

Model-Based Underwater 6D Pose Estimation from RGB

Davide Sapienza[†], Elena Govi[†], Sara Aldhaferi*, Marko Bertogna^{††}, Eloy Roura*,
Éric Pairet*, Micaela Verucchi[‡], Paola Ardón*

Abstract—Object pose estimation underwater allows an autonomous system to perform tracking and intervention tasks. Nonetheless, underwater target pose estimation is remarkably challenging due to, among many factors, limited visibility, light scattering, cluttered environments, and constantly varying water conditions. An approach is to employ sonar or laser sensing to acquire 3D data, however, the data is not clear and the sensors expensive. For this reason, the community has focused on extracting pose estimates from RGB input. In this work, we propose an approach that leverages 2D object detection to reliably compute 6D pose estimates in different underwater scenarios. We test our proposal with 4 objects with symmetrical shapes and poor texture spanning across 33,920 synthetic and 10 real scenes. All objects and scenes are made available in an open-source dataset that includes annotations for object detection and pose estimation. When benchmarking against similar end-to-end methodologies for 6D object pose estimation, our pipeline provides estimates that are $\sim 8\%$ more accurate. We also demonstrate the real-world usability of our pose estimation pipeline on an underwater robotic manipulator in a reaching task.

Index Terms—Underwater, manipulation, dataset, pose estimation, computer vision for manipulation.

I. INTRODUCTION

The hazardous nature of underwater environments makes it challenging and critically dangerous for humans to conduct certain underwater intervention tasks. To mitigate this, underwater robots have been widely adopted for intervention tasks during the last decade [1]. A robotic system used for intervention not only requires a mechanised manipulator arm to interact with nearby objects, but also an *object pose estimation* system to track and understand the surroundings.

For an underwater *object pose estimation* system to extract the pose of the surrounding objects reliably, it must cope with changes in light conditions, blurriness, and water visibility. Employing cutting-edge sensors in complicated setups eases the gathering of cleaner three-dimensional (3D) data. Examples of underwater imaging sensors include sonars and lasers; the former can be cluttered, whereas the latter suffers from distortion on the reconstructed cloud proportional to the sensor motion during the scan [1]–[4]. These underwater sensing constraints suppose a remarkable challenge for underwater object tracking and intervention [5].

The lack of on-dry-like 3D sensing has pushed the advancements of computer vision techniques useful for segmentation and localisation underwater [6], [7] as well as red-green-blue (RGB) imagery and Deep Learning (DL) algorithms [8]–[10] for known targets. Current methodologies for object

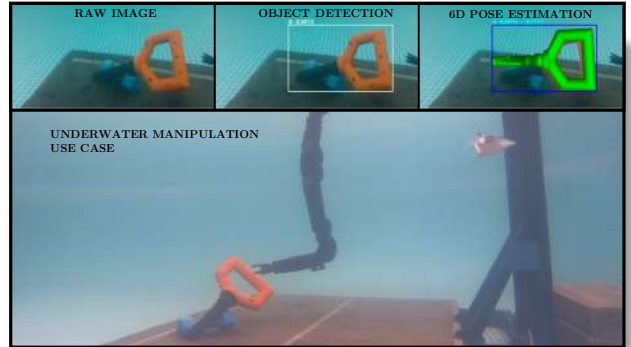


Fig. 1: Illustration of real-time underwater object pose detection. On top, our processing flow: 1) the RGB as input, 2) detect the object and 3) estimate the object pose. Below, example of our pipeline in an underwater manipulation setting.

pose estimation rely on the prior knowledge of the computer-aided-design (CAD) model, which has been proven successful in vehicle tracking [8] and tool detection for intervention applications [10], [11]. Although the promising results in the literature, these are based on datasets and scenarios that target specific applications, thus lacking generalisation.

One of the main challenges for object pose detection underwater is the lack of datasets with variability of underwater environment representations that can potentiate robust learning algorithms. Data collection in uncontrolled underwater environments is considered a laborious and, in some cases, hazardous task. The unavailability of these datasets hinders the training of object pose detection methods that are robust to the low-light and blurry nature of the input data.

On the grounds of the limitations mentioned above, the contribution of our manuscript is twofold. First, we collect and make available a new rich dataset for underwater object pose detection. This dataset consists of randomised simulated underwater scenarios, as well as 10 different real-world underwater and dry scenes across two countries (Italy and UAE). The newly available dataset has RGB and paired depth images with the corresponding CAD models for the objects of interest¹. Second, we present a multi-target underwater pose prediction pipeline that employs a combination of deep Convolutional Neural Networks (CNNs). The architecture employs You Only Look Once (YOLO) version 4 [12] for its proven robustness in detecting object instances in underwater RGB images [13], and an Augmented Autoencoder (AAE) [14] to lift the object's pose estimate to 6D while handling symmetries and partial occlusions. Our method trains in simulation and estimates the target pose in various real environments more efficiently than state-of-the-art methodologies that offer an end-to-end solution to model-based pose estimation. We evaluate the reliability of our proposal using the Benchmark for 6D Object Pose

Manuscript received: February 10, 2023; Revised May 16 and August 8, 2023; Accepted September 7, 2023. This paper was recommended for publication by Editor Pauline Pounds upon evaluation of the Associate Editor and Reviewers' comments.

*TII ARRC. Abu Dhabi, UAE. †Hipert s.r.l and †University of Modena and Reggio Emilia. Modena, Italy. davide.sapienza@unimore.it. Authors thank SpinItalia for supporting the experimental campaign in Italy.

Digital Object Identifier (DOI): see top of this page.

¹Our dataset is available at <https://bit.ly/3LZYvyJ>

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Estimation (BOP) toolkit [15]. We also compare the obtained poses with collected ground truth data and analyse the usability of the method with a robotic arm in a peg-in-a-hole underwater scenario, as shown in Fig. 1.

II. RELATED WORK

A major challenge for underwater object pose estimation is the clarity of sensory input. While imagery suffers from colour degradation and light scattering, 3D perception based on sonars, lasers and customised stereo cameras struggles to acquire accurate and neat point clouds [2], [3], [16]. The lack of on-dry-like 3D sensing motivates underwater object pose estimation to remain an open research topic.

A. Datasets

Underwater datasets for object pose detection are labour-intensive to collect. This section overviews available datasets useful in dry and underwater environments for object tracking and manipulation from multiple types of sensors.

Assuming ideal conditions and perfect acquisition of RGB images from a stereo pair, the corresponding depth information can be computed. However, in reality, ideal conditions are rarely met. For this reason, [1] develops its own underwater 3D laser scanner. The resulting point cloud is used to guide an eight degrees-of-freedom (DoF) fixed-base manipulation system to follow pre-recorded trajectories in real time. In a subsequent demonstration, an underwater vehicle manipulation system (UVMS) uses the same sensor to autonomously pick up an object from the bottom of a water tank. Other methodologies resort to using commercial sensors to overcome the object detection challenge while compensating for the object pose by having a prior of the object shape. For example, [17] presented an underwater dataset called UWHandle of three types of graspable handles collected from a natural seafloor environment. In [17], they project the known handle models on the regions of interest (ROI) detected from a fish-eye camera and test the pose accuracy with AprilTags fiducial [18] markers. [10] also proposes a 3D model-based method with a subset of objects in a controlled water tank setting. While these methodologies propose underwater pose detection, their accuracy is limited to specific environments which limits their extension to manipulation or object tracking in real scenarios.

B. 6D Pose Estimation

Spatial awareness from obtaining 6D pose estimation between a robot and a reference is essential for successful tracking or manipulation tasks. Limited literature exists for object pose estimation in underwater environments.

Authors in [17] proposed a method to detect poses of objects under controlled partial occlusions. Their approach is to regress the 3D poses from monocular silhouettes predicted by a CNN pipeline that uses an associated occlusion mask of the known object 3D model with a transformation vector. This architecture required data acquisition in real environments to estimate the object pose while knowing the target object. Also knowing the 3D model in advance, authors in [8] propose a

real-time 6D relative pose estimation of an Autonomous Underwater Vehicle (AUV) from a single image. This approach uses underwater simulated scenario photos for training. In order to create synthetic images for training, an image-to-image translation network is used to close the gap between rendered and real images. The suggested method predicts the pose of an AUV from a single RGB image that corresponds to the 8 corners of the AUV's 3D model. The resilience and accuracy of the suggested technique are demonstrated in real-world underwater environments with different cameras. While these works motivate the development of underwater object pose estimation methods, they do not cope with uncontrolled object occlusions and blur caused by marine life and diverse conditions in the underwater environment.

Contrary to existing underwater pose estimation methodologies, we make available online a dataset consisting of four different objects across simulated and real environments considering scenes with the objects partly occluded and diversity of backgrounds ranging from homogeneous colours to vegetation. To estimate the object pose, instead of learning an explicit mapping from input images to object poses we propose a pipeline that provides an implicit representation of object orientations defined by samples in a latent space. Thus allowing to train in simulated and generalise to real underwater scenes. Moreover, as explored in Section IV, our proposal shows to be robust to lack of textural surface and symmetrical object geometries.

III. OUR DATASET

This work pursues an underwater multi-object pose estimation for which the training data needs to be diverse and accurate. However, contrary to the vast RGB and depth object datasets for dry environments [19]–[21], there is a lack of datasets for underwater intervention settings. We present a new dataset to push the state-of-the-art in underwater object recognition and pose estimation. Such dataset is then employed in Section IV to compute object poses with our framework.

Our dataset contains RGB annotated frames of four different object categories seen from different points of view alongside the objects' 3D models. The chosen object categories represent common use cases for underwater intervention tasks, such as maintenance or object recovery. The dataset is intended to continue growing with various object representations useful for underwater intervention purposes and to encourage the creation of robust underwater object pose estimation methods.

A. Data Collection and Generation

1) *Data generation*: in order to test our 6D pose estimation framework proposed in Section IV, we designed four different objects commonly used for underwater intervention tasks. These objects represent symmetrical, asymmetrical and texture-less shapes. Fig. 2 illustrates our object categories. Using Unity and the CAD model of our objects, we generate a total of 33,920 simulated scenes of dimension 640×480 with different homogeneous and heterogeneous backgrounds, including object occlusions and various illuminations. Fig. 2 showcases some of our simulated scenes.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.



Fig. 2: Example of our dataset, including real (white font) and simulated scenes (black font). *UW* describes underwater.

2) *Collection in real environments*: we 3D printed our objects to harvest data in real scenes. The dataset is collected using a d455 realsense inside a waterproof container. The camera simultaneously records both, RGB and depth images at 640×480 resolution at 30 frames per second. For the development of the framework in Section IV, we solely use the RGB data for training. While we do not use the depth information for our method, we provide the depth frames in our available dataset for the use of the community. Using the previously described camera setup, we record video sequences of each of the four object categories in 10 different setups, both, individually and using various categorical combinations. The average scene sequence contains 180 RGB frames with corresponding depth data. The different scenes are recorded across two different countries (Italy and UAE), in different real underwater and dry environments. For our underwater scenes, we consider environments, such as partial object occlusions from marine life, shadows and changing lighting conditions, low visibility at night coped with LED illumination and heterogeneous backgrounds. Fig. 2 shows examples of our real underwater scenarios (captioned ‘UW’ in white font). Per recording, we freely rotate the object inside the camera field of view at varying distances as far as $3m$ from the target, as bounded by hardware limitations.

B. Object Recognition Annotation and Processing

1) *Annotation*: given the diversity of environments in our dataset we need an efficient yet high-quality annotation pipeline to define our objects of interest in the images. We annotated a total of 87,100 real RGB images for two-dimensional (2D). Examples of the bounding boxes are shown in Fig. 2. Our annotation process consists of three stages. First, a human annotator labels approximately 1% of the data

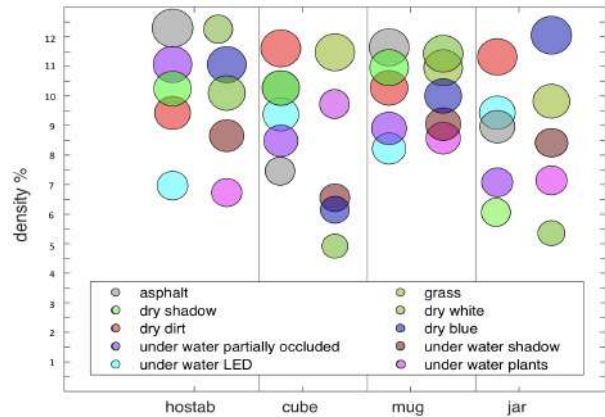


Fig. 3: Dataset collection density across scenes.

across scenes, including partial views of the objects. Second, we designed an auto-labelling tool based on YOLOv4 [12] to create the bounding boxes for the rest of the data based on the 1% of manually annotated data. Finally, three different human annotators checked the created bounding boxes in a sequential fashion to ensure the quality of the annotated region of interest.

2) *Dataset Statistics*: we consider an important feature of our dataset the variability of scenes, particularly the contribution of underwater imagery. Fig. 3 shows the four objects across our 10 scenes and the percentage of RGB images per scene. As seen from the data density, the collected data is spread across different settings. The real underwater collected images represent 42.5% of our dataset, being collected in 4 different underwater conditions. The other 57.5% are image sequences collected from 6 on dry scenes. Fig. 4 shows the location distribution of the centre of our annotated bounding boxes on the RGB images. As shown in the plots, there are some concentrations at the centre of the 640×480 image, with some bounding boxes located across most of the pixels.

C. Ground Truth Extraction for Pose Estimation

Exclusively for sim-to-real evaluation purposes, to obtain the ground truth of the object pose, we attached an April-Tag [18] bundle to the target objects. Thus, the bundle has a fixed map to the centre of geometry of the object. As illustrated in Fig. 5, given that we know the object base frame by its 3D design, O_p , and we can detect the bundle pose, B_p we calculate offline the transformation between the object and the bundle, T_B^O . Using this information we are able to extract the ground pose and estimate the error of our detection (detailed in Section IV-C3).

In general our collected dataset displays features that promote detection robustness across different scenes, including dry and underwater environments, as well as a high-quality set of annotations for future proposals on learning methodologies.

IV. OBJECT POSE ESTIMATION

Obtaining reliable 3D data underwater is difficult given the environment and sensing limitations. Given the aforementioned limitations, we propose a pipeline composed of 2D object detection and 6D pose estimation on the RGB image.

A diagram of our pipeline is illustrated on Fig. 6. On the detected object bounding box, we then estimate the object 6D pose. Namely, we create a pipeline based on YOLOv4 for 2D object detection (as detailed in Section IV-A) and the AAE for 6D pose estimation (as detailed in Section IV-B). The pipeline is multi-object because the YOLOv4 is multi-object and selects the correspondent object's AAE training (one for each class). Our pipeline is designed to find the best trade-off between computational efficiency and accuracy on real-time input while proving to be robust for detection and pose estimation in dry and underwater environments. Opting for this pipeline offers a significant advantage because it can be completely trained in simulation and only requires labelled data for the YOLOv4, while generalising on real scenes (see Section IV-C3). Our dataset is readily available and includes RGB images and 2D labels containing bounding box information of the depicted objects. Conversely, no dataset is necessary for AAE since it self-builds from the CAD model. Nevertheless, synthetic labelled 6D datasets and real labelled 6D datasets were produced and used during the validation of our method (see Sections IV-C1, IV-C2, IV-C3). The significance of 6D labelled datasets extends to being applicable

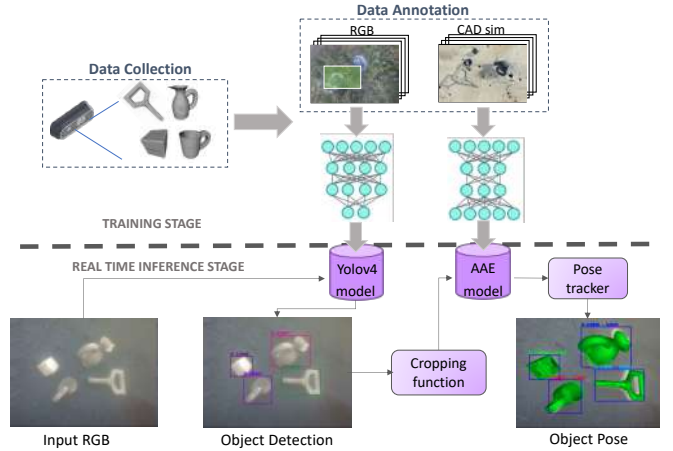


Fig. 6: Given an RGB image as input, the first step consists of 2D detection and classification with multi-object YOLOv4. Given cropped images and their classes, they become the input of one of the four AAE training, one for each object. As output, each training returns a rotation matrix R and a translation matrix t .

to almost all other underwater pose estimation techniques that necessitate 6D labels. Moreover, differently from similar methodologies for 6D pose estimation, our proposed pipeline proves to be unaffected by the object perception point-of-view, self occlusions, and symmetric geometries as illustrated in Section IV-D.

A. 2D Object Detection

For RGB based pose detection methods, accurate 2D detection is essential for the subsequent pose estimation. To train object detection we consider various underwater scenes to enrich the collection of features, as detailed in Section III-A.

In the context of deep learning, YOLOv4 has demonstrated to efficiently perform in a heterogeneous environment, with an excellent trade-off between accuracy and latency [22]. The training used only synthetic data. As detailed in Section III-A, our synthetic data considers different backgrounds, light colours and directions, in order to overcome issues of the underwater scenario. Moreover, YOLOv4 applies data augmentation to the original data. YOLOv4 uses a saturation of 1.5, exposure of 1.5, hue 0.1, and a mosaic effect: it combines multiple images into a single image. We use YOLOv4 architecture as originally proposed in [22] to extract a bounding box with the object features in real-time.

Up to this stage in our pipeline, we have a detected object and the representing class as the output from YOLOv4. At this point, we still need to estimate the detected object 6D pose.

B. 6D Pose Estimation

After YOLOv4, a method for 6D prediction should be chosen. A major issue that usually affects 6D pose estimation is pose ambiguity. Usually, these ambiguities arise due to symmetries or self-occlusion of the objects. For example, when the cup's handle is occluded, the cup's pose is ambiguous. Therefore choosing a pose ambiguity invariant approach has

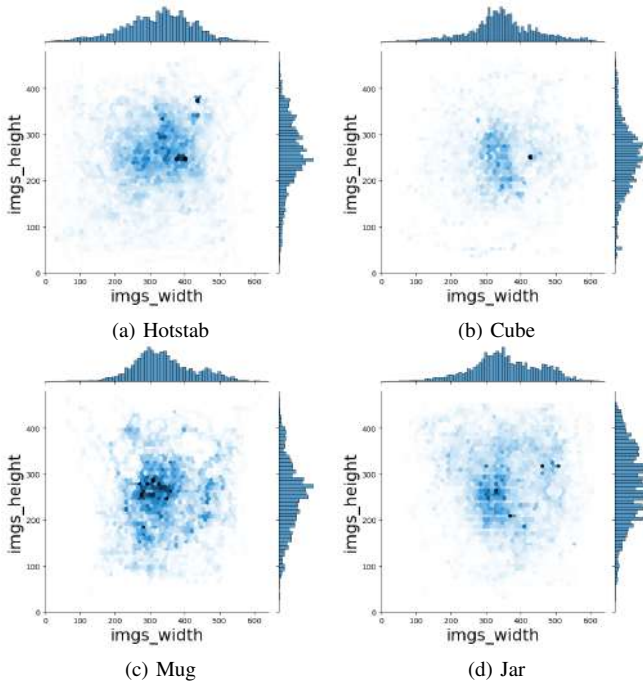


Fig. 4: Annotations representing object locations in the image.

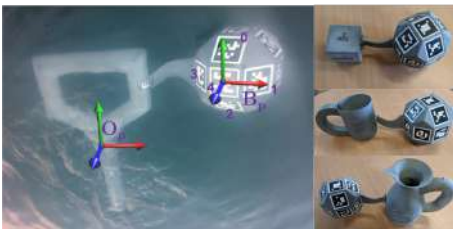


Fig. 5: On the right, example of the annotated ground pose estimation on the hotstab. On the left, rest of the objects.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

been revealed to be essential. In addition, given the underwater scenario, the method should be robust to light changes and blurring effects in real-time. For these reasons, an Autoencoder (AE) structured model is chosen: AAE [14]. On the contrary, alternative approaches [23]–[25], involve regressing local 2D-3D correspondences between the image and CAD model, followed by the application of a Perspective-n-Point (PnP) algorithm to derive the object’s 6D pose. However, these methods present two limitations in our scenario. Firstly, they require clear images to distinguish visual features, which poses a challenge in our underwater environment. Secondly, these methods are not pose invariant, and as such, are not suitable for our scenarios where 3 out of 4 objects are symmetrical. On the other hand, AAE was proposed by [14] as a symmetry invariant approach. This is due to the property of representing the orientation not on a fixed parametrisation, but on the appearance of an object. Our specific underwater dataset, achieves higher performance scores than other widely used models, as shown in Section IV-C, demonstrating to be robust to underwater scenarios. Furthermore, AAE is suitable to real-world use case requirements, as shown in Table IV where inference times are computed.

We explored different methods, as detailed in Section IV-D, and found out that AAE works better in our environment because of the properties outlined above. Since rotations live in a continuous space, it seems natural to directly regress a fixed rotation in the three-dimensional rotation group (SO(3)), like in EfficientPose [26]. However, pose ambiguities and representational constraints can introduce convergence issues [24]. AAE solves this problem by discretising the SO(3) space and then moving the problem from regression to classification. Furthermore, an advantage of AAE is that it does not depend on real data collection and annotations for training. Instead, in the training stage, AAE creates its own dataset by randomising object poses, taken from the CAD model, and Visual Object Classes (VOC) background real images, to bridge the simulation to real images gap. To guarantee the AAE performance on dry and underwater scenarios, we (i) add background images from our collected dataset in real scenarios (detailed in Section III-A2), and (ii) customise the data augmentation for training stage. AAE applies data augmentation at the pre-processing stage when it generates its dataset, allowing the method to be trained solely in simulated data. Table I reports the AAE data augmentation hyperparameters.

A pose of a 3D object in this proposal is represented by the 4×4 matrix $\mathbf{P} = [\mathbf{R}, \mathbf{t}; \mathbf{0}, 1]$, where \mathbf{R} is the rotation represented by a 3×3 matrix and \mathbf{t} is the translation represented by a 3×1 vector. \mathbf{P} transforms a 3D point x_m in the model coordinate system to a 3D point x_c in the camera coordinate system: $\mathbf{P}[x_m; \mathbf{1}] = \mathbf{R}x_m + \mathbf{t} = [x_c; \mathbf{1}]$

AAE is based on AE structure, aiming to obtain an image representation in a low-dimensional Euclidean space. In detail, first, AAE applies a random augmentation f_{aug} to input x and reconstructs the original image. Then, an encoder-decoder training reconstructs the original input. The parameters are learned during the backpropagation phase, based on the per-sample loss: $l = \sum_{i \in D} \|x_i - \hat{x}_i\|_2$. After this initial training, a *codebook* is created by generating a latent representation

	Box	Cup	Jug	Hotstab
Data Augmentation Hyperparameters				
Perspective Transform				
Crop And Pad	✓	✓		✓
Affine	✓	✓	✓	✓
Coarse Dropout	✓	✓		✓
Gaussian Blur			✓	✓
Invert	✓	✓	✓	✓
Multiply	✓	✓	✓	✓
Contrast Normalization			✓	✓
Square Occlusion	0.6	0.4	0.4	0.4
Architecture-Hyperparameters				
Learning Rate	$2e-4$	$2e-4$	$2e-4$	$2e-4$
Optimizer	Adam	Adam	Adam	Adam
Latent Space Dimension	256	256	128	256
Epochs	50,000	70,000	70,000	70,000
Batch Size	32	32	64	64

TABLE I: Hyperparameters chosen for each object’s training.

$z_i \in \mathbb{R}^l$ of each one of the n object views, and their correspondent \mathbf{P}_i matrices. The AAE training must be done for each one of the objects in the dataset. During the test phase, AAE is preceded by the 2D detector and receives as input the already detected and cropped image. The image goes through the encoder which gives its latent space features. Then, the cosine similarity is computed between the input latent representation code and all codes from the codebook. The highest similarity is chosen and the corresponding rotation matrix from the codebook is returned as 3D object orientation.

For the entire pipeline, five training are needed: one multi-object YOLOv4 and four different AAE, one for each object. Despite this, the pipeline is considered multi-object since in inference, given an image, is able to reconstruct the pose of each observable object in time explained in Table IV. One limitation of AAE is that the number of objects in a single scene affects the inference time. However, in the case of four objects the achieved performance respects predefined time constraints. Deep Neural Networks performance, as AAE, is dependent on hyperparameters that determine the network structure. Finding the right hyperparameters is fundamental to ensuring good performance. AAE relies on many hyperparameters [14], which are divided into two groups: those modifying the structure and optimization of the network (such as learning rate, latent space dimension, batch normalization), and those acting on the data augmentation (such as occlusion percentage, inversion, multiplication, drop, Gaussian blurring addition). Given the large hyperparameter space, we use heuristics to only explore the most promising configurations. In our case, we ran experiments, on a total of 40 different configurations to find the best hyperparameter setting.

After different training for architecture network and data augmentation hyperparameters optimization, we choose the best set per each object training (see summary in Table I).

C. Method Evaluation

Our evaluation is threefold. First, using the simulated data per object in our dataset, we provide a baseline of our method’s performance using part of the BOP metrics [15] and Completion over Union (CoU) (see Section IV-C1). Second, using the BOP metrics, we benchmark against current literature. We choose YOLO-6D [24] and EfficientPose [26], as they also are

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

deep learning methods that generalise mappings from images to objects. Nonetheless, as they do not claim to bridge training in simulation to real scenarios performance, the evaluation is done in underwater simulated data (see Section IV-C2). Finally, to test the robustness of our training in simulation and its performance in real scenes, we calculate the error pose using the ground truth (see Section IV-C3). This error estimation provides a baseline for future work to compare our method’s performance and dataset.

For the evaluation in Section IV-C1 and IV-C2, we use three of the widely popular ([24], [26], [27]) BOP toolkit metrics:

- **Mean Average Precision (mAP)** is a 2D bounding box detection metric $mAP(C, \Gamma, R) = avg_{c \in C} AP(c, \Gamma, R)$, where $AP(c, \Gamma, R)$ is the area under the curve, Γ is the set of thresholds used in Intersection over Union (IOU) scores, c is the class, and R is the set of the discretised recall values;
- **Average distance to the model point (ADD)** given the model \mathcal{M} , the estimated pose $\hat{\mathbf{P}}$ and the ground-truth \mathbf{P} $e_{ADD} = avg_{x \in \mathcal{M}} \|\hat{\mathbf{P}}\mathbf{x} - \mathbf{P}\mathbf{x}\|$;
- **Average distance to the closest model point (ADI)** if the model \mathcal{M} has indistinguishable views, then $e_{ADI} = avg_{x_1 \in \mathcal{M}} \min_{x_2 \in \mathcal{M}} \|\hat{\mathbf{P}}\mathbf{x}_1 - \mathbf{P}\mathbf{x}_2\|$. e_{ADI} yields relatively small errors since it does not consider model point distances, but the distance to the closest model diameter.

Moreover, we compute the CoU error, which compares the predicted pose and ground truth masks from the CAD model.

1) *Dataset Baseline with BOP given that the benchmark in Section IV-C2 is produced with simulated data, we provide a performance baseline per object in our dataset with synthetic images to have a fair and uniform comparison baseline.*

For this section, we use 500 synthetic underwater scenes in various settings as detailed in Section III-A. In our dataset, the box, cup, and hotstab are symmetrical, while the jug is not. To evaluate AAE’s performance, we choose e_{ADI} for the symmetrical set and e_{ADD} for the asymmetrical object. Our results with different thresholds of correctness for e_{ADI} are illustrated in Table II. Table II introduces a baseline for our new underwater dataset. For the jug, we computed the e_{ADD} , obtaining 29.2% for $k_m = 0.1$; 56.6% for $k_m = 0.2$, and; 68.6% for $k_m = 0.3$. Nonetheless, these metrics are not pose-ambiguity invariant. Consequently, the objects may present ambiguous poses. This is due to their symmetrical views. For example, the box is texture-less and has two symmetrical axes, thus being difficult to distinguish between its faces. For these cases, we also calculate the CoU on the same set of images. The results are shown in Table III. Contrary to the results in Table II, CoU does not penalise symmetric poses, since it uses only segmentation masks. A clear example is the different pose estimation performance on the box object between e_{ADI} from

Objects	$k_m = 0.1$	$k_m = 0.2$	$k_m = 0.3$
Box	21.6%	51.0%	60.4%
Cup	55.0%	77.8%	85.8%
Jug	72.2%	86.6%	93.0%
Hotstab	44.0%	64.4%	73.8%
Average Perc.:	48.2%	69.95%	78.25%

TABLE II: Recall percentages based on e_{ADI} .

Table II and e_{COU} in Table III.

Additionally, in terms of real-time performance, Table IV shows the inference times of our proposal per detection stage and end-to-end pipeline. These results use the same validation set, thus the statistics of 500 inference times. The first time is discarded to not consider the weight load and initialisation memory levels. Inference times are computed on a Xavier AGX, Jetpack 5.0.2.

In general, since we propose a new dataset it is hard to compare the resulting values to other 6D pose datasets.

2) *Benchmark with Literature to fairly compare with YOLO-6D and EfficientPose, we train and test all the methods using simulated underwater data (see Section III-A). We select a subset of 250 images, from the 500 extracted in Section IV-C1, corresponding to various synthetic scenes of hotstab (symmetric object) and the jug (asymmetric object).*

Training: We iterated on different combinations of batch sizes, learning rates and Adam momentum as optimiser for the training of YOLO-6D and EfficientPose. The performance reported in this section for YOLO-6D is achieved with batch size equal to 32, an adaptive learning rate (it starts with 0.0001 for 10,000 epochs. For EfficientPose, the best performance is achieved with a batch size equal to 1, a learning rate of 0.0001 and 500 epochs. For our proposal, we train as detailed in Section IV-B. However, to ensure comparability with the other two methods, we opted for lighter versions by selecting ϕ equal to 0 as scaling hyperparameter.

Results: Table V shows the results of comparing YOLO-6D, EfficientPose and our method using the BOP metrics. In the case of EfficientPose, contrary to the high performance on the Linemod Dataset, it achieves poor results in our dataset. We observe that EfficientPose’s 2D detection achieves only 77%, consequently compromising the 6D pose estimation. [28] presents an exhaustive study on the potential causes for object detection failure in EfficientPose.

3) *Real data error pose estimation:* to test the robustness of our method’s performance in real scenes, despite being trained in simulation, we calculate the error pose using the ground truth data (see Section III-C). Moreover, this error

CoU error	$\theta = 0.3$	$\theta = 0.5$	$\theta = 0.7$
Box	94.6%	100%	100%
Cup	94.2%	99.4%	100%
Jug	79.6%	97.8%	99.6%
Hotstab	20.2%	65.6%	92.6%
Average Percentages:	72.15%	90.57%	98.05%

TABLE III: Recall percentages based on e_{COU} .

		YOLOv4	AAE	END TO END	
		FPS	FPS	latency (ms)	FPS
single object	min	3.57	8.14	403.13	2.48
	max	2.93	6.37	497.78	2.01
	avg	3.42	7.44	426.51	2.34
multi objects	min	3.52	1.83	830.61	1.20
	max	2.76	1.28	1,144.46	0.87
	avg	3.27	1.72	885.66	1.13

TABLE IV: Total inference time in milliseconds and FPS, as the sum of pre-processing, inference, and post-processing times, for both single object and multi objects scenarios with the corresponding Frames Per Second (FPS) value.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

	ADD	ADI	mAP
Symmetric Object (hotstab)			
EfficientPose	1.32%	8.65%	75.21%
YOLO-6D	9.58%	36.71%	77%
YOLOv4+AAE	11.4%	44.0%	99%
Asymmetric Object (jug)			
EfficientPose	23%	54.8%	73.41%
YOLO-6D	26.50%	58.44%	81.6%
YOLOv4+AAE	29.2%	78.2%	99.5%

TABLE V: EfficientPose, YOLO-6D, and YOLOv4+AAE comparison using BOP metrics for the hotstab and the jug objects. The ADD and ADI recall is computed with a threshold $k_m = 0.1$. The mAP is computed with a 0.5 of IOU.

pose estimation offers a baseline for evaluating future 6D pose estimation methods that require real ground truth labelled data for training. From the 87,100 collected images in our dataset, 83,701 contained the bundle described in Section III-C. From this subset, we selected 20% of the data to process the translation and rotation error in Table VI. Specifically, we calculate the Euclidean distance between the position and orientation estimation error of our proposed pipeline in Fig. 6 with respect to the ground truth pose extracted from the bundle holder. The evaluation subset is available in our website².

As seen by the error estimation, the position error median is approximately 20mm while the normalised orientation error (between 0° and 360°) is around 30° for the different objects. It is worth noting that higher error values correspond to the rotation error rather than the translation one. Particularly, this difference is more evident for the cube and mug objects. This result can be attributed to the texture-less and symmetrical nature of objects. While the pose estimation has room for improvement, particularly for the object rotation, the values are consistent with the original AAE proposal in [14], demonstrating the method’s consistency across symmetric objects.

Scenes	Hotstab		Cube		Mug		Jar	
	t_e	r_e	t_e	r_e	t_e	r_e	t_e	r_e
Asphalt	22.3	25	23.7	23.5	21.1	25	23.3	21
Dry shadow	16	23	28.2	21.2	15	24.3	20.1	22.4
Dry dirt	22.1	29	24.2	29	21.1	24	19	24
UW partly occluded	21	20.9	24	30	19	25	20	24
UW with LED	20	22	29.8	31.4	20	27	21	21.3
Grass	20	27	25	32.1	20	28.1	20	27.1
Dry white	15	25	23.6	29	22	21.8	24.2	23
Dry blue	17	26.3	18	25	19	24.3	22.7	23
UW with shadow	20	20.3	24	25.1	20	23	19.9	20

TABLE VI: Median data for pose error representation of our pipeline using the pose extracted from AprilTag bundles as ground truth for each of our settings. Translation error, t_e in mm and rotation error, r_e in degrees.

D. Underwater Manipulation Use Case

In order to test the robustness of our object pose estimation pipeline we tested it with our underwater manipulation setup. Our testing setup consisted of a static robotic arm, Reach Bravo with 7 DoF on a bench structure that served as the skeleton to hold the arm inside our 2m depth pool. We placed a surface with a solid background and attached a weight to our target object to avoid pose oscillations due to the object’s

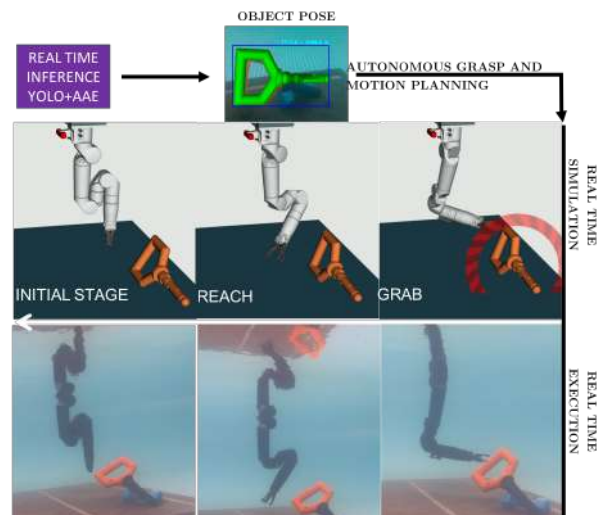


Fig. 7: Our proposed pipeline in an underwater manipulation example. Given the object pose of an object, we autonomously generate (i) grasp configurations on the objects and (ii) a motion plan to reach and grab the target object.

buoyancy. We attached a realsense d455 camera on the arm’s skeleton to have a complete view of the arm and the object. From the camera we extract solely the RGB data as input to our pipeline described in Sections IV-A and IV-B.

Once we have obtained the 6D pose from the pipeline proposed in this letter, we use our in-house developed assistant for manipulation tasks to aid in taking authoring high-level manipulation commands. The assistant then extracts and concatenates in a single task all required reaching goals and motion plans. In the example of reaching for an object, the manipulation assistant takes the semantic label of the detected object and allows the user to select a series of tasks. For the pipeline proof-of-concept, we decided to use a single object in the scene to ease the manipulation goal decision-making process. Using our user interface, we select Open Gripper and Reach Hotstab (see illustration in Fig. 7). Out of 10 different poses, inside the arm’s working space, 6 resulted in accurate poses for the arm to successfully reach³. The failed trials are attributed to flickering in the AAE pose estimation. These oscillations are the result of unavoidable reflections from the water surface due to the shallowness of our testing environment. Nonetheless, for future improvements, a pose tracker and filtering could be put in place to ensure grabbing the object. Moreover, in some of the frames, we noticed that the 2D object detection model would detect some outlier objects in the scene. To ensure the manipulator did not plan trajectories to false positive poses, we selected a priori the target object. However, the 2D object detection model could be further improved by including background data with similar geometries that do not belong to our objects of interest.

V. CONCLUSIONS AND FUTURE WORK

We propose a method for underwater pose estimation using RGB data that copes with diverse environments. We summarise our contribution as twofold: (i) a publicly available

²Our dataset is available at <https://bit.ly/3LZYvyj>

³Experiments: <https://www.youtube.com/watch?v=xPAbwh5JGM>

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

dataset consisting of 4 objects in 10 different real scenes and annotations for object detection and pose estimation, as well as, (ii) an object pose estimation pipeline that adapts to different light conditions and heterogeneous backgrounds in real scenes while being trained in simulated data.

Our dataset consists of some challenging objects with a symmetrical shape and poor texture. Regardless of such object characteristics, our proposed method outperforms alternative model based 6 Degrees of Freedom (6D) pose estimation methods. Our results suggest potential for generalisation with symmetrical, asymmetrical and texture-less object representations. We attribute this performance to the robustness achieved in the AAE through our hyperparametrisation stage. We successfully used our proposed pipeline for a reaching task using an underwater manipulator demonstrating its robustness. It is also worth mentioning that our pipeline has some inherent limitations given the adopted self-supervised learning approach. One of these limitations is the scalability of the pipeline as target objects in the scene increase, since there should exist a AAE model per object. However, we envision using our pipeline for manipulation purposes in which there are limited target objects at a close-reaching range.

This work opens multiple avenues for further research, such as facilitating the object generalisation through an online 3D shape estimation algorithm, and stabilising the pose detection with a pose tracker and filtering to avoid oscillations in the pose estimation. We enable such future work by making our proposed pipeline and dataset publicly available.

REFERENCES

- [1] A. Palomer, P. Ridao, D. Youakim, D. Ribas, J. Forest, and Y. Petillot, "3d laser scanner for underwater manipulation," *Sensors*, vol. 18, no. 4, p. 1086, 2018.
- [2] H. Sakai, J. Akizono, S. Yasumura, and Y. Takahashi, "Underwater laser viewing system and its application," in *Proceedings of 1998 International Symposium on Underwater Technology*, pp. 161–167, IEEE, 1998.
- [3] P. Shu-guang, D. Jia-hao, W. Su-Juan, and S. Dong-ning, "Research on the underwater laser fuze target detecting technique," in *Proceedings. ICCEA 2004. 2004 3rd International Conference on Computational Electromagnetics and Its Applications, 2004.*, pp. 352–355, IEEE, 2004.
- [4] W. Czajewski and A. Sluzek, "Development of a laser-based vision system for an underwater vehicle," in *ISIE'99. Proceedings of the IEEE International Symposium on Industrial Electronics (Cat. No. 99TH8465)*, vol. 1, pp. 173–177, IEEE, 1999.
- [5] P. Sanz, A. Peñalver, J. Sales, J. J. Fernández, J. Pérez, D. Fomas, and J. García, "Multipurpose underwater manipulation for archaeological intervention," *Instrumentation Viewpoint*, no. 18, pp. 50–51, 2015.
- [6] M. J. Islam, C. Edge, Y. Xiao, P. Luo, M. Mehtaz, C. Morse, S. S. Enan, and J. Sattar, "Semantic segmentation of underwater imagery: Dataset and benchmark," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1769–1776, IEEE, 2020.
- [7] B. Yu, J. Wu, and M. J. Islam, "Udepth: Fast monocular depth estimation for visually-guided underwater robots," *arXiv preprint arXiv:2209.12358*, 2022.
- [8] B. Joshi, M. Modasshir, T. Manderson, H. Damron, M. Xanthidis, A. Q. Li, I. Rekleitis, and G. Dudek, "Deepurl: Deep pose estimation framework for underwater relative localization," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1777–1784, IEEE, 2020.
- [9] G. Billings, M. Walter, O. Pizarro, M. Johnson-Roberson, and R. Camilli, "Towards automated sample collection and return in extreme underwater environments," *arXiv preprint arXiv:2112.15127*, 2021.
- [10] M. Jeon, Y. Lee, Y.-S. Shin, H. Jang, and A. Kim, "Underwater object detection and pose estimation using deep learning," *IFAC-PapersOnLine*, vol. 52, no. 21, pp. 78–81, 2019.
- [11] G. Billings and M. Johnson-Roberson, "Silhonet: An rgb method for 6d object pose estimation," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3727–3734, 2019.
- [12] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-YOLOv4: Scaling cross stage partial network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13029–13038, June 2021.
- [13] L. Chen, M. Zheng, S. Duan, W. Luo, and L. Yao, "Underwater target recognition based on improved yolov4 neural network," *Electronics*, vol. 10, no. 14, p. 1634, 2021.
- [14] M. Sundermeyer, Z. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3d orientation learning for 6d object detection from RGB images," *CoRR*, vol. abs/1902.01275, 2019.
- [15] T. Hodaň, F. Michel, E. Brachmann, W. Kehl, A. Glent Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, and C. Rother, "BOP: Benchmark for 6D object pose estimation," *European Conference on Computer Vision (ECCV)*, 2018.
- [16] S. Aldhaheri, G. De Masi, È. Pairet, and P. Ardón, "Underwater robot manipulation: Advances, challenges and prospective ventures," in *OCEANS 2022-Chennai*, pp. 1–7, IEEE, 2022.
- [17] G. Billings and M. Johnson-Roberson, "Silhonet-fisheye: Adaptation of a roi based object pose estimation network to monocular fisheye images," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4241–4248, 2020.
- [18] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE international conference on robotics and automation*, pp. 3400–3407, IEEE, 2011.
- [19] S. Hinterstoisser, C. Cagniard, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient response maps for real-time detection of textureless objects," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 5, pp. 876–888, 2011.
- [20] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv preprint arXiv:1711.00199*, 2017.
- [21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, pp. 740–755, Springer, 2014.
- [22] A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, 2020.
- [23] M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," *CoRR*, vol. abs/1703.10896, 2017.
- [24] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," *CoRR*, vol. abs/1711.08848, 2017.
- [25] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," *arXiv preprint arXiv:1809.10790*, 2018.
- [26] Y. Bukschat and M. Vetter, "Efficientpose: An efficient, accurate and scalable end-to-end 6d multi object pose estimation approach," *CoRR*, vol. abs/2011.04307, 2020.
- [27] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *CoRR*, vol. abs/1711.00199, 2017.
- [28] E. Govi, D. Sapienza, C. Scribano, T. Poppi, G. Franchini, P. Ardón, M. Verucchi, and M. Bertogna, "Uncovering the background-induced bias in rgb based 6-dof object pose estimation," 2023.