a recorded scene and enable the user to manually align object models to images in order to create an overlap between the 2D projection of the model with the real object in the captured images. The object pose is retrieved by manually changing the position of the object and visually verifying the current state of the overlap between the object in recorded views and the projected mask, or silhouette, of the object in the annotated pose using different camera views.

The camera views of a scene provide a set of camera images $I_i \in I$ with associated camera intrinsic:

$$C_{param} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_x & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{1}$$

with focal lengths f_x, f_y and principal points c_x, c_y.

For the current camera pose and object model pose $C_{world}, O_{world} \in SE(3)$ the 2D projection to the current camera view is defined as:

$$\begin{bmatrix} x^{'} \\ y^{'} \\ z^{'} \end{bmatrix} = C_{param} \cdot C_{world}^{-1} \cdot O_{world} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2}$$

The object pixels in a single image $I_i[u, v]$ are calculated with $u = \frac{x^{'}}{z^{'}}$ and $v = \frac{y^{'}}{z^{'}}$.

All changes in one view propagate to all the other views, thus it is not necessary to annotate every image. However, varying viewing angles of the recorded scene is required for exact alignment and to resolve perspective ambiguities (e.g. scale vs. position). The manual alignment method is used for coarse alignment of objects before initiating the auto-align method, or for final adjustments if necessary.

B. Auto-Alignment with NeRF Generated Depth Maps

The initial coarse pose that is manually supplied by the user is automatically refined to make it more precise. This auto-alignment method relies on creating a surface reconstruction using depth maps of each view, which are fused using Truncated Signed Distance Function (TSDF) [40] volume integration followed by the Marching Cube [41] algorithm to retrieve the final mesh of a scene. In preparation for the alignment, point clouds are generated from both the reconstructed scene mesh and from the 3D object model by sampling points on their surfaces. With these generated point clouds, the multi-scale Iterative Closest Point (ICP) [42] algorithm is applied to minimize the distance between the two point clouds of the object and the scene yielding a transformation that improves the initial object pose.

Since depth sensor readings are not reliable for certain materials (e.g. shiny, metal, transparent), we use depth maps generated by NeRF [16]. NeRF learns an implicit scene representation that takes as input a 5D coordinate of a spatial location (x, y, z) and a viewing direction (θ, ϕ) and outputs for a set of samples along a camera ray the volume density σ_i and RGB color $\boldsymbol{c_i}$. For training the NeRF's multi-layer perceptron (MLP), multi-view RGB images of a static scene with camera extrinsic and intrinsic parameters are required. The approach uses volume rendering [43] to calculate the expected color $C(\boldsymbol{r})$ of the camera ray $\boldsymbol{r} = \boldsymbol{o} + t\boldsymbol{d}$ between the near t_n and far t_f scene bounds, where $\boldsymbol{o}$ is the camera center, $\boldsymbol{d}$ is the viewing direction, and $t \in [t_n, t_f]$, according to:

$$C(\boldsymbol{r}) = \sum_{i=1}^{N} w_i \boldsymbol{c}_i \tag{3}$$

where $w_i = T_i(1 - \exp(-\sigma_i \delta_i)), T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$ and $\delta_i = t_{i+1} - t_i$ is the distance between adjacent samples on ray $\boldsymbol{r}$. During training, it minimizes the error between the input image RGB color and rendered colors.

The authors of DexNeRF [12] observed that NeRF can be used to render depth of transparent objects. To obtain the depth map used in the reconstruction, rays are shot for each pixel. Instead of using the weighted sum, we render depth as in DexNeRF by searching for the first sample along the ray for which the density $\sigma_i > m$ where m is a fixed threshold value. The depth is set to the distance of that sample σ_i.

We use Instant-ngp [39] to benefit from fast training times and add the depth render strategy as described above. This enables the alignment method to work with RGB images only, even for shiny, metallic, and transparent objects.

IV. 3D-DAT System Components

In this section, we describe the full system for data collection and annotation by introducing the tools from 3D-DAT (see Figure2). The minimum requirement is the availability of RGB image data together with the corresponding camera poses and camera parameters. In our work, this is made available by a robotic system. We provide an example of how to use a robotic arm for data gathering in Section V-A. Additionally, 3D object models are required.

3D-DAT separates the steps needed for data preparation and annotation in such a way that the first stage is processed unattended for all recorded data:

1. Train a NeRF using scene images, camera poses, and intrinsics.
2. Use the trained Nerf to render synthetic depth maps.
3. Create and sample a TSDF-surface reconstruction from the depth maps.

Training NeRF for a scene with 100 images with 4000 iterations needs 96 secs using an NVIDIA RTX3090 GPU. Additionally, rendering takes 90 secs and reconstruction takes 20 secs.

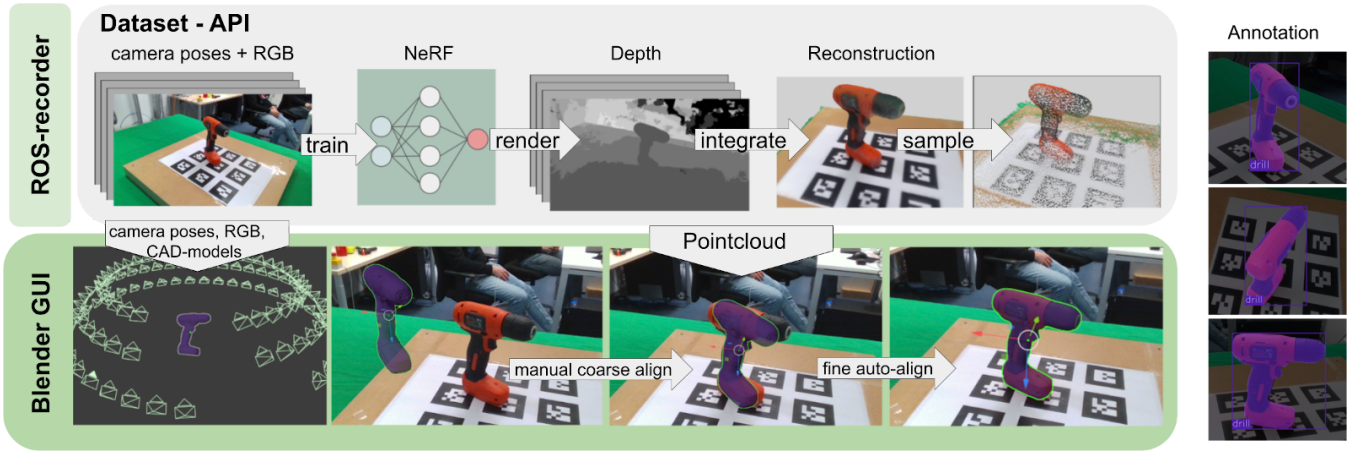


Fig. 2: Workflow of the annotation process: Recorded RGB images, camera poses and intrinsics are used to train NeRF, which is used to generate a pointcloud of each recorded scene. The result together with the images and camera information of a scene are imported to the Blender GUI. After coarse manual alignment, the auto-align method uses ICP to improve the object pose. In addition to the annotated pose, masks and bounding boxes can be generated.

After the necessary data is generated, the user annotation workflow starts:

1. Import images, camera poses, camera intrinsics, 3D object models, and scene pointclouds from the data preparation step.
2. Use the GUI for coarse alignment of object models.
3. Initiate the auto-alignment of the current object using ICP and preprocessed data.
4. Refine the result manually if necessary (repeat 2 and 3).

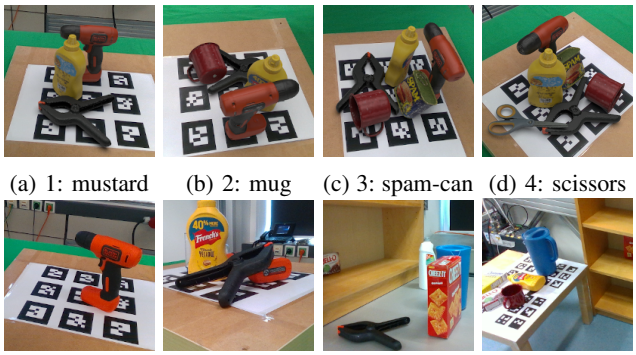
We create the user interface for our annotation tool as a Blender add-on. The move and rotation widgets offered by Blender are used to place and align objects. The user can quickly navigate and browse the imported camera views and has permanent visual feedback from the back projection of the 3D model. The user can adapt the transparency of the 3D model to make adjustments more easily visible by using, for example, only the silhouette of the object.

A dataset-API building upon Open3D [44] framework encapsulates necessary core functions such as importing data, saving poses, interacting with the recorded data, and creating the necessary data for the auto-align method, mask creation, and visualization.

The ROS recorder makes data published on ROS topics persistent. It can be configured to record *Image*, *Pointcloud*, and *Pose* topics. Configurations need to be set up only once for the recording system and can then be reused. To make a recording, a triggering mechanism initiates the recorder to save a snapshot of the configuration to the file system. The recordings are ready to be used with our annotation tools. Note that the depth and pointcloud data is not necessary for annotation but it is interesting for evaluating methods that use those modalities (e.g., [5]).

V. Experiments & User Studies

To evaluate the capability of pose annotation we conduct two user studies. In the first study, we examine pure manual annotation on a set of synthetic and recorded scenes shown in Figure 3. In the second study, we evaluate the different auto-align methods on the set of recorded scenes.



(a) 1: mustard (b) 2: mug (c) 3: spam-can (d) 4: scissors

(e) 5: drill (f) 6: mustard (g) 7: bleach (h) 8: mug

Fig. 3: Scenes (scene number: object) used for the user studies. Figures (a)-(d) show synthetic rendered scenes, (e)-(h) are real scenes captured from cameras.

A. Experimental Setup

To create synthetic scenes, we use the annotation tool to import a prerecorded empty scene. We manually place object models into the scene and use Blender's cycle engine to render all objects overlayed on the imported camera images.

To record real scenes, we use a *KUKA lwr iiwa 14* robot arm with *intel-realsense d415/d435* camera mounted on the end-effector as shown in Figure 1. Extrinsic camera calibration is performed using a marker-based optimization approach [45]. After the end-effector calibration, the markers are no longer needed. We use two trajectories, one sampled on a half-sphere with the center located along the z-axis of the scene, and another with random views facing toward the scene. We use our ROS-recorder to capture all the scenes.

The image resolution for both synthetic and real images

is 1280×720px. Cropped sample images of each scene used in the experiments can be seen in Figure 3. For each scene, 61 views are available.

The scenes are designed to increase the difficulty of the annotation process. The synthetic scenes (Figure 3a-3d) are easier than the real scenes because the camera pose errors are non-existent and object models are perfect. The synthetic scenes are made more difficult by introducing clutter and touching objects. For the scissors in scene 4, the difficulty is increased further because of a slight misalignment of the local coordinate system from its intuitive symmetry axes. The real scenes (Figure 3e-3h) start with the best calibrated robotic setup and a non-occluded object. The object model used in this scene is of high resolution (manually scanned using the Artec Eva scanner), yielding the best 3D model of all real objects. We use the same camera poses as in the synthetic scenes (arranged in a half-sphere), which provide the best views of the object. For the remaining three objects we use available models from the BOP challenge [6] of the YCB-video dataset [5]. For the last two scenes, we further increase the difficulty by switching to random views.

B. Metrics

Annotation accuracy is evaluated by calculating the translation error as:

$$t_{err} = \|a - g\|_2, \tag{4}$$

where a, $g \in \mathbb{R}^3$ correspond to the (x, y, z) coordinates of the object centers for the annotation and ground truth, respectively. The rotation error is measured as the angle between two rotation matrices as follows:

$$\theta_{err} = \arccos\left(\frac{\operatorname{tr} AG^{-1} - 1}{2}\right), \tag{5}$$

where A and $G \in SO(3)$ are rotation matrices of the annotation and the ground truth.

For the real scene, the ground truth poses are not available, thus we instead calculate an average pose over all annotations and substitute the ground truth variables in Equations (4) and (5) accordingly. The mean center of a pose is retrieved by averaging the (x, y, z) coordinates. The mean rotation is calculated using chordal-L2-means[1] over all rotations. From this, we derive a notion of agreement of the pose annotation for a specific scene conducted by different users. Low values indicate high consistency among users on a common annotation pose, thus a greater agreement on the pose is achieved. High variance indicates an inconsistency in the pose of objects. In the case of the synthetic scenes, however, ground truth poses are available and can be directly compared to the manual pose annotations.

C. User Studies

The first user study evaluates the accuracy of manual annotation of synthetic and real scenes. We conduct the study by inviting 12 participants. The participants have some experience using 3D software (e.g., CAD software). One participant is experienced (>30h), and two users have minor experience (<10h) with our tool. The remaining users are new to the tool. After a 20 minute tutorial, including a guided hands-on annotation session, the participants annotate different objects in all eight scenes without assistance.

To evaluate the auto-alignment function we invite the participants a second time, where 9 people from the original group participate again. The second trial is scheduled with a sufficient time gap (>60 days) from the first session to minimize a possible memory effect. The participants are requested to annotate the recorded scenes twice, using the alignment function with data from a depth sensor and rendered depth maps using NeRF. To avoid bias toward a specific method, we switch the order of methods between each user session.

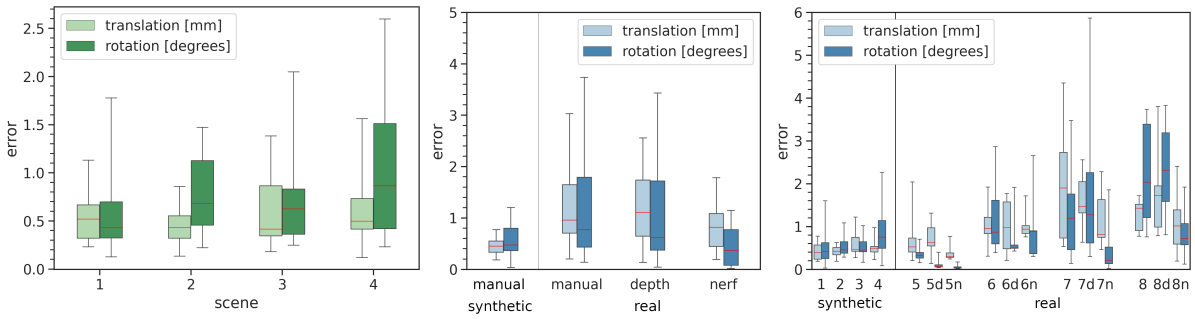
D. Results

The results are shown in Figure 4. For the synthetic scenes, a translation error average of 0.55mm is observed and 10 out of 12 participants stay below 1.0mm error. The average rotation error is 0.77° and 8 out of 12 participants stay below 1.5°. Figure 4a clearly shows the negative effect of the misaligned axes for the object (scissors) of scene 4; participants score worst in finding the correct rotation. Overall, the results on the synthetic scenes show that highly accurate annotation of 6D poses is possible using manual annotation when conditions are optimal.

The deviation from the mean translation and rotation for both synthetic and real scenes is presented in Figure 4b-4c. Comparing synthetic and real scenes shows that user agreement on a common pose is best for synthetic data. There is a significant loss in agreement for real scenes when conducting manual and auto-align annotation using depth sensors. The best result for the real scenes, and most similar to synthetic scenes, is achieved when using auto-align with NeRF. Translation distance from the mean is on average 0.47mm for synthetic scenes. For real scenes, the error is 1.25mm when using only manual annotation, 1.26mm when including auto-alignment with depth sensors, and 0.9mm when including auto-alignment with NeRF. Rotational error from the mean is on average 0.63° for synthetic data. For real scenes, the error is 1.29° when annotated manually, 1.24° when using auto-alignment with depth sensors, and 0.65° for NeRF auto-aligned objects. Deviations are noticeably worse on the more difficult scenes but overall the agreement of the user annotations is still within a small range. Overall, using NeRF data for auto-alignment is closest to the agreement on a pose for synthetic scenes.

The results for annotation speed are presented in Table I. Users spend on average 9.4 mins annotating a whole image sequence for a scene without assistance. Including the auto-alignment in the workflow reduces the time significantly to 4.1 mins when using data from a depth sensor and 2.9 mins when data is generated by NeRF. The NeRF-generated depth maps provide much better input data for the alignment procedure, thus less manual work is required.

[1]using the implementation from scipy `https://scipy.org/`



(a) (b) (c)

Fig. 4: (a) Evaluation of the synthetic part of the user study showing deviations from ground truth poses. (b-c) Evaluation showing deviations from the means of annotated object poses. (b) Comparison of manual methods for synthetic and real scenes, and auto-align methods based on depth sensors or NeRF. (c) Detailed comparison for single scenes. (no postfix=manual, d=depth sensor, n=NeRF.)

	manual [sec]		*depth [sec]*		*NeRF [sec]*	
scene	avg	std	avg	std	avg	std
5 drill	524	220	112	42	**81**	**37**
6 mustard	640	153	**188**	81	190	**64**
7 bleach	521	184	340	148	**162**	**106**
8 mug	581	176	340	172	**256**	**167**
all	567	184	245	154	**172**	**119**

TABLE I: Annotation times without (manual) and with assisting auto-alignment using depth sensor data and NeRF.

VI. Examples

Annotation examples for a variety of different objects are shown in Figure 5. The first row shows an example of annotation for a subset of objects from the HOPE dataset [4] with overlayed masks, bounding boxes, and class labels. The second row depicts a scene with YCB objects [29] for the user study showing a more complicated setup. This shows a weakness of our approach due to the accuracy of 3D models. For example, the pitcher has a slightly different handle in the model. The last two rows show resulting masks retrieved from a scene of transparent and industrial objects made of plastic and metal, which produce noisy depth readings. The reliance on depth data is a strong limitation for other 6D pose annotation methods that we overcome with 3D-DAT, allowing annotation of objects such as those displayed here.

VII. Conclusion

This work introduced 3D-DAT – a set of tools supporting robotic researchers to create datasets for evaluation and training of methods for robotic vision tasks using their robotic setup at hand. It is an efficient toolset for retrieving 6D pose annotation and pixel-wise annotated RGB/RGBD data at the object level. The 3D-DAT GUI is implemented as a Blender add-on and uses multiple views to allow accurate 3D model alignment. It includes an auto-alignment function, using NeRF combined with a reconstruction pipeline and ICP, to improve the speed and precision of annotation.

Results from our user studies show similar accuracy to manually labeled data but with improved annotation speed.

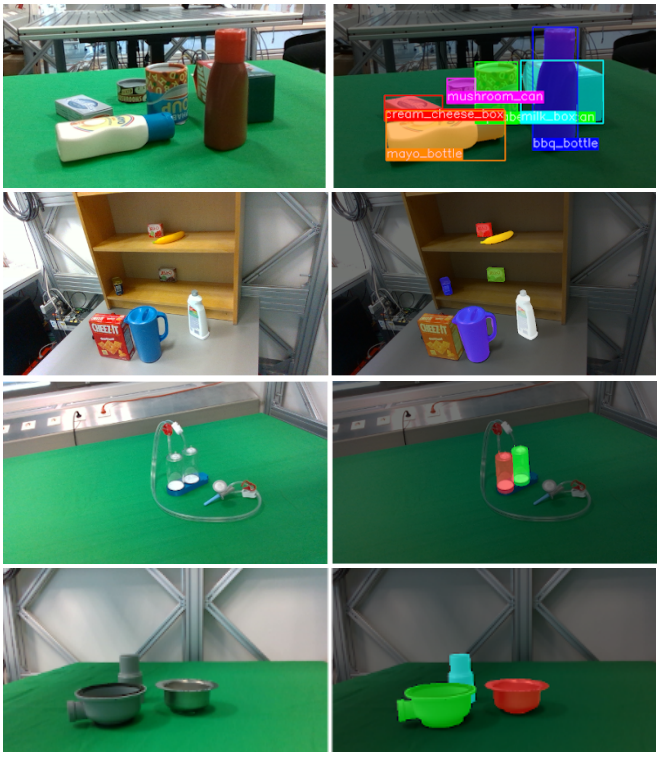


Fig. 5: Top row shows annotation examples from the HOPE dataset, followed by YCB-objects, transparent, and industrial objects in the last row.

Furthermore, using reconstructions created with NeRF results in better and faster annotation compared to using raw depth sensor data. 3D-DAT does not depend on depth sensors, thus enabling annotating transparent and shiny metal objects.

Possible extensions to our method include camera pose refinement using NeRF [46] and including physical considerations into the auto-alignment procedure. Investigating methods, e.g. single-shot pose estimators, to eliminate the manual coarse object alignment is needed to achieve a fully automated annotation pipeline.

REFERENCES

[1] S. Hinterstoißer, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," *Proc. of IEEE ICCV*, pp. 858–865, 2011.

[2] T. Hodan, P. Haluza, S. Obdrzálek, J. Matas, M. I. A. Lourakis, and X. Zabulis, "T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects," *Proc. of IEEE WACV*, pp. 880–888, 2017.

[3] R. Kaskman, S. Zakharov, I. Shugurov, and S. Ilic, "HomebrewedDB: RGB-D dataset for 6D pose estimation of 3D objects," in *Proc. of IEEE/CVF (ICCVW*, 2019, pp. 2767–2776.

[4] S. Tyree, J. Tremblay, T. To, J. Cheng, T. Mossier, and S. Birchfield, "6-DoF pose estimation of household objects for robotic manipulation: An accessible dataset and benchmark." *ICCV Workshop*, 2019.

[5] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," in *Proc. of RSS*, 2018.

[6] T. Hodaň, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas, "BOP challenge 2020 on 6D object localization," *European Conference on Computer Vision Workshops (ECCVW)*, 2020.

[7] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6D pose estimation," in *Proc. of ECCV*, 2020, pp. 574–591.

[8] R. L. Haugaard and A. G. Buch, "SurfEmb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings," *arXiv preprint 2111.13489*, 2021.

[9] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake, "Label fusion: A pipeline for generating ground truth labels for real RGBD data of cluttered scenes," in *Proc. of IEEE ICRA*, 2018, pp. 3325–3242.

[10] R. P. Singh, M. Benallegue, Y. Yoshiyasu, and F. Kanehiro, "Rapid pose label generation through sparse representation of unknown objects," in *Proc. of IEEE ICRA*, 2021, pp. 10 287–10 293.

[11] G. Halmetschlager-Funek, M. Suchi, M. Kampel, and M. Vincze, "An empirical evaluation of ten depth cameras: Bias, precision, lateral noise, different lighting conditions and materials, and multiple sensor setups in indoor environments," *IEEE Robotics & Automation Magazine*, vol. 26, no. 1, pp. 67–77, 2019.

[12] J. Ichnowski, Y. Avigal, J. Kerr, and K. Goldberg, "Dex-NeRF: Using a neural radiance field to grasp transparent objects," in *Proc. of CoRL*, 2020.

[13] H. Fang, H.-S. Fang, S. Xu, and C. Lu, "TransCG: A large-scale real-world dataset for transparent object depth completion and a grasping baseline," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7383–7390, 2022.

[14] B. O. Community, "Blender - a 3D modelling and rendering package," Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

[15] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proce. of IEEE 3DIM*, 2001, pp. 145–152.

[16] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Proc. of ECCV*, 2020, pp. 405–421.

[17] X. Liu, R. Jonschkowski, A. Angelova, and K. Konolige, "KeyPose: Multi-view 3D labeling and keypoint estimation for transparent objects," in *Proc. of IEEE/CVF CVPR*, 2020.

[18] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "GraspNet-1Billion: A large-scale benchmark for general object grasping," in *Proc. of IEEE/CVF CVPR*, 2020, pp. 11 444–11 453.

[19] I. Hjortgren and A. Lindholm, "Measuring data quality efficiently," https://www.annotell.com/articles/measuring-data-quality-efficiently/, 2020, accessed: 2022-03-22.

[20] T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. of ECCV*, 2014.

[21] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2015.

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. of IEEE CVPR*, 2009, pp. 248–255.

[23] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: A database and web-based tool for image annotation," *International Journal of Computer Vision*, vol. 77, no. 1, pp. 157–173, 2008.

[24] B. E. Moore and J. J. Corso, "Fiftyone," *GitHub: https://github.com/voxel51/fiftyone*, 2020.

[25] K. Maninis, S. Caelles, J. Pont-Tuset, and L. V. Gool, "Deep extreme cut: From extreme points to object segmentation," in *Proc. of IEEE/CVF CVPR*, 2018, pp. 616–625.

[26] D. Acuna, H. Ling, A. Kar, and S. Fidler, "Efficient interactive annotation of segmentation datasets with Polygon-RNN++," in *Proc. of IEEE/CVF CVPR*, 2018, pp. 859–868.

[27] Z. Liu, W. Liu, Y. Qin, F. Xiang, M. Gou, S. Xin, M. A. Roa, B. Calli, H. Su, Y. Sun, and P. Tan, "OCRTOC: A cloud-based competition and benchmark for robotic grasping and manipulation," *arXiv preprint 2104.11446*, 2021.

[28] Y. Chao, W. Yang, Y. Xiang, P. Molchanov, A. Handa, J. Tremblay, Y. S. Narang, K. V. Wyk, U. Iqbal, S. Birchfield, J. Kautz, and D. Fox, "DexYCB: A benchmark for capturing hand grasping of objects," in *Proc. of IEEE/CVF CVPR*, 2021, pp. 9044–9053.

[29] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the Yale-CMU-Berkeley object and model set," *IEEE Robotics & Automation Magazine*, vol. 22, no. 3, pp. 36–52, 2015.

[30] R. Monica, J. Aleotti, M. Zillich, and M. Vincze, "Multi-label point cloud annotation by selection of sparse control points," in *Proc. of IEEE 3DV*, 2017, pp. 301–308.

[31] J. Valentin, V. Vineet, M.-M. Cheng, D. Kim, J. Shotton, P. Kohli, M. Nießner, A. Criminisi, S. Izadi, and P. Torr, "SemanticPaint: Interactive 3D labeling and learning at your fingertips," *ACM Transactions on Graphics*, vol. 34, no. 5, pp. 154:1–154:17, 2015.

[32] M. Suchi, T. Patten, D. Fischinger, and M. Vincze, "Easylabel: A semi-automatic pixel-wise object annotation tool for creating robotic RGB-D datasets," in *Proc. of IEEE ICRA*, 2019, pp. 6678–6684.

[33] D. Stumpf, S. Krauß, G. Reis, O. Wasenmüller, and D. Stricker, "SALT: A semi-automatic labeling tool for RGB-D video sequences," in *Proc. of VISIGRAPP*, 2021.

[34] C. Rother, V. Kolmogorov, and A. Blake, ""GrabCut": Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 309–314, 2004.

[35] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "ElasticFusion: Real-time dense SLAM and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[36] X. Liu and K. M. Kitani, "V-MAO: Generative modeling for multi-arm manipulation of articulated objects," in *Proc. of CoRL*, 2021.

[37] X. Chen, H. Zhang, Z. Yu, S. Lewis, and O. C. Jenkins, "Progress-Labeller: Visual data stream annotation for training object-centric 3D perception," *Proc. of IEEE/RSJ IROS*, 2022.

[38] S. Zhi, T. Laidlow, S. Leutenegger, and A. J. Davison, "In-place scene labelling and understanding with implicit scene representation," in *Proc. of IEEE/CVF ICCV*, 2021, pp. 15 838–15 847.

[39] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics*, vol. 41, no. 4, pp. 102:1–102:15, 2022.

[40] Q.-Y. Zhou and V. Koltun, "Dense scene reconstruction with points of interest," *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 112:1–112:8, 2013.

[41] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, p. 163–169, 1987.

[42] J. Park, Q.-Y. Zhou, and V. Koltun, "Colored point cloud registration revisited," in *Proc. of IEEE ICCV*, 2017, pp. 143–152.

[43] N. Max, "Optical models for direct volume rendering," *IEEE Transactions on Visualization and Computer Graphics*, vol. 1, no. 2, pp. 99–108, 1995.

[44] Q. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv preprint 1801.09847*, 2018.

[45] F. Park and B. Martin, "Robot sensor calibration: Solving ax=xb on the Euclidean group," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 5, pp. 717–721, 1994.

[46] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, "BARF: Bundle-adjusting neural radiance fields," in *Proc. of IEEE ICCV*, 2021.