

# Shaken, Not Stirred: A Novel Dataset for Visual Understanding of Glasses in Human-Robot Bartending Tasks

Lukáš Gajdošech<sup>1†</sup>, Hassan Ali<sup>2†</sup>, Jan-Gerrit Habekost<sup>2†</sup>, Martin Madaras<sup>1</sup>, Matthias Kerzel<sup>2</sup>, and Stefan Wermter<sup>2</sup>

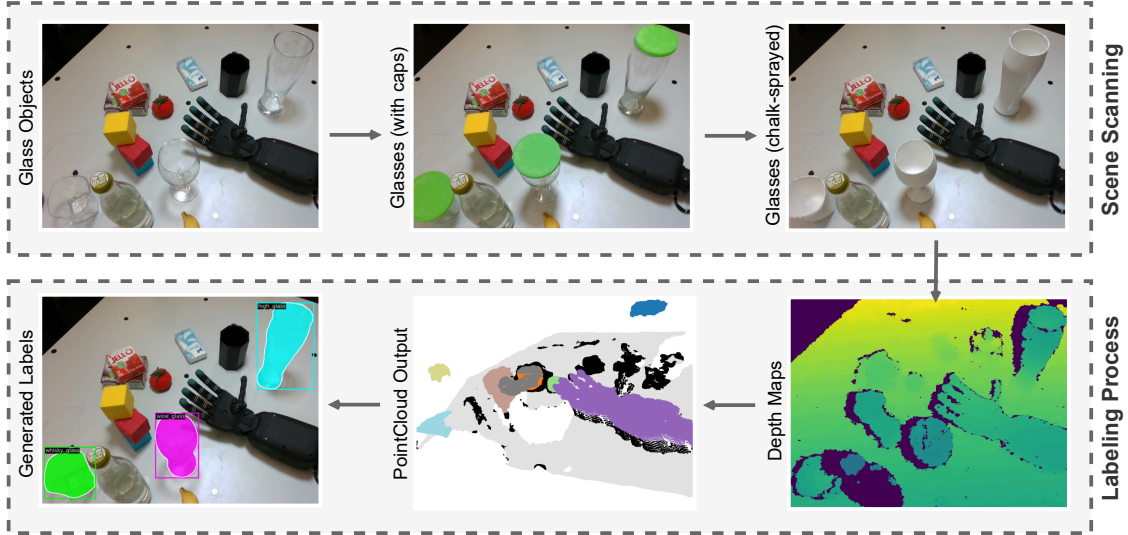


Fig. 1: An overview of our proposed auto-labeling pipeline: Each scene is scanned in three stages. Depth maps and point clouds are generated using RGB-D sensors, followed by height verification, color matching, and object detection.

**Abstract**—Datasets for object detection often do not account for enough variety of glasses, due to their transparent and reflective properties. Specifically, open-vocabulary object detectors, widely used in embodied robotic agents, fail to distinguish subclasses of glasses. This scientific gap poses an issue for robotic applications that suffer from accumulating errors between detection, planning, and action execution. This paper introduces a novel method for acquiring real-world data from RGB-D sensors that minimizes human effort. We propose an auto-labeling pipeline that generates labels for all the acquired frames based on the depth measurements. We provide a novel real-world glass object dataset<sup>3</sup> that was collected on the Neuro-Inspired Collaborator (NICOL), a humanoid robot platform. The dataset consists of 7850 images recorded from five different cameras. We show that our trained baseline model outperforms state-of-the-art open-vocabulary approaches. In addition, we deploy our baseline model in an embodied agent approach to the NICOL platform, on which it achieves a success rate of 81% in a human-robot bartending scenario.

<sup>†</sup>These authors contributed equally to this work.

\*Work supported by Horizon Europe project TERAIS (Grant number 101079338) and the DFG Crossmodal Learning (TRR-169) project.

<sup>1</sup>Lukáš Gajdošech and Martin Madaras are with Department of Applied Informatics, Faculty of Mathematics, Physics and Informatics, Comenius University, Bratislava, Slovakia {lukas.gajdosech, martin.madaras}@fmph.uniba.sk

<sup>2</sup>Hassan Ali, Jan-Gerrit Habekost, Matthias Kerzel and Stefan Wermter are with the Knowledge Technology Group, Department of Informatics, University of Hamburg, Hamburg, Germany {hassan.ali, jan-gerrit.habekost, matthias.kerzel, stefan.wermter}@uni-hamburg.de

<sup>3</sup><https://knowledgegtechnologyh.github.io/GlassNICOLDataset/>

## I. INTRODUCTION

Transparent objects are everywhere, from households, healthcare assistance, and gastronomy to industrial and construction-site environments. Yet, these materials reflect only a very small fraction of visible light—most of it is scattered and passes through. From a computer vision perspective, the processing of these objects is challenging but crucial for a successful deployment in real-world robotics applications [1]. As intelligent robots start to make their way into public spaces, where glass objects in the form of bottles and drinking glasses are integral items, future generations of service robots will be required to not only detect but also manipulate those materials. In addition, machine learning and data-driven methods require a large amount of training data to reach a suitable performance. Similar to robotic Sim2Real applications, existing research also proves the presence of a significant Sim2Real gap in the area of synthetic glass material generation [1]–[3], while those approaches require sophisticated photorealistic rendering methods.

Open-vocabulary approaches are used for object detection in a plethora of embodied robotic setups, such as [4], [5]. Embodied robotic agents, at a minimum, consist of a detection module, an action module, and an LLM utilized for high-level interaction and task planning. Recent approaches aim to fuse detection and language processing by utilizing vision-language models (VLMs) [6]. Being interactive by

design, these robotic agents are typically situated in scenarios involving user interaction and often collaboration, leaving a potentially infinite corpus of object names to be detected.

Well-known open-vocabulary object detection approaches such as Grounding DINO [7], YOLOWorld [8], and OWL-ViT [9] have revolutionized object detection by expanding the recognizable object classes without extensive re-annotation. These methods are built upon end-to-end transformer-based architectures. Global image features are extracted by decoder backbones and fused with language embeddings by utilizing deep metric learning. Despite their success, our experiments suggest that these models exhibit a significant decrease in performance when encountering transparent objects, commonly found in HRI scenarios. To address these challenges, we introduce a novel real-world dataset that was captured on the humanoid robot NICOL (Fig. 2), including a new depth-based automated labeling method. The robot is equipped with a combination of RGB-D scanners and standard RGB cameras. Specific contributions of our work include:

- A **novel real-world dataset** specifically designed for glass object detection, addressing the limitations of synthetic datasets and bridging the Sim2Real gap.
- An **automated labeling pipeline**, shown in Fig. 1, allowing a rapid annotation of data, leveraging existing visual foundation models.
- A **real-time integration** of the visual detector with a physical robot, showing its potential in HRI scenarios.
- A **humanoid bartender task** as a use case, where the robot accurately detects glassware and executes pouring motions, establishing a platform for controlled experiments in realistic, socially engaging contexts.

## II. RELATED WORK

### A. Transparent Material Perception

Missing or incorrect RGB-D data for robotic perception can be estimated using implicit functions and geometric priors [10] and by balancing local and global depth features [11]. The depth map can also be de-projected into a point cloud and processed with techniques like 3D CNNs to achieve depth completion [12]. Iterative refinement is also possible, with an attention module on transparent regions [13] or with indirect geometry representations [14].

RGB often contains information missing from depth data, which can be leveraged with transfer learning [15]. Cues can be extracted from the RGB to fill in depth observations using global optimization [1]. Incorporating affordance detection of the underlying object yields further improvement [16]. Other approaches include multimodality, such as visual-tactile fusion [17] and usage of polarization imaging [18].

Recently, neural radiance fields (NeRFs), have emerged as a promising tool for transparent object perception. Grasp-NeRF introduces a multiview-based 6-DoF grasp detection system [19]. Evo-NeRF extends this approach with real-time NeRF training and grasp adaptation [20]. Multiview methods enhance transparent object perception by integrating multiple viewpoints [21]. Domain randomization techniques have also

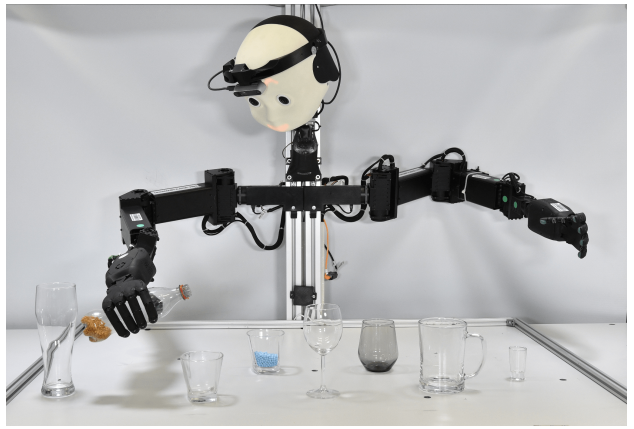


Fig. 2: The NICOL humanoid robot used for down-stream task evaluation.

been used to train models that generalize well to the real-world [22]. The common limitation of existing approaches is their unpredictable performance on unseen scenes, showing the need for the methodology of automated benchmark data aggregation and annotation without human labor [2].

### B. Existing Datasets

The field of transparent object perception has long faced a scarcity of real-world datasets, with the majority of the work driven by synthetic data [2], [14], [23]. Synthetic datasets like Trans10k [24], SuperCaustics [25], and Dex-NeRF [26] offer a vast number of images (ranging from 9k to 100k) but suffer from the Sim2Real gap [27]. Besides common causes like the lack of noise and imperfections presented in real-world environments, transparent objects provide a unique challenge due to their complex light interaction. In contrast, real-world datasets like TransCG [28] and ClearGrasp [1] have attempted to address these limitations by collecting depth data using physical sensors. However, they lack diversity. Many datasets focus on isolated objects placed against controlled backgrounds, limiting their utility in highly cluttered environments [2]. The process of manual ground-truth depth and segmentation annotation remains a significant bottleneck [29]. Few works, like the Toronto Transparent Object Depth dataset leverage automated annotation techniques [12]. Also, Liu et al. [30] presented a multiview automated annotation approach. However, it is limited to isolated scenes and does not address the demanding challenge of dense transparent object detection and classification in realistic HRI scenarios, which our dataset is specifically designed for.

Moreover, many existing large-scale RGB-D datasets [31] do not specifically focus on transparent objects [32]. We also note the existence of datasets like RGBP-Glass [33] and RGB-Thermal [34], introducing additional sensing modalities. Yet, these remain niche applications and do not fully address the generalization problem for robotic perception.

## III. METHODOLOGY

### A. NICOL Platform and Camera Model

The humanoid robot NICOL (see Fig. 2) has two 8-DoF arms, each with an anthropomorphic tendon-driven 5-

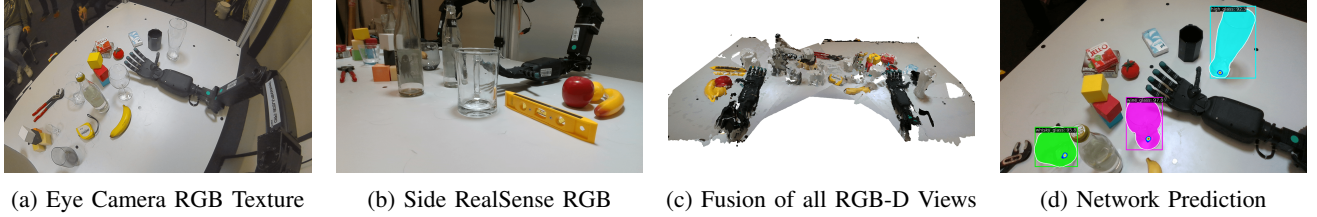


Fig. 3: Different modalities present in our dataset, visualized on `scene_145` from the validation split.

DoF hand. It has a 2-DoF 3D-printed head structure with an integrated facial expression interface. The platform is designed for manipulation and HRI scenarios and offers various sensors for different modalities. A two-meter-wide, one-meter-deep tabletop is attached in front of NICOL. The robot head contains two RGB fish-eye cameras with a 4k resolution. In our setup, a RealSense D435i camera is additionally mounted to NICOL’s head, along with two RealSense cameras to the left and right front pillars of the robot’s frame. Thus, our setup enables a complete depth perception of the objects on the table from multiple perspectives.

The movable head and large area on the table covered by NICOL’s arms, combined with the fish-eye property of the RGB eyes, require a very precise camera calibration to allow for complex manipulation tasks such as pouring liquids. We use a semi-automated calibration method, which scans the table in front of NICOL from different angles to collect calibration data. Small calibration markers are attached to the tabletop that are automatically detected with a blob detection method. The detected positions in the camera image are manually mapped to the corresponding 3D ground truth positions by a human. The collected data is then used with the OpenCV camera calibration method [35] to calculate the camera matrix  $K$ , the distortion coefficients  $D$ , as well as the rotation matrix  $R$  and the translation vector  $t$ . The calibration has to be performed individually for each of the three cameras used in our setup. We reached a reprojection error of  $\sim 2$  pixels for the RGB eye cameras and  $\sim 4.5$  pixels for the RGB-D camera. The two static RealSense cameras were calibrated with the well-known chessboard calibration method. Our camera model utilizes the camera parameters to provide a ray-casting method between the table and the camera sensor. In other words, pixels from one camera can be cast to the corresponding 3D coordinate at the tabletop and, for example, back into one of the other cameras to mark a specific 3D coordinate in the images from all five cameras.

### B. Dataset Design

Recent approaches dealing with transparent object perception usually rely on synthetically rendered data [1], [3], [21]. The amount of real-world data from scenes containing these objects is severely limited, in comparison to datasets with objects from opaque materials [2], [31]. This is especially true for scenes with cluttered environments and a mix of transparent and non-transparent objects. We have designed a new benchmark dataset that is applicable to several computer vision and robotic tasks. Image frames are captured from the

egocentric view of the NICOL robot [36] performing various arm movements. Thus, complexity in various modalities is added as the dataset includes various degrees of occlusion, varying lighting conditions, and differing tabletop textures.

Specifically, our dataset consists of scenes with a mix of objects placed on a table in front of a humanoid robot (see Fig. 3). The collection process involves three separate passes:

- 1) Capturing the scene with clean glasses.
- 2) Placing 3D-printed green caps on top of the glasses for height measurement.
- 3) Replacing glasses with identical instances sprayed by chalk spray, making them opaque for obtaining depth ground truth geometry.

This design allows us to provide glass classification and detection labels, segmentation masks, and ground-truth depth measurements. Using the processing pipeline described in the following subsection, all annotations are created *automatically, without human labor*. As mentioned earlier, our setup consists of three RealSense RGB-D cameras, one attached to the robot’s head and two at the sides of the table. Also, there are two fisheye RGB cameras, one in each eye of the robot. Using this setup, we captured 100 different scenes, with variations in table color and texture, lightning conditions, robot movements, and scene compositions. For each scene, there are 25 different views from the robot, acquired by a rotation in the neck joint. All cameras were calibrated and registered, allowing a projection of detection labels from the head-mounted RGB-D sensors to all other views.

The sum of frames acquired from all five cameras (2 static RGB-D cameras and 25 images times 3 head cameras) results in a number of 77 frames per scene and 7700 frames in the training dataset in total. There are six different glass types present, each with its own class, namely: *high beer glass*, *beer glass with a handle*, *wine glass*, *water glass*, *whiskey glass*, and *shot glass*. For each type, we also provide a high-quality 3D model usable in surface reconstruction tasks [37]. Together with ground truth depth observations from the three RGB-D scanners, this can also be used in monocular depth estimation [38] and depth completion tasks [28]. Our main use case of the data in this paper is the detection and classification of a glass type. In a more general perception task, different glass types can be omitted, and all labels can be treated as general *drinking glass* annotations.

Our dataset intentionally contains various unlabeled bottles, serving as negative examples during training. These objects are made of transparent material but are not typically classified by humans as drinking glasses. In addition to the

7700 frames used for training and validation, we captured separate testing data containing 150 frames: 75 frames captured using the same approach as the training data and another 75 frames captured during the bartender experiment described in Sec. IV-B, thus having a completely different character compared to the training data. The labels of the test data were created manually to mitigate the effect of label noise introduced by our automated pipeline described below.

### C. Auto-Labeling Pipeline

The labeling process uses depth sensing, color verification, and object detection to create accurate segmentation masks and bounding boxes. Initially, depth images are converted into a 3D point cloud, denoted as  $P = \{p_i \in \mathbb{R}^3\}$  where each point  $p_i = (x_i, y_i, z_i)$  represents spatial coordinates obtained from the RealSense camera mounted on the robot's head. The primary surface in the scene, corresponding to the table, is estimated using RANSAC-based plane fitting, yielding the plane parameters  $a, b, c, d$ . Points deviating significantly from this plane are identified as potential objects and subsequently clustered using a density-based clustering algorithm from Open3D [39], ensuring each cluster represents a discrete candidate object. Next, the height of the clusters is computed relative to the table plane using the formula:

$$h = \frac{|d' - d|}{\sqrt{a^2 + b^2 + c^2}}, \quad (1)$$

where  $d'$  is the cluster's plane offset. Clusters with large height deviations from known glasses are discarded. The remaining candidates are given classes based on the closest known glass height, then projected onto the 2D image plane using intrinsic camera parameters, transforming 3D coordinates into pixel locations  $(u, v)$  via perspective projection:

$$\begin{bmatrix} u & v & 1 \end{bmatrix}^T \sim K \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T, \quad (2)$$

where  $K$  is the intrinsic calibration matrix of the camera. To further filter instances, color verification is performed in the CIELAB color space, ensuring that the detected object caps match the expected green hue. A final verification step uses YOLO-World, a deep-learning-based object detector, to confirm the detected object has glass-like features, mitigating false positives from other green objects in the scene [8]. As each of these steps works with different data modalities and false-positive error profiles, we achieve a variant of a cascaded filtering approach by fusion of outputs from several algorithms [40]. After obtaining the final candidates, points from the blobs are used as samples for the Segment Anything Model (SAM) [41]. As shown by our final data, SAM achieves reliable segmentation performance in cluttered environments when guided by our precise sample points. To improve accuracy, we discard overly wide masks, often caused by merged neighboring objects. To finalize the object instance label, we derive bounding boxes  $B_i = (x, y, w, h)$  from the masks. An overview of our labeling pipeline and an example annotation were presented earlier (see Fig. 1).

### D. Baseline Detector

To demonstrate the advantages of our novel dataset, we use it to train an object detector and perform use case scenarios and experiments. For real-time detection performance, we opted for an RTMDet architecture in its medium-size variant [42]. Naturally, different one-stage architectures, like the YOLO family [43], could be employed for this task.

The network variant chosen incorporates various training strategies. Specifically, the network employs a CSPNeXt backbone with a P5 architecture, a deepen factor of 0.67, and a widen factor of 0.75, enhanced with channel attention and synchronized batch normalization. The neck is a CSPNeXtPAFPN structure with two CSP blocks [44], facilitating multi-scale feature aggregation [42]. As proposed by the original authors, the head of the network calculates three loss functions. A Quality Focal Loss ( $\beta = 1.0$ , loss weight = 1.0) for classification [45], Complete IoU (CIoU) Loss (loss weight = 2.0) for bounding box regression [46], and Dice Loss (loss weight = 3.0, reduction = "mean") for mask prediction [42]. To increase robustness and grant better generalization for different scenarios, heavy data augmentation is employed. This includes techniques such as Cached Mosaic [47], Cached MixUp [48], Random Resize, Random Crop, and YOLOX-style HSV augmentation [49].

The training is performed from a pre-trained checkpoint [42], which we fine-tune on our dataset. The learning rate is scheduled using a combination of Linear Warmup (start factor = 0.001, epochs 0–50) and MultiStepLR with decay at epochs 100, 200, and 400. The optimizer is AdamW [50] with a learning rate of 0.01 and a weight decay of 0.05. The model is trained for 500 epochs with a batch size of 8, and evaluated every 50 epochs using the COCO metric for both bounding box and segmentation performance on 5 validation scenes (a 5% split from our 100 scenes).

### E. Glass Base Points

To improve the robotic arm guidance without heavy modification of the detector architecture or lifting to 3D [51], we calculate 2D glass base positions separately. Given these locations and the known heights of the glasses from the classification, it enables robot pouring without exact 3D positions. We provide annotations for them in the dataset. By calculating the average position of each 3D-printed cap, we obtain point  $\mathbf{p} = (x, y, z)^\top$  for each glass and collect them into a set  $\mathbf{C}$ . Taking the normal vector of the table  $\mathbf{n} = (a, b, c)^\top$  and its normalized version  $\hat{\mathbf{n}}$ , the perpendicular distance to the table for each  $\mathbf{p} \in \mathbf{C}$  is given by:

$$d_{\text{proj}} = \frac{ax + by + cz + d}{\|\mathbf{n}\|}. \quad (3)$$

The projection of the point onto the table is computed as  $\mathbf{p}_{\text{proj}} = \mathbf{p} - d_{\text{proj}}\hat{\mathbf{n}}$ . To obtain 2D image coordinates, we use perspective projection with the camera matrix  $K$ . Taking 2D coordinates as centers of small bounding boxes with a fixed size, we introduce them as a separate `keypoint` class into the COCO object detection format.



You are deployed to function as a bartender. There are glass objects on the table. You should always respond as a bartender and offer available services like a bartender. When serving drinks, follow these rules:

1. Pouring Wine:
  - Identify the correct bottle.
  - Identify the correct glass.
  - Example: `<pour_wine(wine glass)>`.
- If there is only one wine glass, pour immediately into it without asking the user.
- If there are multiple wine glasses, ask the user which they prefer before pouring.
2. Pouring Beer:
  - Identify the correct bottle.
  - Identify the correct glass.
  - Example: `<pour_beer(beer glass)>`.
  - Example: `<pour_beer(high beer glass)>`.
- If there is only one beer glass, pour immediately into it without asking the user.
- If there are multiple beer glasses, ask the user which they prefer before pouring.
- You can fill a glass only once. When you pour into a glass, you should remember which glass is already filled. Always prioritize serving drinks accurately and safely. Respond to the user like a professional bartender.

Prompt 1: A summarized version of our system LLM prompt of the bartender task with an example of the pouring wine and pouring beer functionalities.

Finally, we propose a modification of the detector head to obtain heat-map of glass base points. Before applying Non-Maximum Suppression [52], we extract all proposals of the `keypoint` class. We calculate the center location  $(x_c, y_c)$  for each proposal and place Gaussian kernel  $G(x, y)$  of size  $k \times k$  (in our experiments  $k = 15$ ) at these locations:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(x - x_c)^2 + (y - y_c)^2}{2\sigma^2}\right). \quad (4)$$

The contribution of each kernel is weighted by the confidence score  $s_i$  of the corresponding bounding box. The final heatmap  $H(x, y)$  is obtained by summing all contributions:

$$H(x, y) = \sum_i s_i G_i(x, y). \quad (5)$$

2D base points are extracted by identifying local maxima in the heatmap within each detected glass bounding box (see Fig. 3d for an example with heatmap overlaid on detected glasses). 2D points can also be calculated with the approach in Section III-G. Full 3D localization using the depth maps provided in that dataset is proposed for future work.

#### F. LLM Integration: Bartender Task

The NICOL platform supports an LLM integration for embodied agents in real-world robotic tasks [53]. We utilize the grounded LLM to implement a robot bartender scenario, leveraging the LLM’s reasoning to connect the robot’s sensory *perception* (identifying glass objects) and robot’s physical *actions* (precise arm manipulation for pouring). Our task facilitates natural interaction with users since the bartender scenario inherently involves social dynamics like recognizing drinkware, interpreting user requests, and responding appropriately. Moreover, bartending tasks have proven to be engaging testbeds for social HRI, as shown in systems like BRILLO [54]. Although our implementation focuses on verbal communication, the task can be extended to incorporate context-driven and multimodal cues, including nonverbal communication. The LLM prompt is in Prompt 1.

As shown in our previous work [4], the concept of procedural memory in LLMs is effective for rapid acquisition

of text-based skills in robots, enabling them to perform adaptive and context-aware reasoning. We apply this concept to implement interactive use cases in the bartender task, where NICOL assists users by serving drinks. While not exhaustive, our scenario highlights the potential of integrating our proposed glass dataset and detector into a robotic application. The scenario consists of the following use cases:

- 1) Action-Object Alignment: The robot accurately maps the beverage pouring action to appropriate glassware in a way that aligns with user expectations and common conventions (e.g., pouring wine into a wine glass).
- 2) Resolving Ambiguities: The robot identifies potential ambiguities in object selection and seeks clarification through user corrective feedback. For example, if the user orders beer and multiple beer cups are available, the robot intuitively prompts the user to specify their preferred glass type, while providing a list of options.
- 3) Context-Driven Action Sequence: The robot uses LLM context memory to maintain continuity across the interaction, ensuring sequential tasks are carried out in logical order. For example, if the user orders a beer and then sequentially asks for another, then the robot can reason about which glass it already filled and choose the appropriate glass for the next pouring action.

#### G. Robot Integration: Beverage Pouring Action

The Bézier curve-based motion planner described in our previous work [55] is utilized to design a pouring motion for the robot bartender task. The action picks up an already open bottle from a fixed position, moves it towards the opening of the glass, pours particles into it, and returns the bottle to its origin. Careful and precise pouring motions are required to prevent spilling particles and to avoid damaging the glasses<sup>1</sup>. The RealSense camera attached to the head of NICOL is used for this application, but it is also possible to use the left or right eye cameras. We use 3D-printed particles to imitate liquid, as it prevents damage to the electrical hardware.

We use the projection approach of the camera model described in Sec. III-A to determine the glass coordinates in 3D space. First, the center of the bounding box bottom edge  $b$  is projected from the image plane onto the tabletop. Thus, the approximately nearest point between the camera origin and the glass bottom is calculated. Since the given coordinate is located at the outer hull of the glass, an offset  $o_i$  is added that takes the height and diameter of the  $i$ -th glass class into account. The center position of the glass opening changes its relation to the detected bounding box bottom center with respect to the glass’s position in the table plane and the height of the glass. The beverage pouring motion has to account for those spatial changes through a second dynamic pouring offset  $p_x$  and  $p_y$  that scales linearly with the specific  $x$ - and  $y$ -axis positions, as shown in Eq. 6:

$$\begin{aligned} p_x &= \epsilon \cdot p_x^{\min} + (\epsilon \cdot \tau) p_x^{\max} \\ p_y &= \gamma \cdot p_y^{\min} + (\gamma \cdot \tau) p_y^{\max}. \end{aligned} \quad (6)$$

<sup>1</sup>No glass was harmed in any of our experiments.

In the prior equation, suitable offsets for the smallest glass class  $p_x^{min}$  and  $p_y^{min}$  are adjusted with the linear scaling factor for the x-axis position  $\epsilon$  and the y-axis position  $\gamma$ . In addition, suitable offsets for the highest glass class  $p_x^{max}$  and  $p_y^{max}$  are added by adjusting them with the linear scaling factor for the glass height  $\tau$ . As the height offset has shown to be influenced by the x- and y-position of the glass in our preliminary experiments, it is multiplied by the scaling factor for the corresponding axis  $\epsilon$  or  $\gamma$ . The pouring offsets for the smallest and highest glass classes  $p_x^{min}$ ,  $p_y^{min}$ ,  $p_x^{max}$ , and  $p_y^{max}$  are determined by placing the smallest and highest glass at the maximum x- and y-position of the pouring workspace and manually tuning the offsets. The primitive scales well to intermediate glass classes. The normalized scaling factors  $\epsilon$ ,  $\gamma$ , and  $\tau$  are calculated as in Eq. 7:

$$\epsilon = \frac{x_n - x_{min}}{x_{max} - x_{min}}, \quad \gamma = \frac{|y_n| - y_{min}}{y_{max} - y_{min}} \quad (7)$$

$$\tau = \frac{h_n - h_{min}}{h_{max} - h_{min}},$$

where the constants  $x_{min}$ ,  $x_{max}$ ,  $y_{min}$ , and  $y_{max}$  are the workspace bounds. Similarly,  $h_{min}$  and  $h_{max}$  are the height of the smallest and highest glass class.

#### IV. EXPERIMENTS AND EVALUATION

We conduct two experiments to evaluate our system. First, we evaluate the performance of our fine-tuned RTMDet glass classifier using our proposed dataset and compare it against state-of-the-art off-the-shelf open-vocabulary object detectors. Additionally, we verify our claim regarding the insufficiency of synthetic data caused by the challenging glass material. Using 3D models of glasses provided within our dataset, we train the network on data rendered by the SuperCaustics framework [25] and include its performance in the evaluation. In the second experiment, we integrate our glass classifier with an LLM-powered embodied agent using NICOL and perform an end-to-end system evaluation in a bartender-like Human-Robot scenario. Next, we show the evaluation metrics and results of each experiment.

##### A. Transparent Object Detection

We evaluate the detector on the test data using the two standard COCO evaluation metrics *Average Precision* and *Average Recall* at IoU from 0.5 to 0.95. Using data from the eye cameras in addition to the captures from the head-mounted RealSense during training slightly increases the performance. The values are calculated for two cases. In the first case (highlighted in Table I), we use the class names described in Sec. III-B as prompts to the open-vocabulary detectors. The shown values of AP and AR are the mean across all glass types (sometimes referred to as *mAP* and *mAR*, respectively). It should be mentioned that existing detectors are unaware of our classification of glass types. There is an ambiguity even for humans, such as the difference between a *whiskey* and a *water* glass. Therefore, in the second case, we treat all instances as a general *drink glass* class. In preliminary experiments, we have also tried other

TABLE I: Comparison of glass detection using zero-shot detectors and transfer-learned lightweight model

Method	AP@[0.5:0.95]		AR@[0.5:0.95]	
	general class	glass types	general class	glass types
OWL-Vit [9]	0.403	0.014	0.671	0.112
G-DINO [7]	0.638	0.102	0.829	0.357
YOLO-World [8]	0.707	0.191	0.844	0.413
*RTMDet-M [42] (synth data [25])	0.224	0.145	0.475	0.290
*RTMDet-M [42] (head RGB only)	0.774	0.713	0.850	0.801
<b>*RTMDet-M [42] (all RGB views)</b>	<b>0.786</b>	<b>0.718</b>	<b>0.854</b>	<b>0.809</b>

\* Our fine-tuned models.

prompts, such as *transparent drink container*, *drinking glass*, or *drinkware*, with worse results. The testing split, as the rest of our data, does not contain any other objects classifiable as a *drink glass* by a human. We have also experimented with image-guided prompting with a template glass image. In such a configuration, the detectors preferred to look for other containers in a similar pose (such as from a top-right view), rather than focusing on the material and type. This caused an even higher false positive rate, with detections of objects like *ceramic cups* in a pose similar to the template.

The higher AP of our pretrained model, even in the general *drink glass* setting, supports our hypothesis that existing open-vocabulary detectors do not grasp the concept of a drink container made from glass. In their latent space, it seems confused with containers made from other materials like *metal cans*. They also cannot distinguish between transparent containers like bottles and drinking glasses, which clearly serve different functions. Taking a further step towards glass type classification makes existing detectors unusable for our scenario. In terms of AR, existing models perform more appropriately. They do not have many false negatives, generally selecting everything that looks like a rounded container.

##### B. Robot Integration: Bartender Task

In this experiment, we use the LLM integration (see Sec. III-F) and beverage pouring motion (see Sec. III-G) to evaluate the performance of our robot bartender task, deployed on the physical robot hardware. We evaluate our system in an end-to-end manner, i.e., a user (an experimenter) interacts with the system in a way that resembles a real-world scenario. Our experimental setup consists of NICOL with six glass objects placed on the table – a *shot glass*, *wine glass*, *whiskey glass*, *water glass*, *beer glass*, and *high beer glass*. Additionally, two plastic bottles – a *wine bottle* and *beer bottle* – are placed on each side of the table and filled with artificial liquid made from 3D-printed round-shaped particles with distinct colors. The pouring workspace has a width of 55 cm and a depth of 35 cm. The positions of the bottles were swapped halfway through the experiment for a fair comparison of the pouring by each arm. The user stands across the table and issues verbal commands through an external microphone, processed via a Whisper integration.

Each interaction consists of the following: the user orders

wine, then a beer, followed by another beer request. We select these interactions to assess the system’s ability to adapt to different beverage types, resolve ambiguities in glassware selection, and ensure a logical action sequence. This follows as a three-phase process: 1) User Order Recognition: The system processes the user’s order and determines the appropriate drink and glassware, through action-object mapping or user preference. For example: if the user says “*I’d like a glass of wine*”, the robot must infer that wine should be poured into a wine glass, 2) Reasoning & Decision Making: Based on user input, the robot reasons about the correct bottle and glass object. Any ambiguity is clarified through user feedback. For example: if the user says “*Can I get a beer for my friend?*”, the robot must decide the correct glassware based on context and prior interactions (which glass is already filled), 3) Action Execution: The robot performs a pouring action correctly to serve the request. We use OpenAI GPT-4o mini and reset the chat after each interaction.

We conducted 35 trials, meaning that 105 pouring actions were executed in total. We use the *Success* rate of the end-to-end pouring action as a metric to evaluate our experiment. However, we report the average error rate in the following categories: *Spill* referring to slight spilling of particles outside the intended glass, *LLM* representing LLM reasoning inaccuracies, and *Detection*, i.e., errors in the glass detector (cf. Fig. 4). Each pouring motion is considered successful if the robot correctly pours the target drink into the target glass without any particles spilled. Our system achieves a success rate of 81% in the pouring action across the two robot arms, while only 3.8% of the cases exhibited minor spills of particles outside the designated glass, demonstrating the suitability of our approach for precise pouring in a bartender task. Our LLM integration showed high resilience in generating correct robot actions across the different use cases: action-object mapping, ambiguity resolution, and logical action sequencing. In 10.5% of the cases, user correction was needed due to reasoning inconsistencies, such as suggesting the wrong glass for a specific drink. However, only one instance was recorded, where the LLM generated an unintended action, mistakenly handing over the object instead of pouring into it, leading to a failure case. Only 4.8% of the cases resulted in failures due to occasional glass detector errors, such as failing to detect the beer glass.

## V. DISCUSSION

As can be seen in Table I, our new benchmark dataset helps to push the current state-of-the-art detector boundaries. Zero-shot models like YOLO-World [8], OWL-Vit [9], and GroundingDINO [7] are widely used for opaque models, but their performance deteriorates on concepts of transparency, glass material, and drink-purposed containers. On the other hand, our processing pipeline allows us to train a lightweight model from the RTMDet [42] family, achieving a high precision with a smaller rate of false positives. This proves the fact that reliable recognition of glass objects is far from achieved with general models. The glass detection system has shown high performance when integrated with the

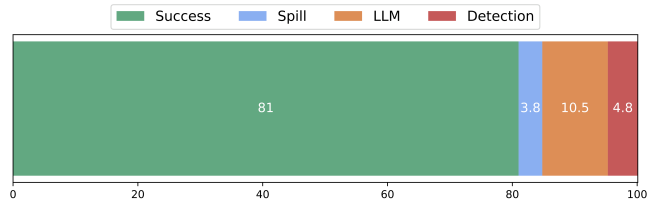


Fig. 4: Results of our end-to-end evaluation of the NICOL robot bartender task, highlighting success rates, spilling occurrences, errors in LLM reasoning, and glass detection.

physical robot, achieving an overall success rate of 81% in a bartending scenario. While the results show that there is still room for improvement in our object detection and robotic action modules, the biggest error source originates from the LLM utilized for high-level planning, making more than 50% of the failure trials. Unfortunately, the glass base keypoints described in Sec. III-E oscillated too much to be applicable in the bartending agent in preliminary experiments. However, 3D planning approaches seem to be a promising area for future research. Our system already shows high performance by planning the motion based on the 2D image perception of the scene. A 3D perception approach, e.g., by utilizing the chalk samples delivered with our dataset for depth estimation, would thus still be capable of improving it.

## VI. CONCLUSION

This paper fills a critical gap in glass object perception by providing a novel real-world dataset. The proposed auto-labeling method reduces the human effort for data aggregation in similar tasks to an absolute minimum. We contribute three distinct open-source repositories: the glass dataset, the glass detector model and experiment code, and the NICOL camera model. The quality of our dataset and camera model allows for very precise projections between different camera perspectives. Our glass detector outperforms SOTA open-vocabulary approaches and reaches a baseline success rate of 81% when integrated with our embodied bartending agent. Future work will include expanding the dataset with more diverse scenes and improving robotic manipulation strategies through enhanced LLM integration. The most interesting next step for the action module is a proprioceptive 3D planning approach that can dynamically adapt to spatial changes, like a user moving a glass while pouring, fostering seamless Human-Robot collaboration in real-world tasks.

## ACKNOWLEDGEMENTS

We thank Svorad Štolc from Photoneo for external consultations of the vision part, Philipp Allgeuer for his contribution to the NICOL robot, and Connor Gäde for his contribution to the bartending experiment.

## REFERENCES

- [1] S. S. Sajjan et al., “ClearGrasp: 3D shape estimation of transparent objects for manipulation,” *ICRA*, pp. 3634–3642, May 2020.
- [2] J. Jiang, G. Cao, J. Deng, T.-T. Do, and S. Luo, “Robotic perception of transparent objects: A review,” *IEEE Transactions on Artificial Intelligence*, vol. 5, pp. 2547–2567, 2023.

- [3] Q. Dai et al., "Domain randomization-enhanced depth simulation and restoration for perceiving and grasping specular and transparent objects," in *ECCV*, 2022, pp. 374–391.
- [4] H. Ali et al., "Robots can multitask too: Integrating a memory architecture and LLMs for enhanced cross-task robot action generation," in *Humanoids*, 2024, pp. 811–818.
- [5] P. Allgeuer, K. Ahrens, and S. Wermter, "Unconstrained open vocabulary image classification: Zero-shot transfer from text to image via CLIP inversion," in *WACV*, Feb 2025, pp. 8206–8217.
- [6] C. Gäde, O. Özdemir, C. Weber, and S. Wermter, "Embodying language models in robot action," in *ESANN*, 2024, pp. 625–630.
- [7] S. Liu et al., "Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection," in *ECCV*, 2023.
- [8] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "YOLO-World: Real-time open-vocabulary object detection," *CVPR*, pp. 16 901–16 911, 2024.
- [9] M. Minderer et al., "Simple open-vocabulary object detection with vision transformers," *ArXiv*, vol. abs/2205.06230, 2022.
- [10] L. Zhu et al., "RGB-D local implicit function for depth completion of transparent objects," in *CVPR*, 2021, pp. 4649–4658.
- [11] T. Li, Z. Chen, H. Liu, and C. Wang, "FDCT: A fast depth completion network for transparent objects," *IEEE RA-L*, 2023.
- [12] H. Xu, Y. R. Wang, S. Eppel, A. Aspuru-Guzik, F. Shkurti, and A. Garg, "Seeing glass: Joint point-cloud and depth completion for transparent objects," in *5th Annual Con. on Robot Learning*, 2021.
- [13] D.-H. Zhai, S. Yu, W. Wang, Y. Guan, and Y. Xia, "TCRNet: Transparent object depth completion with cascade refinements," *IEEE Transactions on Automation Science and Engineering*, 2024.
- [14] T. Tang, J. Liu, J. Zhang, H. Fu, W. Xu, and C. Lu, "RFTrans: Leveraging refractive flow of transparent objects for surface normal estimation and manipulation," *IEEE RA-L*, vol. 9, pp. 3735–3742, 4 2024.
- [15] T. Weng, A. Pallankize, Y. Tang, O. Kroemer, and D. Held, "Multi-modal transfer learning for grasping transparent and specular objects," in *IEEE RA-L*, vol. 5, 2020, pp. 3796–3803.
- [16] J. Jiang, G. Cao, T.-T. Do, and S. Luo, "A4T: Hierarchical affordance detection for transparent objects depth reconstruction and manipulation," *IEEE RA-L*, vol. 7, pp. 9826–9833, 2022.
- [17] P. Murali, B. Porr, and M. Kaboli, "Touch if it's transparent! ACTOR: Active tactile-based category-level transparent object reconstruction," in *IROS*, Dec. 2023, pp. 10 792–10 799.
- [18] R. Yu, W. Ren, M. Zhao, J. Wang, D. Wu, and Y. Xie, "Transparent objects segmentation based on polarization imaging and deep learning," *Optics Communications*, vol. 555, p. 130246, 2024.
- [19] Q. Dai et al., "GraspNeRF: Multiview-based 6-DoF grasp detection for transparent and specular objects using generalizable NeRF," *ICRA*, pp. 1757–1763, 2022.
- [20] J. Kerr et al., "Evo-NeRF: Evolving NeRF for sequential robot grasping of transparent objects," in *CoRL*, 2022.
- [21] Y. R. Wang et al., "MVTrans: Multi-view perception of transparent objects," in *ICRA*, 2023, pp. 3771–3778.
- [22] A. Xompero, R. Sánchez-Matilla, A. Modas, P. Frossard, and A. Cavallo, "Multi-view shape estimation of transparent containers," *ICASSP*, pp. 2363–2367, 2019.
- [23] A. Lukežič, Žiga Trojer, J. Matas, and M. Kristan, "A new dataset and a distractor-aware architecture for transparent object tracking," *Int. Journal of Computer Vision*, vol. 132, pp. 2729–2742, 8 2024.
- [24] E. Xie, W. Wang, W. Wang, M. Ding, C. Shen, and P. Luo, "Segmenting transparent objects in the wild," in *ECCV*, 2020, p. 696–711.
- [25] M. Mousavi and R. Estrada, "SuperCaustics: Real-time, open-source simulation of transparent objects for deep learning applications," *20th IEEE ICMLA*, pp. 649–655, 2021.
- [26] J. Ichnowski, Y. Avigal, J. Kerr, and K. Goldberg, "Dex-NeRF: Using a neural radiance field to grasp transparent objects," in *CoRL*, 2020.
- [27] J. Josifovski, M. Malmir, N. Klarmann, B. L. Žagar, N. Navarro-Guerrero, and A. Knoll, "Analysis of randomization effects on Sim2Real transfer in reinforcement learning for robotic manipulation tasks," *IROS*, pp. 10 193–10 200, 2022.
- [28] 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," in *IEEE RA-L*, vol. 7, no. 3, 2022, pp. 7383–7390.
- [29] X. Chen, H. Zhang, Z. Yu, A. Pipari, and O. C. Jenkins, "ClearPose: Large-scale transparent object dataset and benchmark," in *ECCV*, 2022, pp. 381–396.
- [30] X. Liu, R. Jonschkowski, A. Angelova, and K. Konolige, "KeyPose: Multi-view 3D labeling and keypoint estimation for transparent objects," in *CVPR*, 2020, pp. 11 599–11 607.
- [31] A. Lopes, R. Souza, and H. Pedrini, "A survey on RGB-D datasets," *Computer Vision and Image Understanding*, vol. 222, p. 103489, 2022.
- [32] T.-M. Nguyen et al., "MCD: Diverse large-scale multi-campus dataset for robot perception," *CVPR*, pp. 22 304–22 313, 2024.
- [33] H. Mei et al., "Glass segmentation using intensity and spectral polarization cues," in *CVPR*, June 2022.
- [34] D. Huo, J. Wang, Y. Qian, and Y.-H. Yang, "Glass segmentation with RGB-thermal image pairs," *IEEE Transactions on Image Processing*, vol. 32, pp. 1911–1926, 2023.
- [35] OpenCV, "Open source computer vision library," 2015.
- [36] M. Kerzel, P. Allgeuer, E. Strahl, N. Frick, J.-G. Habekost, M. Eppe, and S. Wermter, "NICOL: A neuro-inspired collaborative semi-humanoid robot that bridges social interaction and reliable manipulation," *IEEE Access*, vol. 11, pp. 123 531–123 542, 2023.
- [37] Z. Li, Y.-Y. Yeh, and M. Chandraker, "Through the looking glass: Neural 3D reconstruction of transparent shapes," *CVPR*, pp. 1259–1268, 2020.
- [38] L. Yang et al., "Depth anything v2," *Advances in Neural Information Processing Systems*, vol. 37, pp. 21 875–21 911, 2025.
- [39] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *ArXiv*, vol. abs/1801.09847, 2018.
- [40] L. A. Klein, *Sensor and Data Fusion Concepts and Applications*. USA: Society of Photo-Optical Instrumentation Engineers, 1999.
- [41] N. Ravi et al., "SAM 2: Segment anything in images and videos," *Arxiv*, vol. abs/2408.00714, 2024.
- [42] C. Lyu et al., "RTMDet: An empirical study of designing real-time object detectors," *ArXiv*, vol. abs/2212.07784, 2022.
- [43] R. Sapkota et al., "YOLO11 to its genesis: A decadal and comprehensive review of the you only look once (YOLO) series," *arXiv*, 2025.
- [44] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh, "CSPNet: A new backbone that can enhance learning capability of CNN," *CVPRW*, pp. 1571–1580, 2019.
- [45] X. Li, W. Wang, X. Hu, J. Li, J. Tang, and J. Yang, "Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection," in *CVPR*, 2021, pp. 11 627–11 636.
- [46] Z. Zheng et al., "Enhancing geometric factors in model learning and inference for object detection and instance segmentation," *IEEE Transactions on Cybernetics*, vol. 52, no. 8, pp. 8574–8586, 2022.
- [47] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *ArXiv*, vol. 2004.10934, 2020.
- [48] H. Zhang, M. Cissé, Y. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *ArXiv*, vol. abs/1710.09412, 2017.
- [49] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: Exceeding YOLO series in 2021," *ArXiv*, vol. abs/2107.08430, 2021.
- [50] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *ICLR*, 2019.
- [51] Z. Wang, Z. Huang, J. Fu, N. Wang, and S. Liu, "Object as query: Lifting any 2D object detector to 3D detection," in *ICCV*, 2023, pp. 3768–3777.
- [52] C. Stauffer and W. Grimson, "Adaptive background mixture models for real-time tracking," in *CVPR*, vol. 2, 02 1999, p. 252 Vol. 2.
- [53] P. Allgeuer, H. Ali, and S. Wermter, "When robots get chatty: Grounding multimodal human-robot conversation and collaboration," in *ICANN*. Springer, 2024, pp. 306–321.
- [54] A. Rossi, S. Rossi, M. D. Maro, and A. Origlia, "BRILLO: Personalised HRI with a bartender robot," *International Journal of Social Robotics*, Mar 2025.
- [55] J.-G. Habekost, C. Gäde, P. Allgeuer, and S. Wermter, "Inverse kinematics for neuro-robotic grasping with humanoid embodied agents," in *IROS*, 2024, pp. 7315–7322.